34 HANDS-ON
Network Simulator labs

2+ HOURS
of video training

600+
practice exam questions

100+
online practice exercises

Official Cert Guide Library

Advance your IT career with hands-on learning

CCNA 200-301

WENDELL ODOM,
CCIE® NO. 1624 EMERITUS

ciscopress.com

CCNA
200-301

Official Cert Library

WENDELL ODOM, CCIE No. 1624 Emeritus

Cisco Press

Pearson

Where are the companion content files?

Register this digital version of CCNA 200-301 Official Cert Guide, Volume 1 to access important downloads.

Register this eBook to unlock the companion files. Follow these steps:

1. Go to **ciscopress.com/account** and log in or create a new account.
2. Enter the ISBN: **9780135792735** (NOTE: Please enter the print book ISBN provided to register the eBook you purchased.)
3. Answer the challenge question as proof of purchase.
4. Click on the "Access Bonus Content" link in the Registered Products section of your account page, to be taken to the page where your downloadable content is available.

This eBook version of the print title does not contain the practice test software that accompanies the print book.

You May Also Like—Premium Edition eBook and Practice Test. To learn about the Premium Edition eBook and Practice Test series, visit **ciscopress.com/practicetest**

Pearson

Where are the companion content files?

Register this digital version of
CCNA 200-301 Official Cert Guide, Volume 2
to access important downloads.

Register this eBook to unlock the companion files. Follow these steps:

1. Go to **ciscopress.com/account** and log in or create a new account.
2. Enter the ISBN: **9781587147135** (NOTE: Please enter the print book ISBN provided to register the eBook you purchased.)
3. Answer the challenge question as proof of purchase.
4. Click on the "Access Bonus Content" link in the Registered Products section of your account page, to be taken to the page where your downloadable content is available.

This eBook version of the print title does not contain the practice test software that accompanies the print book.

You May Also Like—Premium Edition eBook and Practice Test. To learn about the Premium Edition eBook and Practice Test series, visit **ciscopress.com/practicetest**

Official Cert Guide

Advance your IT career with hands-on learning

Hands-On Labs

Review Exercises

Config Checklists

CCNA 200-301

Volume 1

WENDELL ODOM,
CCIE® NO. 1624 EMERITUS

ciscopress.com

CCNA 200-301, Volume 1
Official Cert Guide

In addition to the wealth of updated content, this new edition includes a series of free hands-on exercises to help you master several real-world configuration and troubleshooting activities. These exercises can be performed on the CCNA 200-301 Network Simulator Lite, Volume 1 software included for free on the companion website that accompanies this book. This software, which simulates the experience of working on actual Cisco routers and switches, contains the following 21 free lab exercises, covering topics in Part II and Part III, the first hands-on configuration sections of the book:

1. Configuring Local Usernames
2. Configuring Hostnames
3. Interface Status I
4. Interface Status II
5. Interface Status III
6. Interface Status IV
7. Configuring Switch IP Settings
8. Switch IP Address
9. Switch IP Connectivity I
10. Switch CLI Configuration Process I
11. Switch CLI Configuration Process II
12. Switch CLI Exec Mode
13. Setting Switch Passwords
14. Interface Settings I
15. Interface Settings II
16. Interface Settings III
17. Switch Forwarding I
18. Switch Security I
19. Switch Interfaces and Forwarding Configuration Scenario
20. Configuring VLANs Configuration Scenario
21. VLAN Troubleshooting

If you are interested in exploring more hands-on labs and practice configuration and troubleshooting with more router and switch commands, go to **www.pearsonitcertification.com/networksimulator** for demos and to review the latest products for sale.

CCNA 200-301

Official Cert Guide, Volume 1

WENDELL ODOM, CCIE No. 1624 Emeritus

Cisco Press
221 River St. (3D11C)
Hoboken, NJ 07030

CCNA 200-301 Official Cert Guide, Volume 1

Wendell Odom

Published by:
Cisco Press

2 20

Library of Congress Control Number: 2019908180

ISBN-13: 978-0-13-579273-5
ISBN-10: 0-13-579273-8

Warning and Disclaimer

This book is designed to provide information about the Cisco CCNA 200-301 exam. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The authors, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc., cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Editor-in-Chief: Mark Taub

Business Operation Manager, Cisco Press: Ronald Fligge

Director ITP Product Management: Brett Bartow

Managing Editor: Sandra Schroeder

Development Editor: Christopher Cleveland

Senior Project Editor: Tonya Simpson

Copy Editor: Chuck Hutchinson

Technical Editor: Elan Beer

Editorial Assistant: Cindy Teeters

Cover Designer: Chuti Prasertsith

Composition: Tricia Bronkella

Indexer: Ken Johnson

Proofreader: Debbie Williams

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.

About the Author

Wendell Odom, CCIE No. 1624 Emeritus, has been in the networking industry since 1981. He has worked as a network engineer, consultant, systems engineer, instructor, and course developer; he currently works writing and creating certification study tools. This book is his 28th edition of some product for Pearson, and he is the author of all editions of the CCNA Cert Guides about Routing and Switching from Cisco Press. He has written books about topics from networking basics, certification guides throughout the years for CCENT, CCNA R&S, CCNA DC, CCNP ROUTE, CCNP QoS, and CCIE R&S. He maintains study tools, links to his blogs, and other resources at www.certskills.com.

About the Contributing Author

David Hucaby, CCIE No. 4594, CWNE No. 292, is a network engineer for University of Kentucky Healthcare. He has been authoring Cisco Press titles for 20 years, with a focus on wireless and LAN switching topics. David has bachelor of science and master of science degrees in electrical engineering. He lives in Kentucky with his wife, Marci, and two daughters.

About the Technical Reviewer

Elan Beer, CCIE No. 1837, is a senior consultant and Cisco instructor specializing in data center architecture and multiprotocol network design. For the past 27 years, Elan has designed networks and trained thousands of industry experts in data center architecture, routing, and switching. Elan has been instrumental in large-scale professional service efforts designing and troubleshooting internetworks, performing data center and network audits, and assisting clients with their short- and long-term design objectives. Elan has a global perspective of network architectures via his international clientele. Elan has used his expertise to design and troubleshoot data centers and internetworks in Malaysia, North America, Europe, Australia, Africa, China, and the Middle East. Most recently, Elan has been focused on data center design, configuration, and troubleshooting as well as service provider technologies. In 1993, Elan was among the first to obtain the Cisco Certified System Instructor (CCSI) certification, and in 1996, he was among the first to attain the Cisco System highest technical certification, the Cisco Certified Internetworking Expert. Since then, Elan has been involved in numerous large-scale data center and telecommunications networking projects worldwide.

Acknowledgments

Brett Bartow and I have been a team for a few decades. His support and wisdom have been a big help through what is the most significant change to the Cisco CCNA and CCNP certifications since their beginnings back in 1998. He's always a great partner on working through big picture direction as well as features to make the books the best they can be for our readers. Once again he's the starting point of the team! (And one of the things he does is gather the rest of the team that you see below…)

I don't mean this to sound too melodramatic, but I am too psyched: I got Dave Hucaby to join my team as a coauthor for this edition of the book! Dave's been writing about LAN switching, wireless LANs, and security topics for Cisco Press almost as long as I have, and I've always loved the accuracy and style of his books. Cisco added more than a little wireless LAN content to CCNA this time around. One thing led to another, I wondered if Dave might be willing to join in, and now we get Dave on the wireless chapters! I hope you'll enjoy those chapters as much as I did when preparing the book.

Chris Cleveland did the development editing for the very first Cisco Press exam certification guide way back in 1998, and he still can't seem to get away from us! Seriously, when Brett and I first discuss any new book, the first question is whether Chris has time to develop the book. It's always a pleasure working with you, Chris, for what seems like the 20th time or so by now.

The second question for Brett when starting a new book is whether we might be able to get Elan Beer to do the tech editing. Elan has the right wiring, skills, and experience to do a great job for us with all aspects of the tech editing process. Fantastic job as usual; thanks, Elan.

Sometimes, with a short book timeline as with this book, I don't know who's working on the project for the production group until I've written these notes, but I heard Sandra's and Tonya's names early this time. Knowing they would be on the project again really did give me a chance to exhale, and I have to say that knowing they would be on the project gave me a great sense of calm going into the production phase of the book.

Thanks to Sandra Schroeder, Tonya Simpson, and all the production team for making the magic happen. Not to sound too much like a broken record, but getting to work with familiar people who have been a great help in the past really does help reduce the stress when writing, besides getting the highest-quality product out the door in print and e-book forms. From fixing all my grammar and passive-voice sentences to pulling the design and layout together, they do it all; thanks for putting it all together and making it look easy. And Tonya got to juggle two books of mine at the same time (again)—thanks for managing the whole production process again.

Mike Tanamachi, illustrator and mind reader, did a great job on the figures again. I use a different process with the figures than most authors, with Mike drawing new figures as soon as I outline a new section or chapter. It means more edits when I change my mind and lots of mind reading of what Wendell really wanted versus what I drew poorly on my iPad. Mike came through again with some beautiful finished products.

I could not have made the timeline for this book without Chris Burns of Certskills Professional. Chris owns much of the PTP question support and administration process, works on the labs we put on my blog, and then catches anything I need to toss over my shoulder so I can focus on the books. Chris, you are the man!

A special thank you to you readers who write in with suggestions and possible errors, and especially those of you who post online at the Cisco Learning Network and at my blog (blog.certskills.com). Without question, the comments I receive directly and overhear by participating at CLN made this edition a better book.

Thanks to my wonderful wife, Kris, who helps make this sometimes challenging work lifestyle a breeze. I love walking this journey with you, doll. Thanks to my daughter Hannah, launching to college just as this book releases! And thanks to Jesus Christ, Lord of everything in my life.

Contents at a Glance

Online Appendixes

Contents

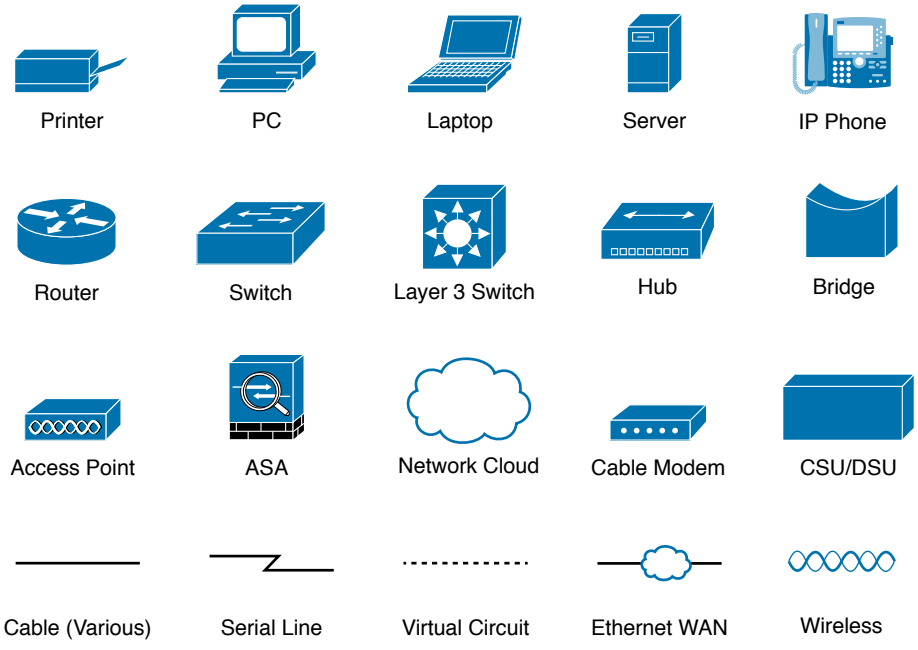
Online Appendixes

Reader Services

To access additional content for this book, simply register your product. To start the registration process, go to www.ciscopress.com/register and log in or create an account*. Enter the product ISBN 9780135792735 and click Submit. After the process is complete, you will find any available bonus content under Registered Products.

*Be sure to check the box that you would like to hear from us to receive exclusive discounts on future editions of this product.

Icons Used in This Book



Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.

- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([ ]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

Introduction

About Cisco Certifications and CCNA

Congratulations! If you're reading far enough to look at this book's Introduction, you've probably already decided to go for your Cisco certification, and the CCNA certification is the one place to begin that journey. If you want to succeed as a technical person in the networking industry at all, you need to know Cisco. Cisco has a ridiculously high market share in the router and switch marketplace, with more than 80 percent market share in some markets. In many geographies and markets around the world, networking equals Cisco. If you want to be taken seriously as a network engineer, Cisco certification makes perfect sense.

The first few pages of this Introduction explain the core features of Cisco's Career Certification program, of which the Cisco Certified Network Associate (CCNA) serves as the foundation for all the other certifications in the program. This section begins with a comparison of the old to the new certifications due to some huge program changes in 2019. It then gives the key features of CCNA, how to get it, and what's on the exam.

The Big Changes to Cisco Certifications in 2019

Cisco announced sweeping changes to its career certification program around mid-year 2019. Because so many of you will have read and heard about the old versions of the CCNA certification, this intro begins with a few comparisons between the old and new CCNA as well as some of the other Cisco career certifications.

First, consider Cisco's career certifications before 2019 as shown in Figure I-1. At that time, Cisco offered nine separate CCNA certifications in different technology tracks. Cisco also had seven Professional-level (CCNP, or Cisco Certified Network Professional) certifications.

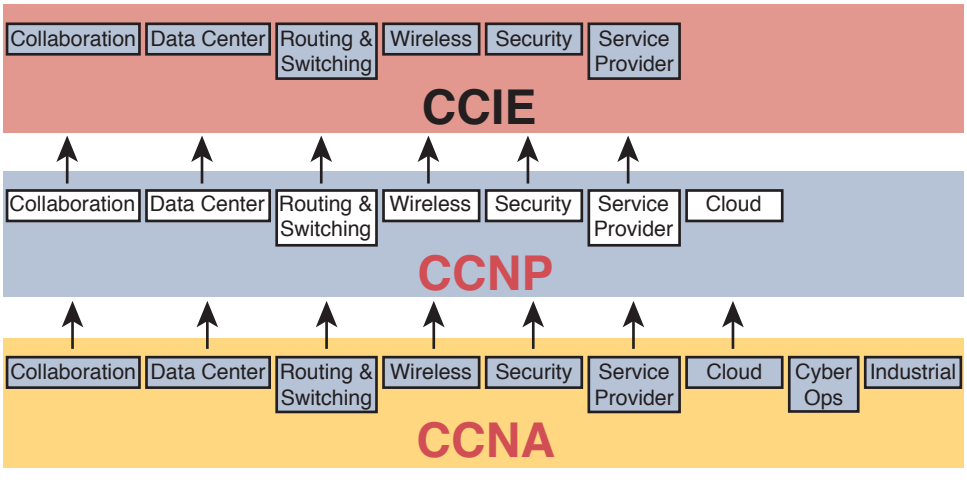


Figure I-1 *Old Cisco Certification Silo Concepts*

Why so many? Cisco began with one track—Routing and Switching—back in 1998. Over time, Cisco identified more and more technology areas that had grown to have enough content to justify another set of CCNA and CCNP certifications on those topics, so Cisco added more tracks. Many of those also grew to support expert level topics with CCIE (Cisco Certified Internetwork Expert).

In 2019, Cisco consolidated the tracks and moved the topics around quite a bit, as shown in Figure I-2.

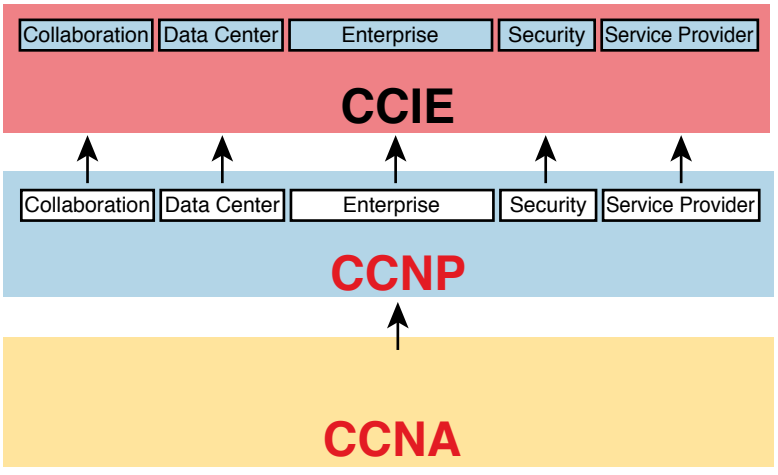


Figure I-2 *New Cisco Certification Tracks and Structure*

All the tracks now begin with the content in the one remaining CCNA certification. For CCNP, you now have a choice of five technology areas for your next steps, as shown in Figure I-2. (Note that Cisco replaced "Routing and Switching" with the term "Enterprise.")

Cisco made the following changes with the 2019 announcements:

CCENT: Retired the only Entry-level certification (CCENT, or Cisco Certified Entry Network Technician), with no replacement.

CCNA: Retired all the CCNA certifications except what was then known as "CCNA Routing and Switching," which became simply "CCNA."

CCNP: Consolidated the Professional level (CCNP) certifications to five tracks, including merging CCNP Routing and Switching and CCNP Wireless into CCNP Enterprise.

CCIE: Achieved better alignment with CCNP tracks through the consolidations.

Cisco needed to move many of the individual exam topics from one exam to another because of the number of changes. For instance, Cisco retired nine CCNA certifications plus the CCDA (Design Associate) certification—but those technologies didn't disappear! Cisco just moved the topics around to different exams in different certifications.

Consider wireless LANs as an example. The 2019 announcements retired both CCNA Wireless and CCNP Wireless as certifications. Some of the old CCNA Wireless topics landed in the new CCNA, while others landed in the two CCNP Enterprise exams about wireless LANs.

For those of you who want to learn more about the transition, check out my blog (blog.certskills.com) and look for posts in the News category from around June 2019. Now on to the details about CCNA as it exists starting in 2019!

How to Get Your CCNA Certification

As you saw in Figure I-2, all career certification paths now begin with CCNA. So how do you get it? Today, you have one and only one option to achieve CCNA certification:

Take and pass one exam: The Cisco 200-301 CCNA exam.

To take the 200-301 exam, or any Cisco exam, you will use the services of Pearson VUE (vue.com). The process works something like this:

1. Establish a login at https://home.pearsonvue.com/ (or use your existing login).
2. Register for, schedule a time and place, and pay for the Cisco 200-301 exam, all from the VUE website.
3. Take the exam at the VUE testing center.
4. You will receive a notice of your score, and whether you passed, before you leave the testing center.

Types of Questions on CCNA 200-301 Exam

The Cisco CCNA and CCNP exams all follow the same general format, with these types of questions:

- Multiple-choice, single-answer
- Multiple-choice, multiple-answer
- Testlet (one scenario with multiple multiple-choice questions)
- Drag-and-drop
- Simulated lab (sim)
- Simlet

Although the first four types of questions in the list should be somewhat familiar to you from other tests in school, the last two are more common to IT tests and Cisco exams in particular. Both use a network simulator to ask questions so that you control and use simulated Cisco devices. In particular:

Sim questions: You see a network topology and lab scenario, and can access the devices. Your job is to fix a problem with the configuration.

Simlet questions: This style combines sim and testlet question formats. As with a sim question, you see a network topology and lab scenario, and can access the devices. However, as with a testlet, you also see multiple multiple-choice questions. Instead of changing/fixing the configuration, you answer questions about the current state of the network.

These two question styles with the simulator give Cisco the ability to test your configuration skills with sim questions, and your verification and troubleshooting skills with simlet questions.

Before taking the test, learn the exam user interface by watching some videos Cisco provides about the exam user interface. To find the videos, just go to cisco.com and search for "Cisco Certification Exam Tutorial Videos."

CCNA 200-301 Exam Content, Per Cisco

Ever since I was in grade school, whenever the teacher announced that we were having a test soon, someone would always ask, "What's on the test?" We all want to know, and we all want to study what matters and avoid studying what doesn't matter.

Cisco tells the world the topics on each of its exams. Cisco wants the public to know the variety of topics and get an idea about the kinds of knowledge and skills required for each topic for every Cisco certification exam. To find the details, go to www.cisco.com/go/certifications, look for the CCNA page, and navigate until you see the exam topics.

This book also lists those same exam topics in several places. From one perspective, every chapter sets about to explain a small set of exam topics, so each chapter begins with the list of exam topics covered in that chapter. However, you might want to also see the exam topics in one place, so Appendix R, "Exam Topics Cross Reference," lists all the exam topics. You may want to download Appendix R in PDF form and keep it handy. The appendix lists the exam topics with two different cross references:

- A list of exam topics and the chapter(s) that covers each topic
- A list of chapters and the exam topics covered in each chapter

Exam Topic Verbs and Depth

Reading and understanding the exam topics, especially deciding the depth of skills required for each exam topic, require some thought. Each exam topic mentions the name of some technology, but it also lists a verb that implies the depth to which you must master the topic. The primary exam topics each list one or more verbs that describe the skill level required. For example, consider the following exam topic:

Configure and **verify** IPv4 addressing and subnetting

Note that this one exam topic has two verbs (*configure* and *verify*). Per this exam topic, you should be able to not only configure IPv4 addresses and subnets, but you should understand them well enough to verify that the configuration works. In contrast, the following exam topic asks you to describe a technology but does not ask you to configure it:

Describe the purpose of first hop redundancy protocol

The *describe* verb tells you to be ready to describe whatever a "first hop redundancy protocol" is. That exam topic also implies that you do not then need to be ready to configure or verify any first hop redundancy protocols (HSRP, VRRP, and GLBP).

Finally, note that the configure and verify exam topics imply that you should be able to describe and explain and otherwise master the concepts so that you understand what you have configured. The earlier "Configure and verify IPv4 addressing and subnetting"

does not mean that you should know how to type commands but have no clue as to what you configured. You must first master the conceptual exam topic verbs. The progression runs something like this:

Describe, Identify, Explain, Compare/Contrast, Configure, Verify, Troubleshoot

For instance, an exam topic that lists "compare and contrast" means that you should be able to describe, identify, and explain the technology. Also, an exam topic with "configure and verify" tells you to also be ready to describe, explain, and compare/contrast.

The Context Surrounding the Exam Topics

Take a moment to navigate to www.cisco.com/go/certifications and find the list of exam topics for the CCNA 200-301 exam. Did your eyes go straight to the list of exam topics? Or did you take the time to read the paragraphs above the exam topics first?

That list of exam topics for the CCNA 200-301 exam includes a little over 50 primary exam topics and about 50 more secondary exam topics. The primary topics have those verbs as just discussed, which tell you something about the depth of skill required. The secondary topics list only the names of more technologies to know.

However, the top of the web page that lists the exam topics also lists some important information that tells us some important facts about the exam topics. In particular, that leading text, found at the beginning of Cisco exam topic pages of most every exam, tells us

- The guidelines may change over time.
- The exam topics are general guidelines about what may be on the exam.
- The actual exam may include "other related topics."

Interpreting these three facts in order, I would not expect to see a change to the published list of exam topics for the exam. I've been writing the Cisco Press CCNA Cert Guides since Cisco announced CCNA back in 1998, and I've never seen Cisco change the official exam topics in the middle of an exam—not even to fix typos. But the introductory words say that they might change the exam topics, so it's worth checking.

As for the second item in the preceding list, even before you know what the acronyms mean, you can see that the exam topics give you a general but not detailed idea about each topic. The exam topics do not attempt to clarify every nook and cranny or to list every command and parameter; however, this book serves as a great tool in that it acts as a much more detailed interpretation of the exam topics. We examine every exam topic, and if we think a concept or command is possibly within an exam topic, we put it into the book. So, the exam topics give us general guidance, and these books give us much more detailed guidance.

The third item in the list uses literal wording that runs something like this: "However, other related topics may also appear on any specific delivery of the exam." That one statement can be a bit jarring to test takers, but what does it really mean? Unpacking the statement, it says that such questions may appear on any one exam but may not; in other words, they don't set about to ask every test taker some questions that include concepts

not mentioned in the exam topics. Second, the phrase "…other **related** topics…" emphasizes that any such questions would be related to some exam topic, rather than being far afield—a fact that helps us in how we respond to this particular program policy.

For instance, the CCNA 200-301 exam includes configuring and verifying the OSPF routing protocol, but it does not mention the EIGRP routing protocol. I personally would be unsurprised to see an OSPF question that required a term or fact not specifically mentioned in the exam topics. I would be surprised to see one that (in my opinion) ventures far away from the OSPF features in the exam topics. Also, I would not expect to see a question about how to configure and verify EIGRP.

And just as one final side point, note that Cisco does on occasion ask a test taker some unscored questions, and those may appear to be in this vein of questions from outside topics. When you sit down to take the exam, the small print mentions that you may see unscored questions and you won't know which ones are unscored. (These questions give Cisco a way to test possible new questions.) But some of these might be ones that fall into the "other related topics" category, but then not affect your score.

You should prepare a little differently for any Cisco exam, in comparison to say an exam back in school, in light of Cisco's "other related questions" policy:

- Do not approach an exam topic with an "I'll learn the core concepts and ignore the edges" approach.
- Instead, approach each exam topic with a "pick up all the points I can" approach by mastering each exam topic, both in breadth and in depth.
- Go beyond each exam topic when practicing configuration and verification by taking a little extra time to look for additional show commands and configuration options, and make sure you understand as much of the show command output that you can.

By mastering the known topics, and looking for places to go a little deeper, you will hopefully pick up the most points you can from questions about the exam topics. Then the extra practice you do with commands may happen to help you learn beyond the exam topics in a way that can help you pick up other points as well.

CCNA 200-301 Exam Content, Per This Book

When we created the Official Cert Guide content for the CCNA 200-301 exam, we considered a few options for how to package the content, and we landed on releasing a two-book set. Figure I-3 shows the setup of the content, with roughly 60 percent of the content in Volume 1 and the rest in Volume 2.

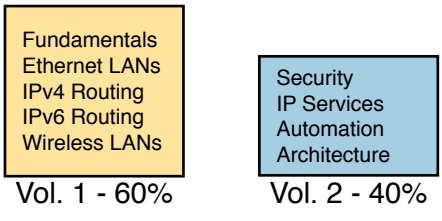


Figure I-3 *Two Books for CCNA 200-301*

The two books together cover all the exam topics in the CCNA 200-301 exam. Each chapter in each book develops the concepts and commands related to an exam topic, with clear and detailed explanations, frequent figures, and many examples that build your understanding of how Cisco networks work.

As for choosing what content to put into the books, note that we begin and finish with Cisco's exam topics, but with an eye toward predicting as many of the "other related topics" as we can. We start with the list of exam topics and apply a fair amount of experience, discussion, and other secret sauce to come up with an interpretation of what specific concepts and commands are worthy of being in the books or not. At the end of the writing process, the books should cover all the published exam topics, with additional depth and breadth that I choose based on the analysis of the exam. As we have done from the very first edition of the *CCNA Official Cert Guide*, we intend to cover each and every topic in depth. But as you would expect, we cannot predict every single fact on the exam given the nature of the exam policies, but we do our best to cover all known topics.

Book Features

This book includes many study features beyond the core explanations and examples in each chapter. This section acts as a reference to the various features in the book.

Chapter Features and How to Use Each Chapter

Each chapter of this book is a self-contained short course about one small topic area, organized for reading and study, as follows:

"Do I Know This Already?" quizzes: Each chapter begins with a pre-chapter quiz.

Foundation Topics: This is the heading for the core content section of the chapter.

Chapter Review: This section includes a list of study tasks useful to help you remember concepts, connect ideas, and practice skills-based content in the chapter.

Figure I-4 shows how each chapter uses these three key elements. You start with the DIKTA quiz. You can use the score to determine whether you already know a lot, or not so much, and determine how to approach reading the Foundation Topics (that is, the technology content in the chapter). When finished, use the Chapter Review tasks to start working on mastering your memory of the facts and skills with configuration, verification, and troubleshooting.

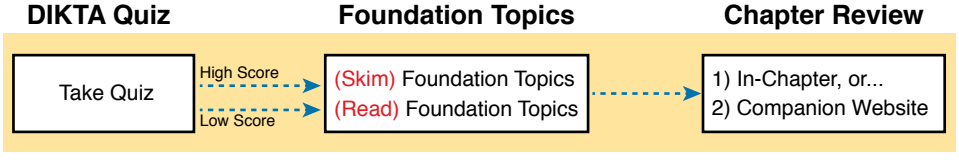


Figure I-4 *Three Primary Tasks for a First Pass Through Each Chapter*

In addition to these three main chapter features, each "Chapter Review" section uses a variety of other book features, including the following:

- **Review Key Topics:** Inside the "Foundation Topics" section, the Key Topic icon appears next to the most important items, for the purpose of later review and mastery. While all content matters, some is, of course, more important to learn, or needs more review to master, so these items are noted as key topics. The Chapter Review lists the key topics in a table; scan the chapter for these items to review them. Or review the key topics interactively using the companion website.
- **Complete Tables from Memory:** Instead of just rereading an important table of information, you will find some tables have been turned into memory tables, an interactive exercise found on the companion website. Memory tables repeat the table, but with parts of the table removed. You can then fill in the table to exercise your memory, and click to check your work.
- **Key Terms You Should Know:** You do not need to be able to write a formal definition of all terms from scratch; however, you do need to understand each term well enough to understand exam questions and answers. The Chapter Review lists the key terminology from the chapter. Make sure you have a good understanding of each term and use the Glossary to cross-check your own mental definitions. You can also review key terms with the "Key Terms Flashcards" app on the companion website.
- **Labs:** Many exam topics use verbs such as *configure* and *verify*; all these refer to skills you should practice at the user interface (CLI) of a router or switch. The Chapter and Part Reviews refer you to these other tools. The upcoming section titled "About Building Hands-On Skills" discusses your options.
- **Command References:** Some book chapters cover a large number of router and switch commands. The Chapter Review includes reference tables for the commands used in that chapter, along with an explanation. Use these tables for reference, but also use them for study. Just cover one column of the table, and see how much you can remember and complete mentally.
- **Review DIKTA Questions:** Although you have already seen the DIKTA questions from the chapters, re-answering those questions can prove a useful way to review facts. The Part Review suggests that you repeat the DIKTA questions but using the Pearson Test Prep (PTP) exam.
- **Subnetting Exercises:** Chapters 12, 13, 14, 22, and 24 ask you to perform some math processes related to either IPv4 or IPv6 addressing. The Chapter Review asks you to do additional practice problems. The problems can be found in Appendices D through H, in PDF form, on the companion website. The website also includes interactive versions of most of the exercises from those appendices.

Part Features and How to Use the Part Review

The book organizes the chapters into parts for the purpose of helping you study for the exam. Each part groups a small number of related chapters together. Then the study process (described just before Chapter 1) suggests that you pause after each part to do a

review of all chapters in the part. Figure I-5 lists the titles of the eight parts and the chapters in those parts (by chapter number) for this book.

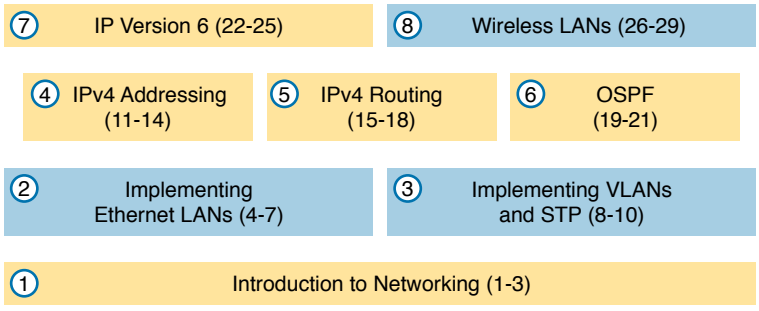


Figure I-5 *The Book Parts (by Title), and Chapter Numbers in Each Part*

The Part Review that ends each part acts as a tool to help you with spaced review sessions. Spaced reviews—that is, reviewing content several times over the course of your study—help improve retention. The Part Review activities include many of the same kinds of activities seen in the Chapter Review. Avoid skipping the Part Review, and take the time to do the review; it will help you in the long run.

The Companion Website for Online Content Review

We created an electronic version of every Chapter and Part Review task that could be improved though an interactive version of the tool. For instance, you can take a "Do I Know This Already?" quiz by reading the pages of the book, but you can also use our testing software. As another example, when you want to review the key topics from a chapter, you can find all those in electronic form as well.

All the electronic review elements, as well as other electronic components of the book, exist on this book's companion website. The companion website gives you a big advantage: you can do most of your Chapter and Part Review work from anywhere using the interactive tools on the site. The advantages include

- **Easier to use:** Instead of having to print out copies of the appendixes and do the work on paper, you can use these new apps, which provide you with an easy-to-use, interactive experience that you can easily run over and over.
- **Convenient:** When you have a spare 5–10 minutes, go to the book's website and review content from one of your recently finished chapters.
- **Untethered from the book:** You can access your review activities from anywhere—no need to have the book with you.
- **Good for tactile learners:** Sometimes looking at a static page after reading a chapter lets your mind wander. Tactile learners might do better by at least typing answers into an app, or clicking inside an app to navigate, to help keep you focused on the activity.

The interactive Chapter Review elements should improve your chances of passing as well. Our in-depth reader surveys over the years show that those who do the Chapter and Part Reviews learn more. Those who use the interactive versions of the review elements also tend to do more of the Chapter and Part Review work. So take advantage of the tools and maybe you will be more successful as well. Table I-1 summarizes these interactive applications and the traditional book features that cover the same content.

Table I-1 *Book Features with Both Traditional and App Options*

Feature	Traditional	App
Key Topic	Table with list; flip pages to find	Key Topics Table app
Config Checklist	Just one of many types of key topics	Config Checklist app
Key Terms	Listed in each "Chapter Review" section, with the Glossary in the back of the book	Glossary Flash Cards app
Subnetting Practice	Appendixes D–H, with practice problems and answers	A variety of apps, one per problem type

The companion website also includes links to download, navigate, or stream for these types of content:

- Pearson Sim Lite Desktop App
- Pearson Test Prep (PT) Desktop App
- Pearson Test Prep (PT) Web App
- Videos as mentioned in book chapters

How to Access the Companion Website

To access the companion website, which gives you access to the electronic content with this book, start by establishing a login at www.ciscopress.com and register your book. To do so, simply go to www.ciscopress.com/register and enter the ISBN of the print book: 9780135792735. After you have registered your book, go to your account page and click the **Registered Products** tab. From there, click the **Access Bonus Content** link to get access to the book's companion website.

Note that if you buy the *Premium Edition eBook and Practice Test* version of this book from Cisco Press, your book will automatically be registered on your account page. Simply go to your account page, click the **Registered Products** tab, and select **Access Bonus Content** to access the book's companion website.

How to Access the Pearson Test Prep (PTP) App

You have two options for installing and using the Pearson Test Prep application: a web app and a desktop app.

To use the Pearson Test Prep application, start by finding the registration code that comes with the book. You can find the code in these ways:

- **Print book:** Look in the cardboard sleeve in the back of the book for a piece of paper with your book's unique PTP code.
- **Premium Edition:** If you purchase the Premium Edition eBook and Practice Test directly from the Cisco Press website, the code will be populated on your account page after purchase. Just log in at www.ciscopress.com, click **account** to see details of your account, and click the **digital purchases** tab.
- **Amazon Kindle:** For those who purchase a Kindle edition from Amazon, the access code will be supplied directly from Amazon.
- **Other Bookseller E-books:** Note that if you purchase an e-book version from any other source, the practice test is not included because other vendors to date have not chosen to vend the required unique access code.

NOTE Do not lose the activation code because it is the only means with which you can access the QA content with the book.

Once you have the access code, to find instructions about both the PTP web app and the desktop app, follow these steps:

Step 1. Open this book's companion website, as was shown earlier in this Introduction under the heading "How to Access the Companion Website."

Step 2. Click the **Practice Exams** button.

Step 3. Follow the instructions listed there both for installing the desktop app and for using the web app.

Note that if you want to use the web app only at this point, just navigate to www.pearsontestprep.com, establish a free login if you do not already have one, and register this book's practice tests using the registration code you just found. The process should take only a couple of minutes.

NOTE Amazon eBook (Kindle) customers: It is easy to miss Amazon's email that lists your PTP access code. Soon after you purchase the Kindle eBook, Amazon should send an email. However, the email uses very generic text, and makes no specific mention of PTP or practice exams. To find your code, read every email from Amazon after you purchase the book. Also do the usual checks for ensuring your email arrives like checking your spam folder.

NOTE Other eBook customers: As of the time of publication, only the publisher and Amazon supply PTP access codes when you purchase their eBook editions of this book.

Feature Reference

The following list provides an easy reference to get the basic idea behind each book feature:

- **Practice exam:** The book gives you the rights to the Pearson Test Prep (PTP) testing software, available as a web app and desktop app. Use the access code on a piece of cardboard in the sleeve in the back of the book, and use the companion website to download the desktop app or navigate to the web app (or just go to www.pearsontestprep.com).
- **E-book:** Pearson offers an e-book version of this book that includes extra practice tests. If interested, look for the special offer on a coupon card inserted in the sleeve in the back of the book. This offer enables you to purchase the *CCNA 200-301 Official Cert Guide, Volume 1, Premium Edition eBook and Practice Test* at a 70 percent discount off the list price. The product includes three versions of the e-book, PDF (for reading on your computer), EPUB (for reading on your tablet, mobile device, or Nook or other e-reader), and Mobi (the native Kindle version). It also includes additional practice test questions and enhanced practice test features.
- **Subnetting videos:** The companion website contains a series of videos that show you how to calculate various facts about IP addressing and subnetting (in particular, using the shortcuts described in this book).
- **Mentoring videos:** The companion website also includes a number of videos about other topics as mentioned in individual chapters.
- **Subnetting practice apps:** The companion website contains appendixes with a set of subnetting practice problems and answers. This is a great resource to practice building subnetting skills. You can also do these same practice problems with applications from the "Chapter and Part Review" section of the companion website.
- **CCNA 200-301 Network Simulator Lite:** This lite version of the best-selling CCNA Network Simulator from Pearson provides you with a means, right now, to experience the Cisco command-line interface (CLI). No need to go buy real gear or buy a full simulator to start learning the CLI. Just install it from the companion website.
- **CCNA Simulator:** If you are looking for more hands-on practice, you might want to consider purchasing the CCNA Network Simulator. You can purchase a copy of this software from Pearson at http://pearsonitcertification.com/networksimulator or other retail outlets. To help you with your studies, Pearson has created a mapping guide that maps each of the labs in the simulator to the specific sections in each volume of the CCNA Cert Guide. You can get this mapping guide free on the Extras tab on the book product page: www.ciscopress.com/title/9780135792735.
- **PearsonITCertification.com:** The website www.pearsonitcertification.com is a great resource for all things IT-certification related. Check out the great CCNA articles, videos, blogs, and other certification preparation tools from the industry's best authors and trainers.

- **Author's website and blogs:** The author maintains a website that hosts tools and links useful when studying for CCNA. In particular, the site has a large number of free lab exercises about CCNA content, additional sample questions, and other exercises. Additionally, the site indexes all content so you can study based on the book chapters and parts. To find it, navigate to blog.certskills.com.

Book Organization, Chapters, and Appendixes

This book contains 29 core chapters, with each chapter covering a subset of the topics on the CCNA exam. The book organizes the chapters into parts of three to five chapters. The core chapters cover the following topics:

- **Part I: Introduction to Networking**
  - **Chapter 1, "Introduction to TCP/IP Networking,"** introduces the central ideas and terms used by TCP/IP, and contrasts the TCP/IP networking model with the OSI model.
  - **Chapter 2, "Fundamentals of Ethernet LANs,"** introduces the concepts and terms used when building Ethernet LANs.
  - **Chapter 3, "Fundamentals of WANs and IP Routing,"** covers the basics of the data-link layer for WANs in the context of IP routing but emphasizes the main network layer protocol for TCP/IP. This chapter introduces the basics of IPv4, including IPv4 addressing and routing.
- **Part II: Implementing Ethernet LANs**
  - **Chapter 4, "Using the Command-Line Interface,"** explains how to access the text-based user interface of Cisco Catalyst LAN switches.
  - **Chapter 5, "Analyzing Ethernet LAN Switching,"** shows how to use the Cisco CLI to verify the current status of an Ethernet LAN and how it switches Ethernet frames.
  - **Chapter 6, "Configuring Basic Switch Management,"** explains how to configure Cisco switches for basic management features, such as remote access using Telnet and SSH.
  - **Chapter 7, "Configuring and Verifying Switch Interfaces,"** shows how to configure a variety of switch features that apply to interfaces, including duplex/speed.
- **Part III: Implementing VLANs and STP**
  - **Chapter 8, "Implementing Ethernet Virtual LANs,"** explains the concepts and configuration surrounding virtual LANs, including VLAN trunking.
  - **Chapter 9, "Spanning Tree Protocol Concepts,"** discusses the concepts behind IEEE Spanning Tree Protocol (STP), including Rapid STP (RSTP) and how they make some switch interfaces block frames to prevent frames from looping continuously around a redundant switched LAN.
  - **Chapter 10, "RSTP and EtherChannel Configuration,"** shows how to configure and verify RSTP and Layer 2 EtherChannels on Cisco switches.

- **Part IV: IPv4 Addressing**
  - **Chapter 11, "Perspectives on IPv4 Subnetting,"** walks you through the entire concept of subnetting, from starting with a Class A, B, or C network to a completed subnetting design as implemented in an enterprise IPv4 network.
  - **Chapter 12, "Analyzing Classful IPv4 Networks,"** explains how IPv4 addresses originally fell into several classes, with unicast IP addresses being in Class A, B, and C. This chapter explores all things related to address classes and the IP network concept created by those classes.
  - **Chapter 13, "Analyzing Subnet Masks,"** shows how an engineer can analyze the key facts about a subnetting design based on the subnet mask. This chapter shows how to look at the mask and IP network to determine the size of each subnet and the number of subnets.
  - **Chapter 14, "Analyzing Existing Subnets,"** describes how most troubleshooting of IP connectivity problems starts with an IP address and mask. This chapter shows how to take those two facts and find key facts about the IP subnet in which that host resides.
- **Part V: IPv4 Routing**
  - **Chapter 15, "Operating Cisco Routers,"** is like Chapter 8, focusing on basic device management, but it focuses on routers instead of switches.
  - **Chapter 16, "Configuring IPv4 Addressing and Static Routes,"** discusses how to add IPv4 address configuration to router interfaces and how to configure static IPv4 routes.
  - **Chapter 17, "IP Routing in the LAN,"** shows how to configure and troubleshoot different methods of routing between VLANs, including Router-on-a-Stick (ROAS), Layer 3 switching with SVIs, Layer 3 switching with routed ports, and using Layer 3 EtherChannels.
  - **Chapter 18, "Troubleshooting IPv4 Routing,"** focuses on how to use two key troubleshooting tools to find routing problems: the **ping** and **traceroute** commands.
- **Part VI: OSPF**
  - **Chapter 19, "Understanding OSPF Concepts,"** introduces the fundamental operation of the Open Shortest Path First (OSPF) protocol, focusing on link state fundamentals, neighbor relationships, flooding link state data, and calculating routes based on the lowest cost metric.
  - **Chapter 20, "Implementing OSPF,"** takes the concepts discussed in the previous chapter and shows how to configure and verify those same features.
  - **Chapter 21, "OSPF Network Types and Neighbors,"** takes the next steps in OSPF configuration and verification by looking in more depth at the concepts of how routers enable OSPF on interfaces, and the conditions that must be true before two routers will succeed in becoming OSPF neighbors.
- **Part VII: IP Version 6**
  - **Chapter 22, "Fundamentals of IP Version 6,"** discusses the most basic concepts of IP version 6, focusing on the rules for writing and interpreting IPv6 addresses.

- **Chapter 23, "IPv6 Addressing and Subnetting,"** works through the two branches of unicast IPv6 addresses—global unicast addresses and unique local addresses—that act somewhat like IPv4 public and private addresses, respectively.
- **Chapter 24, "Implementing IPv6 Addressing on Routers,"** shows how to configure IPv6 routing and addresses on routers, while discussing a variety of special IPv6 addresses.
- **Chapter 25, "Implementing IPv6 Routing,"** shows how to add static routes to an IPv6 router's routing table.

- **Part VIII: Wireless LANs**
  - **Chapter 26, "Fundamentals of Wireless Networks,"** introduces the foundational concepts of wireless 802.11 LANs, including wireless topologies and basic wireless radio communications protocols.
  - **Chapter 27, "Analyzing Cisco Wireless Architectures,"** turns your attention to the questions related to systematic and architectural issues surrounding how to build wireless LANs and explains the primary options available for use.
  - **Chapter 28, "Securing Wireless Networks,"** explains the unique security challenges that exist in a wireless LAN and the protocols and standards used to prevent different kinds of attacks.
  - **Chapter 29, "Building a Wireless LAN,"** shows how to configure and secure a wireless LAN using a Wireless LAN Controller (WLC).
- **Part IX: Print Appendixes**
  - **Appendix A, "Numeric Reference Tables,"** lists several tables of numeric information, including a binary-to-decimal conversion table and a list of powers of 2.
  - **Appendix B, "CCNA 200-301, Volume 1 Exam Updates,"** is a place for the author to add book content mid-edition. Always check online for the latest PDF version of this appendix; the appendix lists download instructions.
  - **Appendix C, "Answers to the 'Do I Know This Already?' Quizzes,"** includes the explanations to all the "Do I Know This Already" quizzes.
  - The **Glossary** contains definitions for all the terms listed in the "Key Terms You Should Know" sections at the conclusion of the chapters.
- **Part X: Online Appendixes**
- **Practice Appendixes**

The following appendixes are available in digital format from the companion website. These appendixes provide additional practice for several networking processes that use some math.

- **Appendix D, "Practice for Chapter 12: Analyzing Classful IPv4 Networks"**
- **Appendix E, "Practice for Chapter 13: Analyzing Subnet Masks"**
- **Appendix F, "Practice for Chapter 14: Analyzing Existing Subnets"**
- **Appendix G, "Practice for Chapter 22: Fundamentals of IP Version 6"**

- **Appendix H, "Practice for Chapter 24: Implementing IPv6 Addressing on Routers"**
- **Content from Previous Editions**

Although the publisher restarts numbering at edition "1" each time, the name of the related exam changes in a significant way. In function, this book is in effect part of the 9th edition of the CCNA Cert Guide materials from Cisco Press. From edition to edition, some readers over the years have asked that we keep some select chapters with the book. Keeping content that Cisco removed from the exam, but that may still be useful, can help the average reader as well as instructors who use the materials to teach courses with this book. The following appendices hold this edition's content from previous editions:

- **Appendix J, "Topics from Previous Editions,"** is a collection of small topics from prior editions. None of the topics justify a complete appendix by themselves, so we collect the small topics into this single appendix.
- **Appendix K, "Analyzing Ethernet LAN Designs,"** examines various ways to design Ethernet LANs, discussing the pros and cons, and explains common design terminology.
- **Appendix L, "Subnet Design,"** takes a design approach to subnetting. This appendix begins with a classful IPv4 network and asks why a particular mask might be chosen, and if chosen, what subnet IDs exist.
- **Appendix M, "Practice for Appendix L: Subnet Design"**
- **Appendix N, "Variable-Length Subnet Masks,"** moves away from the assumption of one subnet mask per network to multiple subnet masks per network, which makes subnetting math and processes much more challenging. This appendix explains those challenges.
- **Appendix O, "Spanning Tree Protocol Implementation,"** shows how to configure and verify STP on Cisco switches.
- **Appendix P, "LAN Troubleshooting,"** examines the most common LAN switching issues and how to discover those issues when troubleshooting a network. The appendix includes troubleshooting topics for STP/RSTP, Layer 2 EtherChannel, LAN switching, VLANs, and VLAN trunking.
- **Appendix Q, "Troubleshooting IPv4 Routing Protocols,"** walks through the most common problems with IPv4 routing protocols, while alternating between OSPF examples and EIGRP examples.
- **Miscellaneous Appendixes**
  - **Appendix I, "Study Planner,"** is a spreadsheet with major study milestones, where you can track your progress through your study.
  - **Appendix R, "Exam Topics Cross Reference,"** provides some tables to help you find where each exam objective is covered in the book.

About Building Hands-On Skills

You need skills in using Cisco routers and switches, specifically the Cisco command-line interface (CLI). The Cisco CLI is a text-based command-and-response user interface; you type a command, and the device (a router or switch) displays messages in response. To answer sim and simlet questions on the exams, you need to know a lot of commands, and you need to be able to navigate to the right place in the CLI to use those commands.

This next section walks through the options of what is included in the book, with a brief description of lab options outside the book.

Config Lab Exercises

Some router and switch features require multiple configuration commands. Part of the skill you need to learn is to remember which configuration commands work together, which ones are required, and which ones are optional. So, the challenge level goes beyond just picking the right parameters on one command. You have to choose which commands to use, in which combination, typically on multiple devices. And getting good at that kind of task requires practice.

Each Config Lab lists details about a straightforward lab exercise for which you should create a small set of configuration commands for a few devices. Each lab presents a sample lab topology, with some requirements, and you have to decide what to configure on each device. The answer then shows a sample configuration. Your job is to create the configuration and then check your answer versus the supplied answer.

Config Lab content resides outside the book at the author's blog site (blog.certskills.com). You can navigate to the Config Lab in a couple of ways from the site, or just go directly to https://blog.certskills.com/category/hands-on/config-lab/ to reach a list of all Config Labs. Figure I-6 shows the logo that you will see with each Config Lab.

Figure I-6 *Config Lab Logo in the Author's Blogs*

These Config Labs have several benefits, including the following:

Untethered and responsive: Do them from anywhere, from any web browser, from your phone or tablet, untethered from the book or DVD.

Designed for idle moments: Each lab is designed as a 5- to 10-minute exercise if all you are doing is typing in a text editor or writing your answer on paper.

Two outcomes, both good: Practice getting better and faster with basic configuration, or if you get lost, you have discovered a topic that you can now go back and reread to complete your knowledge. Either way, you are a step closer to being ready for the exam!

Blog format: The format allows easy adds and changes by me and easy comments by you.

Self-assessment: As part of final review, you should be able to do all the Config Labs, without help, and with confidence.

Note that the blog organizes these Config Lab posts by book chapter, so you can easily use these at both Chapter Review and Part Review. See the "Your Study Plan" element that follows the Introduction for more details about those review sections.

A Quick Start with Pearson Network Simulator Lite

The decision of how to get hands-on skills can be a little scary at first. The good news: You have a free and simple first step to experience the CLI: install and use the Pearson Network Simulator Lite (or NetSim Lite) that comes with this book.

This book comes with a lite version of the best-selling CCNA Network Simulator from Pearson, which provides you with a means, right now, to experience the Cisco CLI. No need to go buy real gear or buy a full simulator to start learning the CLI. Just install it from the companion website.

This latest version of NetSim Lite includes labs associated with Part II of this book, plus a few more from Part III. Part I includes concepts only, with Part II being the first part with commands. So, make sure to use the NetSim Lite to learn the basics of the CLI to get a good start.

Of course, one reason that you get access to the NetSim Lite is that the publisher hopes you will buy the full product. However, even if you do not use the full product, you can still learn from the labs that come with NetSim Lite while deciding about what options to pursue.

The Pearson Network Simulator

The Config Labs and the Pearson Network Simulator Lite both fill specific needs, and they both come with the book. However, you need more than those two tools.

The single best option for lab work to do along with this book is the paid version of the Pearson Network Simulator. This simulator product simulates Cisco routers and switches so that you can learn for CCNA certification. But more importantly, it focuses on learning for the exam by providing a large number of useful lab exercises. Reader surveys tell us that those people who use the Simulator along with the book love the learning process and rave about how the book and Simulator work well together.

Of course, you need to make a decision for yourself and consider all the options. Thankfully, you can get a great idea of how the full Simulator product works by using the Pearson Network Simulator Lite product included with the book. Both have the same base code, same user interface, and same types of labs. Try the Lite version to decide if you want to buy the full product.

Note that the Simulator and the books work on a different release schedule. For a time in 2019 (and probably into 2020), the Simulator will be the one created for the previous versions of the exams (ICND1 100-105, ICND2 200-105, and CCNA 200-125).

Interestingly, Cisco did not add a large number of new topics that require CLI skills to the CCNA 200-301 exam as compared with its predecessor, so the old Simulator covers most of the CLI topics. So, during the interim before the products based on the 200-301 exam come out, the old Simulator products should be quite useful.

On a practical note, when you want to do labs when reading a chapter or doing Part Review, the Simulator organizes the labs to match the book. Just look for the Sort by Chapter tab in the Simulator's user interface. However, during the months in 2019 for which the Simulator is the older edition listing the older exams in the title, you will need to refer to a PDF that lists those labs versus this book's organization. You can find that PDF on the book product page under the Downloads tab here: www.ciscopress.com/title/9780135792735.

More Lab Options

If you decide against using the full Pearson Network Simulator, you still need hands-on experience. You should plan to use some lab environment to practice as much CLI as possible.

First, you can use real Cisco routers and switches. You can buy them, new or used, or borrow them at work. You can rent them for a fee. If you have the right mix of gear, you could even do the Config Lab exercises from my blog on that gear or try to re-create examples from the book.

Cisco also makes a simulator that works very well as a learning tool: Cisco Packet Tracer. Cisco now makes Packet Tracer available for free. However, unlike the Pearson Network Simulator, it does not include lab exercises that direct you as to how to go about learning each topic. If interested in more information about Packet Tracer, check out my series about using Packet Tracer at my blog (blog.certskills.com); just search for "Packet Tracer."

Cisco offers a virtualization product that lets you run router and switch operating system (OS) images in a virtual environment. This tool, the Virtual Internet Routing Lab (VIRL), lets you create a lab topology, start the topology, and connect to real router and switch OS images. Check out http://virl.cisco.com for more information.

You can even rent virtual Cisco router and switch lab pods from Cisco, in an offering called Cisco Learning Labs (https://learningnetworkstore.cisco.com/cisco-learning-labs).

This book does not tell you what option to use, but you should plan on getting some hands-on practice somehow. The important thing to know is that most people need to practice using the Cisco CLI to be ready to pass these exams.

For More Information

If you have any comments about the book, submit them via www.ciscopress.com. Just go to the website, select **Contact Us**, and type your message.

Cisco might make changes that affect the CCNA certification from time to time. You should always check www.cisco.com/go/ccna for the latest details.

The *CCNA 200-301 Official Cert Guide, Volume 1*, helps you attain CCNA certification. This is the CCNA certification book from the only Cisco-authorized publisher. We at Cisco Press believe that this book certainly can help you achieve CCNA certification, but the real work is up to you! I trust that your time will be well spent.

This page intentionally left blank

Your Study Plan

You just got this book. You have probably already read (or quickly skimmed) the Introduction. You are probably now wondering whether to start reading here or skip ahead to Chapter 1, "Introduction to TCP/IP Networking."

Stop to read this section about how to create your own study plan for the CCNA 200-301 exam. Your study will go much better if you take time (maybe 15 minutes) to think about a few key points about how to study before starting on this journey. That is what this section will help you do.

A Brief Perspective on Cisco Certification Exams

Cisco sets the bar pretty high for passing the CCNA 200-301 exam. Most anyone can study and pass the exam, but it takes more than just a quick read through the book and the cash to pay for the exam.

The challenge of the exam comes from many angles. First, the exam covers a lot of concepts and many commands specific to Cisco devices. Beyond knowledge, all these Cisco exams also require deep skills. You must be able to analyze and predict what really happens in a network, and you must be able to configure Cisco devices to work correctly in those networks.

The more challenging questions on these exams work a lot like a jigsaw puzzle, but with four out of every five puzzle pieces not even in the room. To solve the puzzle, you have to mentally re-create the missing pieces. To do that, you must know each networking concept and remember how the concepts work together.

For instance, you might encounter a question that asks you why two routers cannot exchange routing information using the OSPF routing protocol. The question would supply some of the information, like some pieces of the jigsaw puzzle, as represented with the white pieces in Figure 1. You have to apply your knowledge of IPv4 routing, IPv4 addressing, and the OSPF protocol to the scenario in the question to come up with some of the other pieces of the puzzle. For a given question, some pieces of the puzzle might remain a mystery, but with enough of the puzzle filled in, you should be able to answer the question. And some pieces will just remain unknown for a given question.

These skills require that you prepare by doing more than just reading and memorizing. Of course, you need to read many pages in this book to learn many individual facts and how these facts relate to each other. But a big part of this book lists exercises that require more than just simply reading, exercises that help you build the skills to solve these networking puzzles.

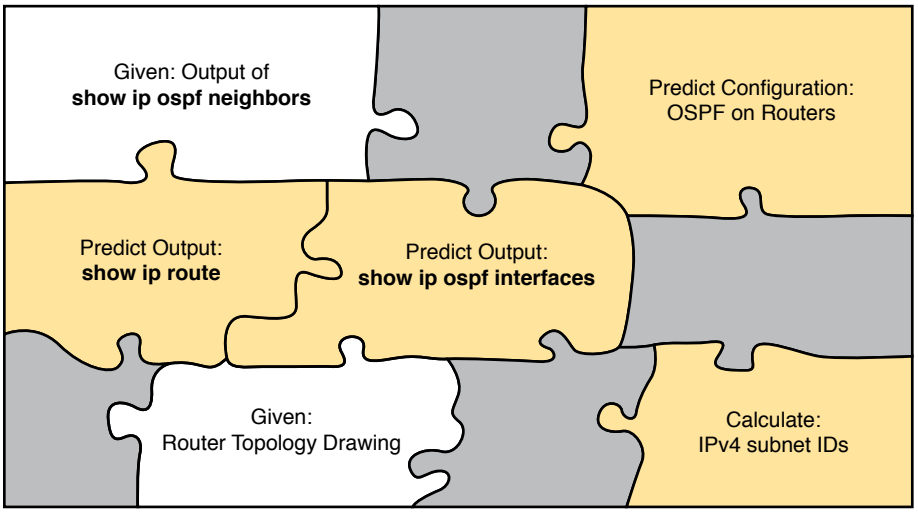


Figure 1 *Filling In Puzzle Pieces with Your Analysis Skills*

Five Study Plan Steps

What do you need to do to be ready to pass, beyond reading and remembering all the facts? You need to develop skills. You need to mentally link each idea with other related ideas. Doing that requires additional work. To help you along the way, the next few pages give you five key planning steps to take so that you can more effectively build those skills and make those connections, before you dive into this exciting but challenging world of learning networking on Cisco gear.

Step 1: Think in Terms of Parts and Chapters

The first step in your study plan is to get the right mindset about the size and nature of the task you have set out to accomplish. This is a large book, and to be ready for the CCNA 200-301 exam, you need to complete it and then the *CCNA 200-301 Official Cert Guide, Volume 2*. You cannot think about these two books as one huge task, or you might get discouraged. So break the task down into smaller tasks.

The good news here is that the book is designed with obvious breakpoints and built-in extensive review activities. In short, the book is more of a study system than a book.

The first step in your study plan is to visualize this book not as one large book but as components. First, visualize the book as eight smaller parts. Then, within each part, visualize each part as three or four chapters. Your study plan has you working through the chapters in each part and then reviewing the material in that part before moving on, as shown in Figure 2.

Now your plan has the following:

1 large task: Read and master all content in the book.

8 medium tasks/book: Read and master a part.

4 small tasks/part: Read and master a chapter.

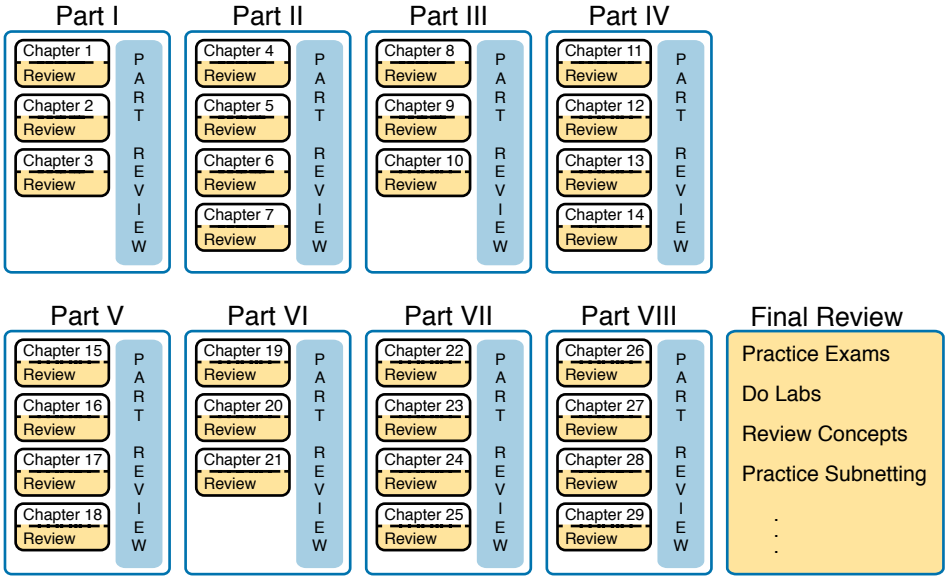


Figure 2 *Eight Parts, with an Average of Four Chapters Each, with Part Reviews*

Step 2: Build Your Study Habits Around the Chapter

For your second step, possibly the most important step, approach each chapter with the same process as shown in Figure 3. The chapter pre-quiz (called a DIKTA quiz, or "Do I Know This Already?" quiz) helps you decide how much time to spend reading versus skimming the core of the chapter, called the "Foundation Topics." The "Chapter Review" section then gives you instructions about how to study and review what you just read.

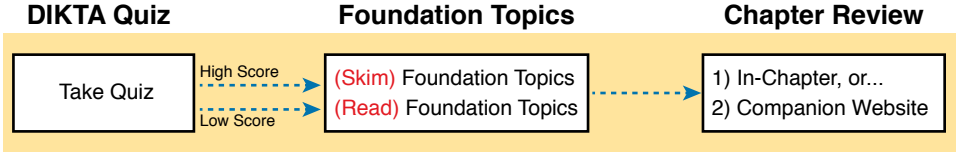


Figure 3 *Suggested Approach to Each Chapter*

The book has no long chapters, on purpose. They average about 20 pages for the Foundation Topics (which is the part of the chapter with new content). Because we kept the size reasonable, you can complete all of a chapter in one or two short study sessions. For instance, when you begin a new chapter, if you have an hour or an hour and a half, you should be able to complete a first reading of the chapter and at least make a great start on it. And even if you do not have enough time to read the entire chapter, look for the major headings inside the chapter; each chapter has two to three major headings, and those make a great place to stop reading when you need to wait to complete the reading in the next study sessions.

The Chapter Review tasks are very important to your exam-day success. Doing these tasks after you've read the chapter really does help you get ready. Do not put off using these tasks until later! The chapter-ending review tasks help you with the first phase of deepening

your knowledge and skills of the key topics, remembering terms, and linking the concepts together in your brain so that you can remember how it all fits together. The following list describes most of the activities you will find in the "Chapter Review" sections:

- Review key topics
- Review key terms
- Answer the DIKTA questions
- Re-create config checklists
- Review command tables
- Review memory tables
- Do lab exercises
- Watch video
- Do subnetting exercises

Step 3: Use Book Parts for Major Milestones

Studies show that to master a concept and/or skill, you should plan to go through multiple study sessions to review the concept and to practice the skill. The "Chapter Review" section at the end of each chapter is the first such review, while the Part Review, at the end of each part, acts as that second review.

Plan time to do the Part Review task at the end of each part, using the Part Review elements found at the end of each part. You should expect to spend about as much time on one Part Review as you would on one entire chapter. So in terms of planning your time, think of the Part Review itself as another chapter.

Figure 4 lists the names of the parts in this book, with some color coding. Note that Parts II and III are related (Ethernet), and Parts IV through VII are also related (IP version 4 and IP Version 6). Each part ends with a Part Review section of two to four pages, with notes about what tools and activities to use.

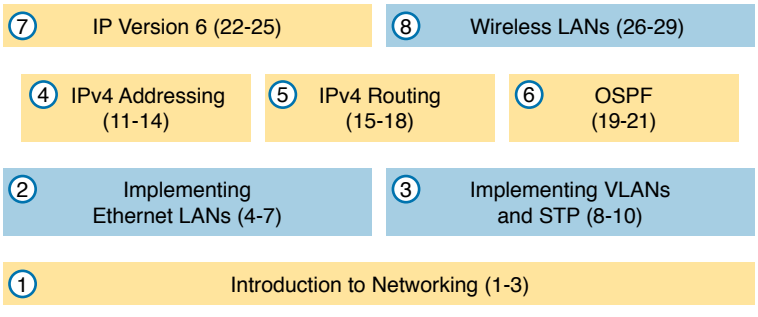


Figure 4 *Parts as Major Milestones*

Also, consider setting a goal date for finishing each part of the book (and a reward, as well). Plan a break, some family time, some time out exercising, eating some good food, whatever helps you get refreshed and motivated for the next part.

Step 4: Use Volume 2's Final Review Chapter

Your fourth step has one overall task: perform the details outlined in the "Final Exam Review" chapter at the end of the *CCNA 200-301 Official Cert Guide, Volume 2*. Note that you have no exam to take at the end of this Volume 1 book, so keep working with Volume 2 when you complete this book. Once you're finished with both books, Volume 2's "Final Exam Review" will direct you.

Step 5: Set Goals and Track Your Progress

Your fifth study plan step spans the entire timeline of your study effort. Before you start reading the book and doing the rest of these study tasks, take the time to make a plan, set some goals, and be ready to track your progress.

While making lists of tasks may or may not appeal to you, depending on your personality, goal setting can help everyone studying for these exams. And to do the goal setting, you need to know what tasks you plan to do.

NOTE If you read this, and decide that you want to try to do better with goal setting beyond your exam study, check out a blog series I wrote about planning your networking career here: http://blog.certskills.com/tag/development-plan/.

As for the list of tasks to do when studying, you do not have to use a detailed task list. (You could list every single task in every chapter-ending "Chapter Review" section, every task in the Part Reviews, and every task in the "Final Review" chapter.) However, listing the major tasks can be enough.

You should track at least two tasks for each typical chapter: reading the "Foundation Topics" section and doing the Chapter Review at the end of the chapter. And, of course, do not forget to list tasks for Part Reviews and Final Review. Table 1 shows a sample for Part I of this book.

Table 1 Sample Excerpt from a Planning Table

Element	Task	Goal Date	First Date Completed	Second Date Completed (Optional)
Chapter 1	Read Foundation Topics			
Chapter 1	Do Chapter Review tasks			
Chapter 2	Read Foundation Topics			
Chapter 2	Do Chapter Review tasks			
Chapter 3	Read Foundation Topics			
Chapter 3	Do Chapter Review tasks			
Part I Review	Do Part Review activities			

NOTE Appendix I, "Study Planner," on the companion website, contains a complete planning checklist like Table 1 for the tasks in this book. This spreadsheet allows you to update and save the file to note your goal dates and the tasks you have completed.

Use your goal dates as a way to manage your study, and not as a way to get discouraged if you miss a date. Pick reasonable dates that you can meet. When setting your goals, think about how fast you read and the length of each chapter's "Foundation Topics" section, as listed in the table of contents. Then, when you finish a task sooner than planned, move up the next few goal dates.

If you miss a few dates, do *not* start skipping the tasks listed at the ends of the chapters! Instead, think about what is impacting your schedule—real life, commitment, and so on—and either adjust your goals or work a little harder on your study.

Things to Do Before Starting the First Chapter

Now that you understand the big ideas behind a good study plan for the book, take a few more minutes for a few overhead actions that will help. Before leaving this section, look at some other tasks you should do either now or around the time you are reading the first few chapters to help make a good start in the book.

Bookmark the Companion Website

The companion website contains links to all the tools you need for chapter and part review. In fact, it includes a chapter-by-chapter and part-by-part breakdown of all the review activities. Before you finish the first chapter, make sure and follow the instructions in the Introduction's section titled "The Companion Website for Online Content Review," get access, and bookmark the page.

Also, if you did not yet read about the companion website in the Introduction or explore the site, take a few minutes to look at the resources available on the site.

Bookmark/Install Pearson Test Prep

This book, like many other Cisco Press books, includes the rights to use the Pearson Test Prep (PTP) software, along with rights to use some exam questions related to this book. PTP has many useful study features:

- Both a web and desktop version for your convenience and choice
- History tracking of your simulated exam attempts, synchronized between web and desktop
- Study mode, which lets you see the correct answers with each question and the related explanations
- Practice exam mode, which simulates exam conditions, hiding answers/explanations and timing the exam event
- Filters to let you choose questions based on chapter(s) and/or part(s)

You should take a few minutes to set up your PTP installation. Refer to the section titled "How to Access the Pearson Test Prep (PTP) App" in the Introduction for details.

Understand This Book's PTP Databases and Modes

When you activate a product in PTP, you gain the rights to that product's exams. Understanding those exams helps you choose when to use them and when to delay using different exams to save those questions for later. The retail version of this book comes with four exams, as shown in Figure 5; the premium edition adds exams 3 and 4, which are similar in purpose to exams 1 and 2.

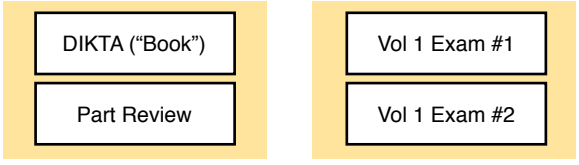


Figure 5 *PTP Exams/Exam Databases and When to Use Them*

When using PTP, you can choose to use any of these exam databases at any time, both in study mode and practice exam mode. However, many people find it best to avoid using some exams until you do your final exam review at the end of reading the *CCNA 200-301 Official Cert Guide, Volume 2*. So, consider using this plan:

- During Chapter Review, use PTP to review the DIKTA questions for that chapter, using study mode.
- During Part Review, use the questions built specifically for Part Review (the Part Review questions) for that part of the book, using study mode.
- Save the remaining exams to use with the "Final Review" chapter at the end of the Volume 2 book.

Alternatively, use exams 1 and 2 at any time during your study, and consider buying the premium edition of the book to add two more exams. For instance, you could review each chapter by answering the questions from that chapter in exams 1 and 2, and wait to use exams 3 and 4 until your final exam review at the end of Volume 2.

NOTE The *CCNA 200-301 Official Cert Guide, Volume 2*, includes several CCNA exams as well—exams that include questions from Volume 1 and Volume 2. You can use those exams during final review to practice simulated CCNA 200-301 exams.

Additionally, take the time to experiment with the study modes in the PTP applications:

Study mode: Study mode works best when you are still working on understanding and learning the content. In study mode, you can see the answers immediately, so you can study the topics more easily.

Practice mode: This mode lets you practice an exam event somewhat like the actual exam. It gives you a preset number of questions, from all chapters, with a timed event. Practice exam mode also gives you a score for that timed event.

Practice Viewing Per-Chapter DIKTA Questions

Take a few minutes to experiment with and understand how to use PTP to answer questions from a single chapter's DIKTA quiz, as follows:

Step 1. Start the PTP web or desktop app.

Step 2. From the main (home) menu, select the item for this product, with a name like *CCNA 200-301 Official Cert Guide, Volume 1*, and click **Open Exam**.

Step 3. The top of the next window that appears should list some exams. Check the **Book Questions** box, and uncheck the other boxes. This selects the "book" questions (that is, the DIKTA questions from the beginning of each chapter).

Step 4. On this same window, click at the bottom of the screen to deselect all objectives (chapters). Then select the box beside each chapter in the part of the book you are reviewing.

Step 5. Select any other options on the right side of the window.

Step 6. Click **Start** to start reviewing the questions.

Practice Viewing Per-Part Review Questions

Your PTP access also includes a Part Review exam created solely for study during the Part Review process. To view these questions, follow the same process as you did with DIKTA/book questions, but select the Part Review database rather than the book database. PTP has a clear name for this database: Part Review Questions.

Join the Cisco Learning Network CCNA Study Group

Register (for free) at the Cisco Learning Network (CLN, http://learningnetwork.cisco.com) and join the CCNA study group. This group allows you to both lurk and participate in discussions about topics related to the CCNA exam. Register (for free), join the groups, and set up an email filter to redirect the messages to a separate folder. Even if you do not spend time reading all the posts yet, later, when you have time to read, you can browse through the posts to find interesting topics (or just search the posts from the CLN website).

Getting Started: Now

Now dive in to your first of many short, manageable tasks: reading the relatively short Chapter 1. Enjoy!

This first part of the book introduces the fundamentals of the most important topics in TCP/IP networking. Chapter 1 provides a broad look at TCP/IP, introducing the common terms, big concepts, and major protocols for TCP/IP. Chapter 2 then examines local-area networks (LAN), which are networks that connect devices that are located near each other; for instance, in the same building. Chapter 3 then shows how to connect those LANs across long distances with wide-area networks (WAN) with a focus on how routers connect LANs and WANs to forward data between any two devices in the network.

Part I

Introduction to Networking

Chapter 1: Introduction to TCP/IP Networking

Chapter 2: Fundamentals of Ethernet LANs

Chapter 3: Fundamentals of WANs and IP Routing

Part I Review

CHAPTER 1

Introduction to TCP/IP Networking

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.3 Compare physical interface and cabling types

1.3.a Single-mode fiber, multimode fiber, copper

1.3.b Connections (Ethernet shared media and point-to-point)

Welcome to the first chapter in your study for CCNA! This chapter begins Part I, which focuses on the basics of networking.

Networks work correctly because the various devices and software follow the rules. Those rules come in the form of standards and protocols, which are agreements of a particular part of how a network should work. However, the sheer number of standards and protocols available can make it difficult for the average network engineer to think about and work with networks—so the world of networking has used several networking models over time. Networking models define a structure and different categories (layers) of standards and protocols. As new standards and protocols emerge over time, networkers can think of those new details in the context of a working model.

You can think of a networking model as you think of a set of architectural plans for building a house. A lot of different people work on building your house, such as framers, electricians, bricklayers, painters, and so on. The blueprint helps ensure that all the different pieces of the house work together as a whole. Similarly, the people who make networking products, and the people who use those products to build their own computer networks, follow a particular networking model. That networking model defines rules about how each part of the network should work, as well as how the parts should work together so that the entire network functions correctly.

Today, TCP/IP rules as the most pervasive networking model in use. You can find support for TCP/IP on practically every computer operating system (OS) in existence today, from mobile phones to mainframe computers. Every network built using Cisco products today supports TCP/IP. And not surprisingly, the CCNA exam focuses heavily on TCP/IP. This chapter uses TCP/IP for one of its main purposes: to present various concepts about networking using the context of the different roles and functions in the TCP/IP model.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 1-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Perspectives on Networking	None
TCP/IP Networking Model	1–4
Data Encapsulation Terminology	5–7

1. Which of the following protocols are examples of TCP/IP transport layer protocols? (Choose two answers.)
   - **a.** Ethernet
   - **b.** HTTP
   - **c.** IP
   - **d.** UDP
   - **e.** SMTP
   - **f.** TCP
2. Which of the following protocols are examples of TCP/IP data-link layer protocols? (Choose two answers.)
   - **a.** Ethernet
   - **b.** HTTP
   - **c.** IP
   - **d.** UDP
   - **e.** SMTP
   - **f.** TCP
   - **g.** PPP
3. The process of HTTP asking TCP to send some data and making sure that it is received correctly is an example of what?
   - **a.** Same-layer interaction
   - **b.** Adjacent-layer interaction
   - **c.** OSI model
   - **d.** All of these answers are correct.
4. The process of TCP on one computer marking a TCP segment as segment 1, and the receiving computer then acknowledging the receipt of TCP segment 1 is an example of what?
   - **a.** Data encapsulation
   - **b.** Same-layer interaction
   - **c.** Adjacent-layer interaction
   - **d.** OSI model
   - **e.** All of these answers are correct.

5. The process of a web server adding a TCP header to the contents of a web page, followed by adding an IP header and then adding a data-link header and trailer, is an example of what?
   a. Data encapsulation
   b. Same-layer interaction
   c. OSI model
   d. All of these answers are correct.
6. Which of the following terms is used specifically to identify the entity created when encapsulating data inside data-link layer headers and trailers?
   a. Data
   b. Chunk
   c. Segment
   d. Frame
   e. Packet
7. Which OSI encapsulation term can be used instead of the term frame?
   a. Layer 1 PDU
   b. Layer 2 PDU
   c. Layer 3 PDU
   d. Layer 5 PDU
   e. Layer 7 PDU

Foundation Topics

Perspectives on Networking

So, you are new to networking. Like many people, your perspective about networks might be that of a user of the network, as opposed to the network engineer who builds networks. For some, your view of networking might be based on how you use the Internet, from home, using a high-speed Internet connection like digital subscriber line (DSL) or cable TV, as shown in Figure 1-1.

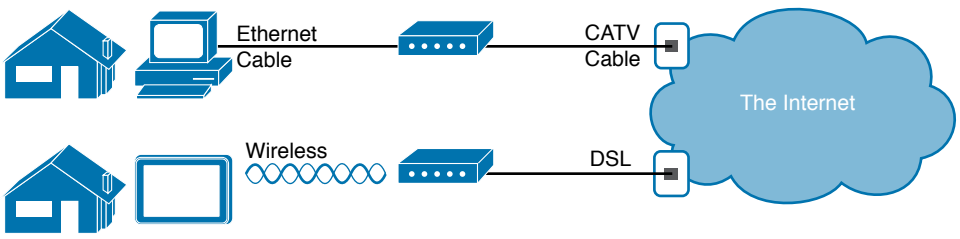


Figure 1-1 *End-User Perspective on High-Speed Internet Connections*

The top part of the figure shows a typical high-speed cable Internet user. The PC connects to a cable modem using an Ethernet cable. The cable modem then connects to a cable TV (CATV) outlet in the wall using a round coaxial cable—the same kind of cable used to connect your TV to the CATV wall outlet. Because cable Internet services provide service continuously, the user can just sit down at the PC and start sending email, browsing websites, making Internet phone calls, and using other tools and applications.

The lower part of the figure uses two different technologies. First, the tablet computer uses wireless technology that goes by the name wireless local-area network (wireless LAN), or Wi-Fi, instead of using an Ethernet cable. In this example, the router uses a different technology, DSL, to communicate with the Internet.

Both home-based networks and networks built for use by a company make use of similar networking technologies. The Information Technology (IT) world refers to a network created by one corporation, or enterprise, for the purpose of allowing its employees to communicate, as an *enterprise network*. The smaller networks at home, when used for business purposes, often go by the name small office/home office (SOHO) networks.

Users of enterprise networks have some idea about the enterprise network at their company or school. People realize that they use a network for many tasks. PC users might realize that their PC connects through an Ethernet cable to a matching wall outlet, as shown at the top of Figure 1-2. Those same users might use wireless LANs with their laptop when going to a meeting in the conference room as well. Figure 1-2 shows these two end-user perspectives on an enterprise network.

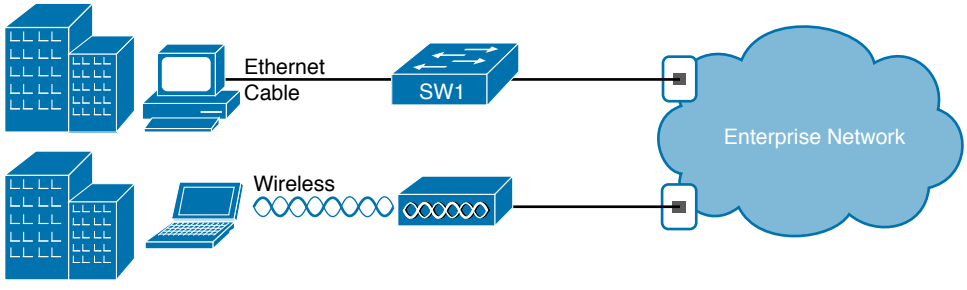


Figure 1-2 *Example Representation of an Enterprise Network*

NOTE In networking diagrams, a cloud represents a part of a network whose details are not important to the purpose of the diagram. In this case, Figure 1-2 ignores the details of how to create an enterprise network.

Some users might not even have a concept of the network at all. Instead, these users just enjoy the functions of the network—the ability to post messages to social media sites, make phone calls, search for information on the Internet, listen to music, and download countless apps to their phones—without caring about how it works or how their favorite device connects to the network.

Regardless of how much you already know about how networks work, this book and the related certification help you learn how networks do their job. That job is simply this: moving data from one device to another. The rest of this chapter, and the rest of this first

part of the book, reveals the basics of how to build enterprise networks so that they can deliver data between two devices.

TCP/IP Networking Model

A *networking model*, sometimes also called either a *networking architecture* or *networking blueprint*, refers to a comprehensive set of documents. Individually, each document describes one small function required for a network; collectively, these documents define everything that should happen for a computer network to work. Some documents define a *protocol*, which is a set of logical rules that devices must follow to communicate. Other documents define some physical requirements for networking. For example, a document could define the voltage and current levels used on a particular cable when transmitting data.

You can think of a networking model as you think of an architectural blueprint for building a house. Sure, you can build a house without the blueprint. However, the blueprint can ensure that the house has the right foundation and structure so that it will not fall down, and it has the correct hidden spaces to accommodate the plumbing, electrical, gas, and so on. Also, the many different people that build the house using the blueprint—such as framers, electricians, bricklayers, painters, and so on—know that if they follow the blueprint, their part of the work should not cause problems for the other workers.

Similarly, you could build your own network—write your own software, build your own networking cards, and so on—to create a network. However, it is much easier to simply buy and use products that already conform to some well-known networking model or blueprint. Because the networking product vendors build their products with some networking model in mind, their products should work well together.

History Leading to TCP/IP

Today, the world of computer networking uses one networking model: TCP/IP. However, the world has not always been so simple. Once upon a time, networking protocols didn't exist, including TCP/IP. Vendors created the first networking protocols; these protocols supported only that vendor's computers.

For example, IBM, the computer company with the largest market share in many markets back in the 1970s and 1980s, published its Systems Network Architecture (SNA) networking model in 1974. Other vendors also created their own proprietary networking models. As a result, if your company bought computers from three vendors, network engineers often had to create three different networks based on the networking models created by each company, and then somehow connect those networks, making the combined networks much more complex. The left side of Figure 1-3 shows the general idea of what a company's enterprise network might have looked like back in the 1980s, before TCP/IP became common in enterprise internetworks.

Answers to the "Do I Know This Already?" quiz:

1 D and F **2** A and G **3** B **4** B **5** A **6** D **7** B

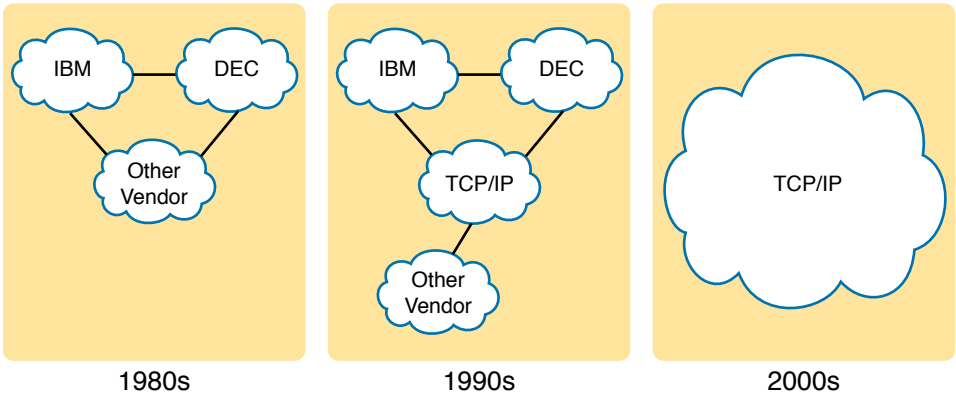


Figure 1-3 *Historical Progression: Proproprietary Models to the Open TCP/IP Model*

Although vendor-defined proprietary networking models often worked well, having an open, vendor-neutral networking model would aid competition and reduce complexity. The International Organization for Standardization (ISO) took on the task to create such a model, starting as early as the late 1970s, beginning work on what would become known as the Open Systems Interconnection (OSI) networking model. ISO had a noble goal for the OSI model: to standardize data networking protocols to allow communication among all computers across the entire planet. ISO worked toward this ambitious and noble goal, with participants from most of the technologically developed nations on Earth participating in the process.

A second, less-formal effort to create an open, vendor-neutral, public networking model sprouted forth from a U.S. Department of Defense (DoD) contract. Researchers at various universities volunteered to help further develop the protocols surrounding the original DoD work. These efforts resulted in a competing open networking model called TCP/IP.

During the 1990s, companies began adding OSI, TCP/IP, or both to their enterprise networks. However, by the end of the 1990s, TCP/IP had become the common choice, and OSI fell away. The center part of Figure 1-3 shows the general idea behind enterprise networks in that decade—still with networks built upon multiple networking models but including TCP/IP.

Here in the twenty-first century, TCP/IP dominates. Proprietary networking models still exist, but they have mostly been discarded in favor of TCP/IP. The OSI model, whose development suffered in part because of a slower formal standardization process as compared with TCP/IP, never succeeded in the marketplace. And TCP/IP, the networking model originally created almost entirely by a bunch of volunteers, has become the most prolific network model ever, as shown on the right side of Figure 1-3.

In this chapter, you will read about some of the basics of TCP/IP. Although you will learn some interesting facts about TCP/IP, the true goal of this chapter is to help you understand what a networking model or networking architecture really is and how it works.

Also in this chapter, you will learn about some of the jargon used with OSI. Will any of you ever work on a computer that is using the full OSI protocols instead of TCP/IP? Probably not. However, you will often use terms relating to OSI.

Overview of the TCP/IP Networking Model

The TCP/IP model both defines and references a large collection of protocols that allow computers to communicate. To define a protocol, TCP/IP uses documents called *Requests For Comments* (RFC). (You can find these RFCs using any online search engine.) The TCP/IP model also avoids repeating work already done by some other standards body or vendor consortium by simply referring to standards or protocols created by those groups. For example, the Institute of Electrical and Electronic Engineers (IEEE) defines Ethernet LANs; the TCP/IP model does not define Ethernet in RFCs, but refers to IEEE Ethernet as an option.

The TCP/IP model creates a set of rules that allows us all to take a computer (or mobile device) out of the box, plug in all the right cables, turn it on, and connect to and use the network. You can use a web browser to connect to your favorite website, use most any app, and it all works. How? Well, the OS on the computer implements parts of the TCP/IP model. The Ethernet card, or wireless LAN card, built in to the computer implements some LAN standards referenced by the TCP/IP model. In short, the vendors that created the hardware and software implemented TCP/IP.

To help people understand a networking model, each model breaks the functions into a small number of categories called *layers*. Each layer includes protocols and standards that relate to that category of functions, as shown in Figure 1-4.

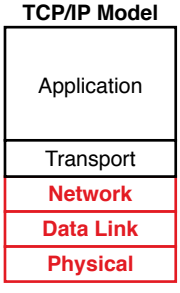


Figure 1-4 *The TCP/IP Networking Models*

The TCP/IP model shows the more common terms and layers used when people talk about TCP/IP today. The bottom layer focuses on how to transmit bits over each individual link. The data-link layer focuses on sending data over one type of physical link: for instance, networks use different data-link protocols for Ethernet LANs versus wireless LANs. The network layer focuses on delivering data over the entire path from the original sending computer to the final destination computer. And the top two layers focus more on the applications that need to send and receive data.

NOTE A slightly different four-layer original version of the TCP/IP model exists in RFC 1122, but for the purposes of both real networking and for today's CCNA, use the five-layer model shown here in Figure 1-4.

Many of you will have already heard of several TCP/IP protocols, like the examples listed in Table 1-2. Most of the protocols and standards in this table will be explained in more detail as you work through this book. Following the table, this section takes a closer look at the layers of the TCP/IP model.

Table 1-2 TCP/IP Architectural Model and Example Protocols

TCP/IP Architecture Layer	Example Protocols
Application	HTTP, POP3, SMTP
Transport	TCP, UDP
Network	IP, ICMP
Data Link & Physical	Ethernet, 802.11 (Wi-Fi)

TCP/IP Application Layer

TCP/IP application layer protocols provide services to the application software running on a computer. The application layer does not define the application itself, but it defines services that applications need. For example, application protocol HTTP defines how web browsers can pull the contents of a web page from a web server. In short, the application layer provides an interface between software running on a computer and the network itself.

Arguably, the most popular TCP/IP application today is the web browser. Many major software vendors either have already changed or are changing their application software to support access from a web browser. And thankfully, using a web browser is easy: You start a web browser on your computer and select a website by typing the name of the website, and the web page appears.

HTTP Overview

What really happens to allow that web page to appear on your web browser?

Imagine that Bob opens his browser. His browser has been configured to automatically ask for web server Larry's default web page, or *home page*. The general logic looks like Figure 1-5.

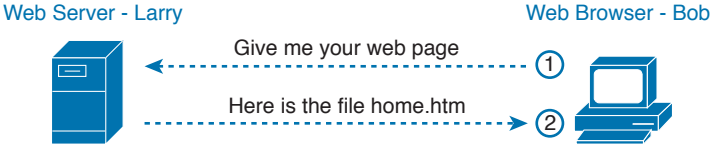


Figure 1-5 *Basic Application Logic to Get a Web Page*

So, what really happened? Bob's initial request actually asks Larry to send his home page back to Bob. Larry's web server software has been configured to know that the default web page is contained in a file called home.htm. Bob receives the file from Larry and displays the contents of the file in Bob's web browser window.

HTTP Protocol Mechanisms

Taking a closer look, this example shows how applications on each endpoint computer—specifically, the web browser application and web server application—use a TCP/IP application layer protocol. To make the request for a web page and return the contents of the web page, the applications use the Hypertext Transfer Protocol (HTTP).

HTTP did not exist until Tim Berners-Lee created the first web browser and web server in the early 1990s. Berners-Lee gave HTTP functionality to ask for the contents of web pages, specifically by giving the web browser the ability to request files from the server and giving the server a way to return the content of those files. The overall logic matches what was shown in Figure 1-5; Figure 1-6 shows the same idea, but with details specific to HTTP.

NOTE The full version of most web addresses—also called Uniform Resource Locators (URL) or Universal Resource Identifiers (URI)—begins with the letters *http*, which means that HTTP is used to transfer the web pages.

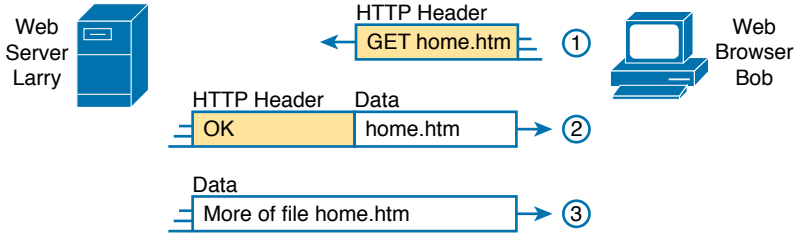


Figure 1-6 *HTTP GET Request, HTTP Reply, and One Data-Only Message*

To get the web page from Larry, at Step 1, Bob sends a message with an HTTP header. Generally, protocols use headers as a place to put information used by that protocol. This HTTP header includes the request to "get" a file. The request typically contains the name of the file (home.htm, in this case), or if no filename is mentioned, the web server assumes that Bob wants the default web page.

Step 2 in Figure 1-6 shows the response from web server Larry. The message begins with an HTTP header, with a return code (200), which means something as simple as "OK" returned in the header. HTTP also defines other return codes so that the server can tell the browser whether the request worked. (Here is another example: If you ever looked for a web page that was not found, and then received an HTTP 404 "not found" error, you received an HTTP return code of 404.) The second message also includes the first part of the requested file.

Step 3 in Figure 1-6 shows another message from web server Larry to web browser Bob, but this time without an HTTP header. HTTP transfers the data by sending multiple messages, each with a part of the file. Rather than wasting space by sending repeated HTTP headers that list the same information, these additional messages simply omit the header.

TCP/IP Transport Layer

Although many TCP/IP application layer protocols exist, the TCP/IP transport layer includes a smaller number of protocols. The two most commonly used transport layer protocols are the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP).

Transport layer protocols provide services to the application layer protocols that reside one layer higher in the TCP/IP model. How does a transport layer protocol provide a service to a higher-layer protocol? This section introduces that general concept by focusing on a single service provided by TCP: error recovery. The *CCNA 200-301 Official Cert Guide*, Volume 2, includes a chapter, "Introduction to TCP/IP Transport and Applications," which examines the transport layer.

TCP Error Recovery Basics

To appreciate what the transport layer protocols do, you must think about the layer above the transport layer, the application layer. Why? Well, each layer provides a service to the layer above it, like the error-recovery service provided to application layer protocols by TCP.

For example, in Figure 1-5, Bob and Larry used HTTP to transfer the home page from web server Larry to Bob's web browser. But what would have happened if Bob's HTTP GET request had been lost in transit through the TCP/IP network? Or, what would have happened if Larry's response, which included the contents of the home page, had been lost? Well, as you might expect, in either case, the page would not have shown up in Bob's browser.

TCP/IP needs a mechanism to guarantee delivery of data across a network. Because many application layer protocols probably want a way to guarantee delivery of data across a network, the creators of TCP included an error-recovery feature. To recover from errors, TCP uses the concept of acknowledgments. Figure 1-7 outlines the basic idea behind how TCP notices lost data and asks the sender to try again.

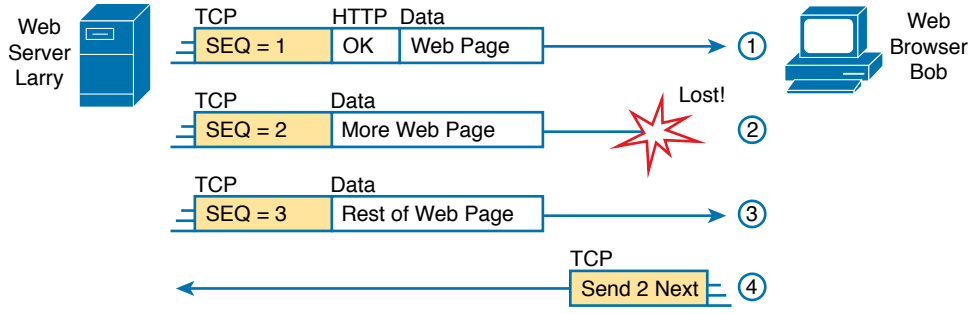


Figure 1-7 *TCP Error-Recovery Services as Provided to HTTP*

Figure 1-7 shows web server Larry sending a web page to web browser Bob, using three separate messages. Note that this figure shows the same HTTP headers as Figure 1-6, but it also shows a TCP header. The TCP header shows a sequence number (SEQ) with each message. In this example, the network has a problem, and the network fails to deliver the TCP message (called a segment) with sequence number 2. When Bob receives messages with sequence numbers 1 and 3, but does not receive a message with sequence number 2, Bob realizes that message 2 was lost. That realization by Bob's TCP logic causes Bob to send a TCP segment back to Larry, asking Larry to send message 2 again.

Same-Layer and Adjacent-Layer Interactions

Figure 1-7 also demonstrates a function called *adjacent-layer interaction*, which refers to the concepts of how adjacent layers in a networking model, on the same computer, work together. In this example, the higher-layer protocol (HTTP) wants error recovery, so it uses the next lower-layer protocol (TCP) to perform the service of error recovery; the lower layer provides a service to the layer above it.

Figure 1-7 also shows an example of a similar function called *same-layer interaction*. When a particular layer on one computer wants to communicate with the same layer on another computer, the two computers use headers to hold the information that they want

to communicate. For example, in Figure 1-7, Larry set the sequence numbers to 1, 2, and 3 so that Bob could notice when some of the data did not arrive. Larry's TCP process created that TCP header with the sequence number; Bob's TCP process received and reacted to the TCP segments.

Table 1-3 summarizes the key points about how adjacent layers work together on a single computer and how one layer on one computer works with the same networking layer on another computer.

Key Topic

Table 1-3 Summary: Same-Layer and Adjacent-Layer Interactions

Concept	Description
Same-layer interaction on different computers	The two computers use a protocol to communicate with the same layer on another computer. The protocol defines a header that communicates what each computer wants to do.
Adjacent-layer interaction on the same computer	On a single computer, one lower layer provides a service to the layer just above. The software or hardware that implements the higher layer requests that the next lower layer perform the needed function.

TCP/IP Network Layer

The application layer includes many protocols. The transport layer includes fewer protocols, most notably, TCP and UDP. The TCP/IP network layer includes a small number of protocols, but only one major protocol: the Internet Protocol (IP). In fact, the name TCP/IP is simply the names of the two most common protocols (TCP and IP) separated by a /.

IP provides several features, most importantly, addressing and routing. This section begins by comparing IP's addressing and routing with another commonly known system that uses addressing and routing: the postal service. Following that, this section introduces IP addressing and routing. (More details follow in Chapter 3, "Fundamentals of WANs and IP Routing.")

Internet Protocol and the Postal Service

Imagine that you just wrote two letters: one to a friend on the other side of the country and one to a friend on the other side of town. You addressed the envelopes and put on the stamps, so both are ready to give to the postal service. Is there much difference in how you treat each letter? Not really. Typically, you would just put them in the same mailbox and expect the postal service to deliver both letters.

The postal service, however, must think about each letter separately, and then make a decision of where to send each letter so that it is delivered. For the letter sent across town, the people in the local post office probably just need to put the letter on another truck.

For the letter that needs to go across the country, the postal service sends the letter to another post office, then another, and so on, until the letter gets delivered across the country. At each post office, the postal service must process the letter and choose where to send it next.

To make it all work, the postal service has regular routes for small trucks, large trucks, planes, boats, and so on, to move letters between postal service sites. The service must be able to receive and forward the letters, and it must make good decisions about where to send each letter next, as shown in Figure 1-8.

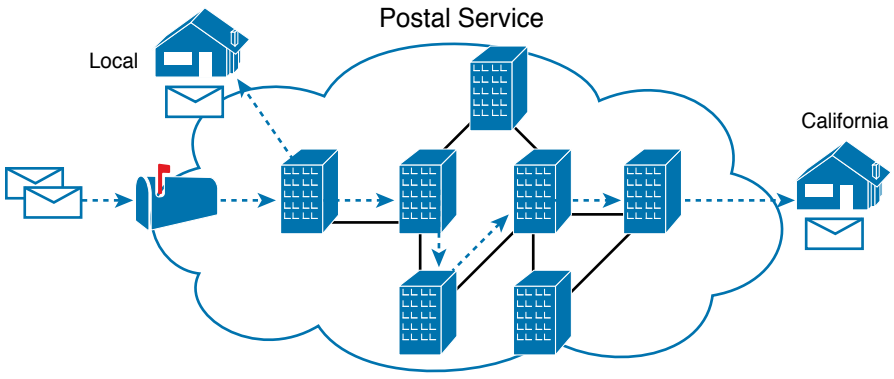


Figure 1-8 *Postal Service Forwarding (Routing) Letters*

Still thinking about the postal service, consider the difference between the person sending the letter and the work that the postal service does. The person sending the letters expects that the postal service will deliver the letter most of the time. However, the person sending the letter does not need to know the details of exactly what path the letters take. In contrast, the postal service does not create the letter, but it accepts the letter from the customer. Then, the postal service must know the details about addresses and postal codes that group addresses into larger groups, and it must have the ability to deliver the letters.

The TCP/IP application and transport layers act like the person sending letters through the postal service. These upper layers work the same way regardless of whether the endpoint host computers are on the same LAN or are separated by the entire Internet. To send a message, these upper layers ask the layer below them, the network layer, to deliver the message.

The lower layers of the TCP/IP model act more like the postal service to deliver those messages to the correct destinations. To do so, these lower layers must understand the underlying physical network because they must choose how to best deliver the data from one host to another.

So, what does this all matter to networking? Well, the network layer of the TCP/IP networking model, primarily defined by the Internet Protocol (IP), works much like the postal service. IP defines that each host computer should have a different IP address, just as the postal service defines addressing that allows unique addresses for each house, apartment, and business. Similarly, IP defines the process of routing so that devices called routers can work like the post office, forwarding packets of data so that they are delivered to the correct destinations. Just as the postal service created the necessary infrastructure to deliver letters—post offices, sorting machines, trucks, planes, and personnel—the network layer defines the details of how a network infrastructure should be created so that the network can deliver data to all computers in the network.

Internet Protocol Addressing Basics

IP defines addresses for several important reasons. First, each device that uses TCP/IP—each TCP/IP *host*—needs a unique address so that it can be identified in the network. IP also defines how to group addresses together, just like the postal system groups addresses based on postal codes (like ZIP codes in the United States).

To understand the basics, examine Figure 1-9, which shows the familiar web server Larry and web browser Bob; but now, instead of ignoring the network between these two computers, part of the network infrastructure is included.

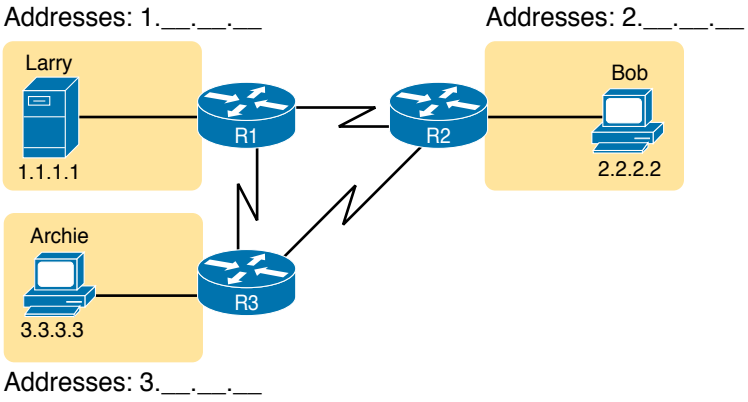


Figure 1-9 *Simple TCP/IP Network: Three Routers with IP Addresses Grouped*

First, note that Figure 1-9 shows some sample IP addresses. Each IP address has four numbers, separated by periods. In this case, Larry uses IP address 1.1.1.1, and Bob uses 2.2.2.2. This style of number is called a dotted-decimal notation (DDN).

Figure 1-9 also shows three groups of addresses. In this example, all IP addresses that begin with 1 must be on the upper left, as shown in shorthand in the figure as 1.__.__.__. All addresses that begin with 2 must be on the right, as shown in shorthand as 2.__.__.__. Finally, all IP addresses that begin with 3 must be at the bottom of the figure.

In addition, Figure 1-9 introduces icons that represent IP routers. Routers are networking devices that connect the parts of the TCP/IP network together for the purpose of routing (forwarding) IP packets to the correct destination. Routers do the equivalent of the work done by each post office site: They receive IP packets on various physical interfaces, make decisions based on the IP address included with the packet, and then physically forward the packet out some other network interface.

IP Routing Basics

The TCP/IP network layer, using the IP protocol, provides a service of forwarding IP packets from one device to another. Any device with an IP address can connect to the TCP/IP network and send packets. This section shows a basic IP routing example for perspective.

NOTE The term *IP host* refers to any device, regardless of size or power, that has an IP address and connects to any TCP/IP network.

Figure 1-10 repeats the familiar case in which web server Larry wants to send part of a web page to Bob, but now with details related to IP. On the lower left, note that server Larry has the familiar application data, HTTP header, and TCP header ready to send. In addition, the message now contains an IP header. The IP header includes a source IP address of Larry's IP address (1.1.1.1) and a destination IP address of Bob's IP address (2.2.2.2).

Key Topic

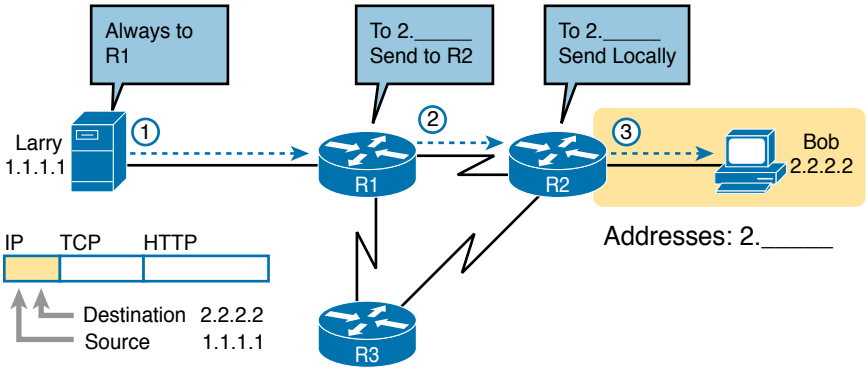


Figure 1-10 *Basic Routing Example*

Step 1, on the left of Figure 1-10, begins with Larry being ready to send an IP packet. Larry's IP process chooses to send the packet to some router—a nearby router on the same LAN—with the expectation that the router will know how to forward the packet. (This logic is much like you or me sending all our letters by putting them in a nearby mailbox.) Larry doesn't need to know anything more about the topology or the other routers.

At Step 2, Router R1 receives the IP packet, and R1's IP process makes a decision. R1 looks at the destination address (2.2.2.2), compares that address to its known IP routes, and chooses to forward the packet to Router R2. This process of forwarding the IP packet is called *IP routing* (or simply *routing*).

At Step 3, Router R2 repeats the same kind of logic used by Router R1. R2's IP process will compare the packet's destination IP address (2.2.2.2) to R2's known IP routes and make a choice to forward the packet to the right, on to Bob.

You will learn IP in more depth than any other protocol while preparing for CCNA. More than half the chapters in this book discuss some feature that relates to addressing, IP routing, and how routers perform routing.

TCP/IP Data-Link and Physical Layers

The TCP/IP model's data-link and physical layers define the protocols and hardware required to deliver data across some physical network. The two work together quite closely; in fact, some standards define both the data-link and physical layer functions. The physical layer defines the cabling and energy (for example, electrical signals) that flow over the cables. Some rules and conventions exist when sending data over the cable; however, those rules exist in the data-link layer of the TCP/IP model.

Focusing on the data-link layer for a moment, just like every layer in any networking model, the TCP/IP data-link layer provides services to the layer above it in the model (the network layer). When a host's or router's IP process chooses to send an IP packet to another router or host, that host or router then uses link-layer details to send that packet to the next host/router.

Because each layer provides a service to the layer above it, take a moment to think about the IP logic related to Figure 1-10. In that example, host Larry's IP logic chooses to send the IP

packet to a nearby router (R1). However, while Figure 1-10 shows a simple line between Larry and router R1, that drawing means that some Ethernet LAN sits between the two. Figure 1-11 shows four steps of what occurs at the link layer to allow Larry to send the IP packet to R1.

Key Topic

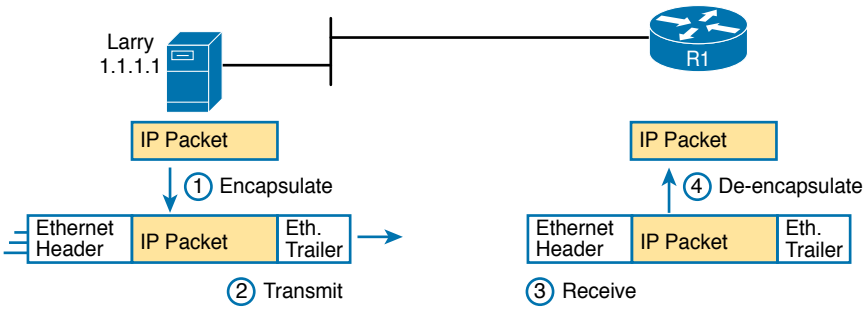


Figure 1-11 *Larry Using Ethernet to Forward an IP Packet to Router R1*

NOTE Figure 1-11 depicts the Ethernet as a series of lines. Networking diagrams often use this convention when drawing Ethernet LANs, in cases where the actual LAN cabling and LAN devices are not important to some discussion, as is the case here. The LAN would have cables and devices, like LAN switches, which are not shown in this figure.

Figure 1-11 shows four steps. The first two occur on Larry, and the last two occur on Router R1, as follows:

Step 1. Larry encapsulates the IP packet between an Ethernet header and Ethernet trailer, creating an Ethernet *frame*.

Step 2. Larry physically transmits the bits of this Ethernet frame, using electricity flowing over the Ethernet cabling.

Step 3. Router R1 physically receives the electrical signal over a cable and re-creates the same bits by interpreting the meaning of the electrical signals.

Step 4. Router R1 de-encapsulates the IP packet from the Ethernet frame by removing and discarding the Ethernet header and trailer.

By the end of this process, Larry and R1 have worked together to deliver the packet from Larry to Router R1.

NOTE Protocols define both headers and trailers for the same general reason, but headers exist at the beginning of the message and trailers exist at the end.

The data-link and physical layers include a large number of protocols and standards. For example, the link layer includes all the variations of Ethernet protocols and wireless LAN protocols discussed throughout this book.

In short, the TCP/IP physical and data-link layers include two distinct functions, respectively: functions related to the physical transmission of the data, plus the protocols and rules that control the use of the physical media.

Data Encapsulation Terminology

As you can see from the explanations of how HTTP, TCP, IP, and Ethernet do their jobs, when sending data, each layer adds its own header (and for data-link protocols, also a trailer) to the data supplied by the higher layer. The term *encapsulation* refers to the process of putting headers (and sometimes trailers) around some data.

Many of the examples in this chapter show the encapsulation process. For example, web server Larry encapsulated the contents of the home page inside an HTTP header in Figure 1-6. The TCP layer encapsulated the HTTP headers and data inside a TCP header in Figure 1-7. IP encapsulated the TCP headers and the data inside an IP header in Figure 1-10. Finally, the Ethernet link layer encapsulated the IP packets inside both a header and a trailer in Figure 1-11.

The process by which a TCP/IP host sends data can be viewed as a five-step process. The first four steps relate to the encapsulation performed by the four TCP/IP layers, and the last step is the actual physical transmission of the data by the host. In fact, if you use the five-layer TCP/IP model, one step corresponds to the role of each layer. The steps are summarized in the following list:

Step 1. **Create and encapsulate the application data with any required application layer headers.** For example, the HTTP OK message can be returned in an HTTP header, followed by part of the contents of a web page.

Step 2. **Encapsulate the data supplied by the application layer inside a transport layer header.** For end-user applications, a TCP or UDP header is typically used.

Step 3. **Encapsulate the data supplied by the transport layer inside a network layer (IP) header.** IP defines the IP addresses that uniquely identify each computer.

Step 4. **Encapsulate the data supplied by the network layer inside a data-link layer header and trailer.** This layer uses both a header and a trailer.

Step 5. **Transmit the bits.** The physical layer encodes a signal onto the medium to transmit the frame.

The numbers in Figure 1-12 correspond to the five steps in this list, graphically showing the same concepts. Note that because the application layer often does not need to add a header, the figure does not show a specific application layer header, but the application layer will also at times add a header as well.

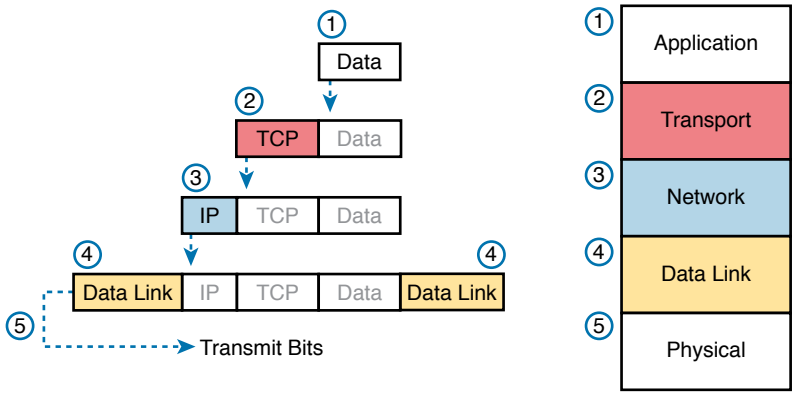


Figure 1-12 *Five Steps of Data Encapsulation: TCP/IP*

Names of TCP/IP Messages

One reason this chapter takes the time to show the encapsulation steps in detail has to do with terminology. When talking and writing about networking, people use *segment*, *packet*, and *frame* to refer to the messages shown in Figure 1-13 and the related list. Each term has a specific meaning, referring to the headers (and possibly trailers) defined by a particular layer and the data encapsulated following that header. Each term, however, refers to a different layer: segment for the transport layer, packet for the network layer, and frame for the link layer. Figure 1-13 shows each layer along with the associated term.

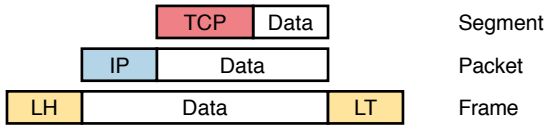


Figure 1-13 *Perspectives on Encapsulation and "Data"**

* The letters LH and LT stand for link header and link trailer, respectively, and refer to the data-link layer header and trailer.

Figure 1-13 also shows the encapsulated data as simply "data." When focusing on the work done by a particular layer, the encapsulated data typically is unimportant. For example, an IP packet can indeed have a TCP header after the IP header, an HTTP header after the TCP header, and data for a web page after the HTTP header. However, when discussing IP, you probably just care about the IP header, so everything after the IP header is just called data. So, when drawing IP packets, everything after the IP header is typically shown simply as data.

OSI Networking Model and Terminology

At one point in the history of the OSI model, many people thought that OSI would win the battle of the networking models discussed earlier. If that had occurred, instead of running TCP/IP on every computer in the world, those computers would be running with OSI.

However, OSI did not win that battle. In fact, OSI no longer exists as a networking model that could be used instead of TCP/IP, although some of the original protocols referenced by the OSI model still exist.

So, why is OSI even in this book? Terminology. During those years in which many people thought the OSI model would become commonplace in the world of networking (mostly in the late 1980s and early 1990s), many vendors and protocol documents started using terminology from the OSI model. That terminology remains today. So, while you will never need to work with a computer that uses OSI, to understand modern networking terminology, you need to understand something about OSI.

Comparing OSI and TCP/IP Layer Names and Numbers

The OSI model has many similarities to the TCP/IP model from a basic conceptual perspective. It has layers, and each layer defines a set of typical networking functions. As with TCP/IP, the OSI layers each refer to multiple protocols and standards that implement the functions specified by each layer. In other cases, just as for TCP/IP, the OSI committees did not create new protocols or standards, but instead referenced other protocols that were already defined. For example, the IEEE defines Ethernet standards, so the OSI committees did not waste time specifying a new type of Ethernet; it simply referred to the IEEE Ethernet standards.

Today, the OSI model can be used as a standard of comparison to other networking models. Figure 1-14 compares the seven-layer OSI model with both the four-layer and five-layer TCP/IP models.

Key Topic

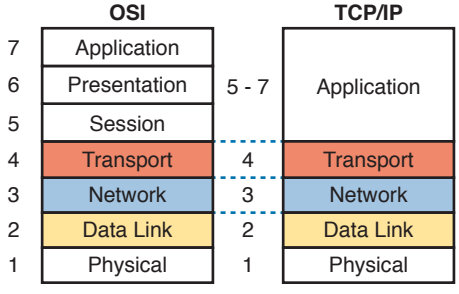


Figure 1-14 *OSI Model Compared to the Two TCP/IP Models*

Note that the TCP/IP model in use today, on the right side of the figure, uses the exact same layer names as OSI at the lower layers. The functions generally match as well, so for the purpose of discussing networking, and reading networking documentation, think of the bottom four layers as equivalent, in name, in number, and in meaning.

Even though the world uses TCP/IP today rather than OSI, we tend to use the numbering from the OSI layer. For instance, when referring to an application layer protocol in a TCP/IP network, the world still refers to the protocol as a "Layer 7 protocol." Also, while TCP/IP includes more functions at its application layer, OSI breaks those into session, presentation, and application layers. Most of the time, no one cares much about the distinction, so you will see references like "Layer 5–7 protocol," again using OSI numbering.

For the purposes of this book, know the mapping between the five-layer TCP/IP model and the seven-layer OSI model shown in Figure 1-14, and know that layer number references to Layer 7 really do match the application layer of TCP/IP as well.

OSI Data Encapsulation Terminology

Like TCP/IP, each OSI layer asks for services from the next lower layer. To provide the services, each layer makes use of a header and possibly a trailer. The lower layer encapsulates the higher layer's data behind a header.

OSI uses a more generic term to refer to messages, rather than frame, packet, and segment. OSI uses the term *protocol data unit* (PDU). A PDU represents the bits that include the headers and trailers for that layer, as well as the encapsulated data. For example, an IP packet, as shown in Figure 1-13, using OSI terminology, is a PDU, more specifically a *Layer 3 PDU* (abbreviated L3PDU) because IP is a Layer 3 protocol. OSI simply refers to the Layer *x* PDU (LxPDU), with *x* referring to the number of the layer being discussed, as shown in Figure 1-15.

Key Topic

L#H - Layer # Header
L#T - Layer # Trailer

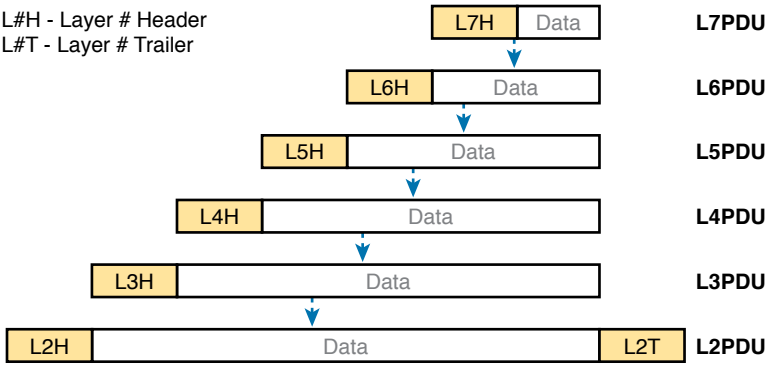


Figure 1-15 *OSI Encapsulation and Protocol Data Units*

Chapter Review

The "Your Study Plan" element, just before Chapter 1, discusses how you should study and practice the content and skills for each chapter before moving on to the next chapter. That element introduces the tools used here at the end of each chapter. If you haven't already done so, take a few minutes to read that section. Then come back here and do the useful work of reviewing the chapter to help lock into memory what you just read.

Review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 1-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 1-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP Online

Review All the Key Topics

Key Topic

Table 1-5 Key Topics for Chapter 1

Key Topic Elements	Description	Page Number
Table 1-3	Provides definitions of same-layer and adjacent-layer interaction	22
Figure 1-10	Shows the general concept of IP routing	25
Figure 1-11	Depicts the data-link services provided to IP for the purpose of delivering IP packets from host to host	26
Figure 1-12	Five steps to encapsulate data on the sending host	28
Figure 1-13	Shows the meaning of the terms *segment*, *packet*, and *frame*	28
Figure 1-14	Compares the OSI and TCP/IP network models	29
Figure 1-15	Terminology related to encapsulation	30

Key Terms You Should Know

adjacent-layer interaction, de-encapsulation, encapsulation, frame, networking model, packet, protocol data unit (PDU), same-layer interaction, segment

CHAPTER 2

Fundamentals of Ethernet LANs

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.b L2 and L3 Switches

1.2 Describe characteristics of network topology architectures

1.2.e Small office/home office (SOHO)

1.3 Compare physical interface and cabling types

1.3.a Single-mode fiber, multimode fiber, copper

1.3.b Connections (Ethernet shared media and point-to-point)

Most enterprise computer networks can be separated into two general types of technology: local-area networks (LANs) and wide-area networks (WANs). LANs typically connect nearby devices: devices in the same room, in the same building, or in a campus of buildings. In contrast, WANs connect devices that are typically relatively far apart. Together, LANs and WANs create a complete enterprise computer network, working together to do the job of a computer network: delivering data from one device to another.

Many types of LANs have existed over the years, but today's networks use two general types of LANs: Ethernet LANs and wireless LANs. Ethernet LANs happen to use cables for the links between nodes, and because many types of cables use copper wires, Ethernet LANs are often called *wired LANs*. Ethernet LANs also make use of fiber-optic cabling, which includes a fiberglass core that devices use to send data using light. In comparison to Ethernet, wireless LANs do not use wires or cables, instead using radio waves for the links between nodes; Part V of this book discusses Wireless LANs at length.

This chapter introduces Ethernet LANs, with more detailed coverage in Parts II and III of this book.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 2-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
An Overview of LANs	1–2
Building Physical Ethernet LANs with UTP	3–4
Building Physical Ethernet LANs with Fiber	5
Sending Data in Ethernet Networks	6–9

1. In the LAN for a small office, some user devices connect to the LAN using a cable, while others connect using wireless technology (and no cable). Which of the following is true regarding the use of Ethernet in this LAN?
   a. Only the devices that use cables are using Ethernet.
   b. Only the devices that use wireless are using Ethernet.
   c. Both the devices using cables and those using wireless are using Ethernet.
   d. None of the devices are using Ethernet.
2. Which of the following Ethernet standards defines Gigabit Ethernet over UTP cabling?
   a. 10GBASE-T
   b. 100BASE-T
   c. 1000BASE-T
   d. None of the other answers is correct.
3. Which of the following is true about Ethernet crossover cables for Fast Ethernet?
   a. Pins 1 and 2 are reversed on the other end of the cable.
   b. Pins 1 and 2 on one end of the cable connect to pins 3 and 6 on the other end of the cable.
   c. Pins 1 and 2 on one end of the cable connect to pins 3 and 4 on the other end of the cable.
   d. The cable can be up to 1000 meters long to cross over between buildings.
   e. None of the other answers is correct.
4. Each answer lists two types of devices used in a 100BASE-T network. If these devices were connected with UTP Ethernet cables, which pairs of devices would require a straight-through cable? (Choose three answers.)
   a. PC and router
   b. PC and switch
   c. Hub and switch
   d. Router and hub
   e. Wireless access point (Ethernet port) and switch

5. Which of the following are advantages of using multimode fiber for an Ethernet link instead of UTP or single-mode fiber?
   a. To achieve the longest distance possible for that single link.
   b. To extend the link beyond 100 meters while keeping initial costs as low as possible.
   c. To make use of an existing stock of laser-based SFP/SFP+ modules.
   d. To make use of an existing stock of LED-based SFP/SFP+ modules.
6. Which of the following is true about the CSMA/CD algorithm?
   a. The algorithm never allows collisions to occur.
   b. Collisions can happen, but the algorithm defines how the computers should notice a collision and how to recover.
   c. The algorithm works with only two devices on the same Ethernet.
   d. None of the other answers is correct.
7. Which of the following is true about the Ethernet FCS field?
   a. Ethernet uses FCS for error recovery.
   b. It is 2 bytes long.
   c. It resides in the Ethernet trailer, not the Ethernet header.
   d. It is used for encryption.
8. Which of the following are true about the format of Ethernet addresses? (Choose three answers.)
   a. Each manufacturer puts a unique OUI code into the first 2 bytes of the address.
   b. Each manufacturer puts a unique OUI code into the first 3 bytes of the address.
   c. Each manufacturer puts a unique OUI code into the first half of the address.
   d. The part of the address that holds this manufacturer's code is called the MAC.
   e. The part of the address that holds this manufacturer's code is called the OUI.
   f. The part of the address that holds this manufacturer's code has no specific name.
9. Which of the following terms describe Ethernet addresses that can be used to send one frame that is delivered to multiple devices on the LAN? (Choose two answers.)
   a. Burned-in address
   b. Unicast address
   c. Broadcast address
   d. Multicast address

Foundation Topics

An Overview of LANs

The term *Ethernet* refers to a family of LAN standards that together define the physical and data-link layers of the world's most popular wired LAN technology. The standards, defined by the Institute of Electrical and Electronics Engineers (IEEE), define the cabling,

the connectors on the ends of the cables, the protocol rules, and everything else required to create an Ethernet LAN.

Typical SOHO LANs

To begin, first think about a small office/home office (SOHO) LAN today, specifically a LAN that uses only Ethernet LAN technology. First, the LAN needs a device called an Ethernet *LAN switch*, which provides many physical ports into which cables can be connected. An Ethernet uses *Ethernet cables*, which is a general reference to any cable that conforms to any of several Ethernet standards. The LAN uses Ethernet cables to connect different Ethernet devices or nodes to one of the switch's Ethernet ports.

Figure 2-1 shows a drawing of a SOHO Ethernet LAN. The figure shows a single LAN switch, five cables, and five other Ethernet nodes: three PCs, a printer, and one network device called a *router*. (The router connects the LAN to the WAN, in this case to the Internet.)

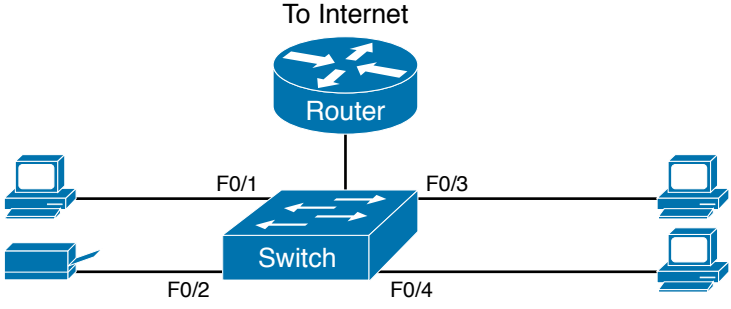


Figure 2-1 *Typical Small Ethernet-Only SOHO LAN*

Although Figure 2-1 shows the switch and router as separate devices, many SOHO Ethernet LANs today combine the router and switch into a single device. Vendors sell consumer-grade integrated networking devices that work as a router and Ethernet switch, as well as doing other functions. These devices typically have "router" on the packaging, but many models also have four-port or eight-port Ethernet LAN switch ports built in to the device.

Typical SOHO LANs today also support wireless LAN connections. You can build a single SOHO LAN that includes both Ethernet LAN technology as well as wireless LAN technology, which is also defined by the IEEE. Wireless LANs, defined by the IEEE using standards that begin with 802.11, use radio waves to send the bits from one node to the next.

Most wireless LANs rely on yet another networking device: a wireless LAN access point (AP). The AP acts somewhat like an Ethernet switch, in that all the wireless LAN nodes communicate with the wireless AP. If the network uses an AP that is a separate physical device, the AP then needs a single Ethernet link to connect the AP to the Ethernet LAN, as shown in Figure 2-2.

Note that Figure 2-2 shows the router, Ethernet switch, and wireless LAN access point as three separate devices so that you can better understand the different roles. However, most SOHO networks today would use a single device, often labeled as a "wireless router," that does all these functions.

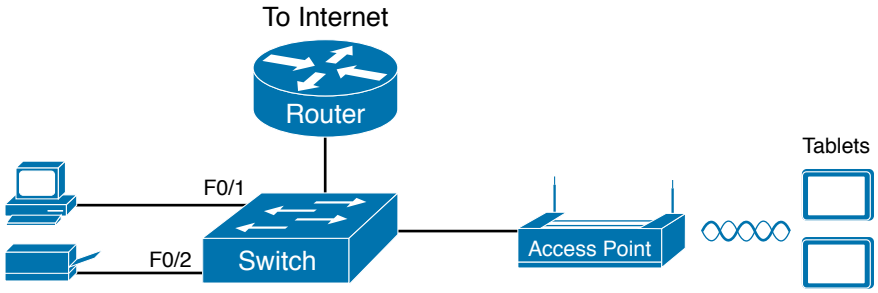


Figure 2-2 *Typical Small Wired and Wireless SOHO LAN*

Typical Enterprise LANs

Enterprise networks have similar needs compared to a SOHO network, but on a much larger scale. For example, enterprise Ethernet LANs begin with LAN switches installed in a wiring closet behind a locked door on each floor of a building. The electricians install the Ethernet cabling from that wiring closet to cubicles and conference rooms where devices might need to connect to the LAN. At the same time, most enterprises also support wireless LANs in the same space, to allow people to roam around and still work and to support a growing number of devices that do not have an Ethernet LAN interface.

Figure 2-3 shows a conceptual view of a typical enterprise LAN in a three-story building. Each floor has an Ethernet LAN switch and a wireless LAN AP. To allow communication between floors, each per-floor switch connects to one centralized distribution switch. For example, PC3 can send data to PC2, but it would first flow through switch SW3 to the first floor to the distribution switch (SWD) and then back up through switch SW2 on the second floor.

Key Topic

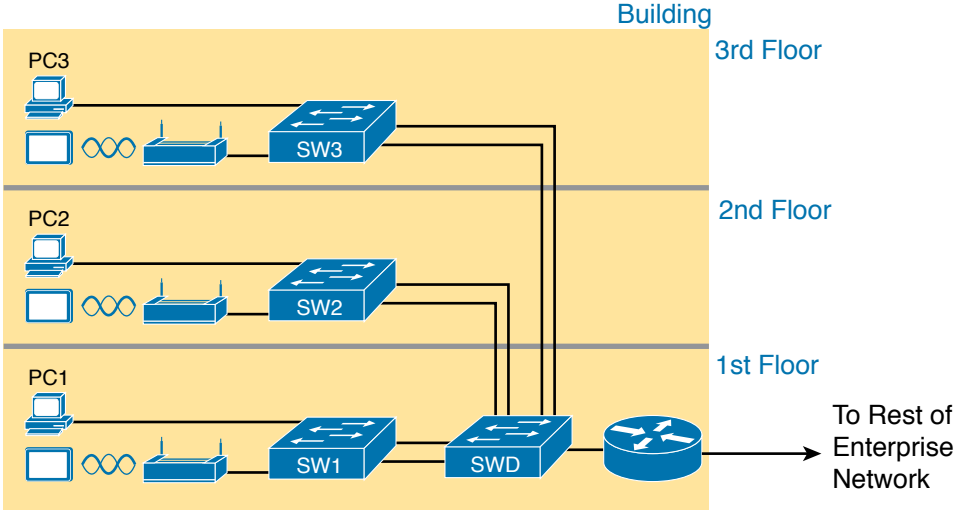


Figure 2-3 *Single-Building Enterprise Wired and Wireless LAN*

Answers to the "Do I Know This Already?" quiz:

1 A **2** C **3** B **4** B, D, and E **5** B **6** B **7** C **8** B, C, and E **9** C and D

The figure also shows the typical way to connect a LAN to a WAN using a router. LAN switches and wireless access points work to create the LAN itself. Routers connect to both the LAN and the WAN. To connect to the LAN, the router simply uses an Ethernet LAN interface and an Ethernet cable, as shown on the lower right of Figure 2-3.

The rest of this chapter focuses on Ethernet in particular.

The Variety of Ethernet Physical Layer Standards

The term *Ethernet* refers to an entire family of standards. Some standards define the specifics of how to send data over a particular type of cabling, and at a particular speed. Other standards define protocols, or rules, that the Ethernet nodes must follow to be a part of an Ethernet LAN. All these Ethernet standards come from the IEEE and include the number 802.3 as the beginning part of the standard name.

Ethernet supports a large variety of options for physical Ethernet links given its long history over the last 40 or so years. Today, Ethernet includes many standards for different kinds of optical and copper cabling, and for speeds from 10 megabits per second (Mbps) up to 400 gigabits per second (Gbps). The standards also differ as far as the types and length of the cables.

The most fundamental cabling choice has to do with the materials used inside the cable for the physical transmission of bits: either copper wires or glass fibers. Devices using UTP cabling transmit data over electrical circuits via the copper wires inside the cable. Fiber-optic cabling, the more expensive alternative, allows Ethernet nodes to send light over glass fibers in the center of the cable. Although more expensive, optical cables typically allow longer cabling distances between nodes.

To be ready to choose the products to purchase for a new Ethernet LAN, a network engineer must know the names and features of the different Ethernet standards supported in Ethernet products. The IEEE defines Ethernet physical layer standards using a couple of naming conventions. The formal name begins with 802.3 followed by some suffix letters. The IEEE also uses more meaningful shortcut names that identify the speed, as well as a clue about whether the cabling is UTP (with a suffix that includes *T*) or fiber (with a suffix that includes *X*). Table 2-2 lists a few Ethernet physical layer standards. First, the table lists enough names so that you get a sense of the IEEE naming conventions.

Key Topic

Table 2-2 Examples of Types of Ethernet

Speed	Common Name	Informal IEEE Standard Name	Formal IEEE Standard Name	Cable Type, Maximum Length
10 Mbps	Ethernet	10BASE-T	802.3	Copper, 100 m
100 Mbps	Fast Ethernet	100BASE-T	802.3u	Copper, 100 m
1000 Mbps	Gigabit Ethernet	1000BASE-LX	802.3z	Fiber, 5000 m
1000 Mbps	Gigabit Ethernet	1000BASE-T	802.3ab	Copper, 100 m
10 Gbps	10 Gig Ethernet	10GBASE-T	802.3an	Copper, 100 m

NOTE Fiber-optic cabling contains long thin strands of fiberglass. The attached Ethernet nodes send light over the glass fiber in the cable, encoding the bits as changes in the light.

NOTE You might expect that a standard that began at the IEEE almost 40 years ago would be stable and unchanging, but the opposite is true. The IEEE, along with active industry partners, continues to develop new Ethernet standards with longer distances, different cabling options, and faster speeds. Check out the Ethernet Alliance web page (www.EthernetAlliance.org)and look for the roadmap for some great graphics and tables about the latest happenings with Ethernet.

Consistent Behavior over All Links Using the Ethernet Data-Link Layer

Although Ethernet includes many physical layer standards, Ethernet acts like a single LAN technology because it uses the same data-link layer standard over all types of Ethernet physical links. That standard defines a common Ethernet header and trailer. (As a reminder, the header and trailer are bytes of overhead data that Ethernet uses to do its job of sending data over a LAN.) No matter whether the data flows over a UTP cable or any kind of fiber cable, and no matter the speed, the data-link header and trailer use the same format.

While the physical layer standards focus on sending bits over a cable, the Ethernet data-link protocols focus on sending an *Ethernet frame* from source to destination Ethernet node. From a data-link perspective, nodes build and forward frames. As first defined in Chapter 1, "Introduction to TCP/IP Networking," the term *frame* specifically refers to the header and trailer of a data-link protocol, plus the data encapsulated inside that header and trailer. The various Ethernet nodes simply forward the frame, over all the required links, to deliver the frame to the correct destination.

Figure 2-4 shows an example of the process. In this case, PC1 sends an Ethernet frame to PC3. The frame travels over a UTP link to Ethernet switch SW1, then over fiber links to Ethernet switches SW2 and SW3, and finally over another UTP link to PC3. Note that the bits actually travel at four different speeds in this example: 10 Mbps, 1 Gbps, 10 Gbps, and 100 Mbps, respectively.

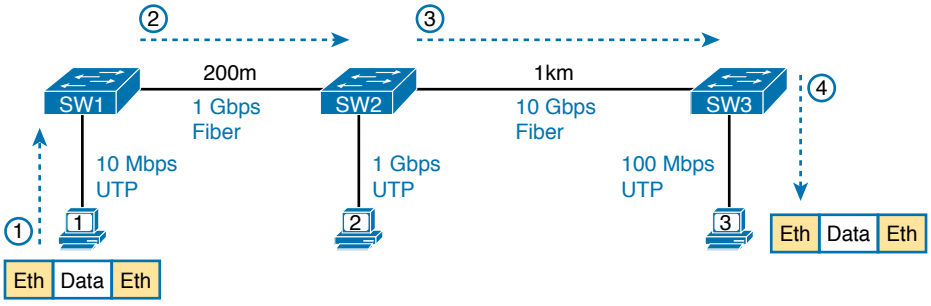


Figure 2-4 *Ethernet LAN Forwards a Data-Link Frame over Many Types of Links*

So, what is an Ethernet LAN? It is a combination of user devices, LAN switches, and different kinds of cabling. Each link can use different types of cables, at different speeds.

However, they all work together to deliver Ethernet frames from the one device on the LAN to some other device.

The rest of this chapter takes these concepts a little deeper. The next section examines how to build a physical Ethernet network using UTP cabling, followed by a similar look at using fiber cabling to build Ethernet LANs. The chapter ends with some discussion of the rules for forwarding frames through an Ethernet LAN.

Building Physical Ethernet LANs with UTP

The next section of this chapter focuses on the individual physical links between any two Ethernet nodes, specifically those that use Unshielded Twisted Pair (UTP) cabling. Before the Ethernet network as a whole can send Ethernet frames between user devices, each node must be ready and able to send data over an individual physical link.

This section focuses on the three most commonly used Ethernet standards: 10BASE-T (Ethernet), 100BASE-T (Fast Ethernet, or FE), and 1000BASE-T (Gigabit Ethernet, or GE). Specifically, this section looks at the details of sending data in both directions over a UTP cable. It then examines the specific wiring of the UTP cables used for 10-Mbps, 100-Mbps, and 1000-Mbps Ethernet.

Transmitting Data Using Twisted Pairs

While it is true that Ethernet sends data over UTP cables, the physical means to send the data uses electricity that flows over the wires inside the UTP cable. To better understand how Ethernet sends data using electricity, break the idea down into two parts: how to create an electrical circuit and then how to make that electrical signal communicate 1s and 0s.

First, to create one electrical circuit, Ethernet defines how to use the two wires inside a single twisted pair of wires, as shown in Figure 2-5. The figure does not show a UTP cable between two nodes, but instead shows two individual wires that are inside the UTP cable. An electrical circuit requires a complete loop, so the two nodes, using circuitry on their Ethernet ports, connect the wires in one pair to complete a loop, allowing electricity to flow.

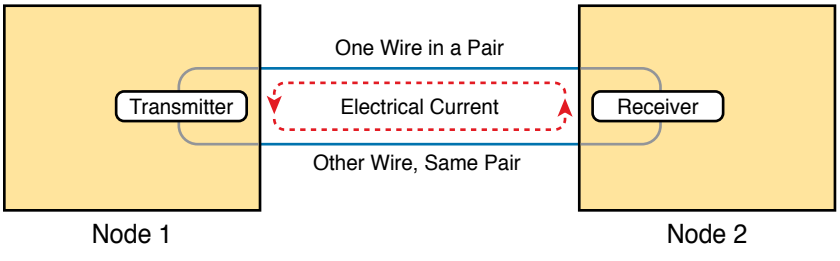


Figure 2-5 *Creating One Electrical Circuit over One Pair to Send in One Direction*

To send data, the two devices follow some rules called an *encoding scheme*. The idea works a lot like when two people talk using the same language: The speaker says some words in a particular language, and the listener, because she speaks the same language, can understand the spoken words. With an encoding scheme, the transmitting node changes the electrical signal over time, while the other node, the receiver, using the same rules, interprets those changes as either 0s or 1s. (For example, 10BASE-T uses an encoding scheme that encodes

a binary 0 as a transition from higher voltage to lower voltage during the middle of a 1/10,000,000th-of-a-second interval.)

Note that in an actual UTP cable, the wires will be twisted together, instead of being parallel as shown in Figure 2-5. The twisting helps solve some important physical transmission issues. When electrical current passes over any wire, it creates electromagnetic interference (EMI) that interferes with the electrical signals in nearby wires, including the wires in the same cable. (EMI between wire pairs in the same cable is called *crosstalk*.) Twisting the wire pairs together helps cancel out most of the EMI, so most networking physical links that use copper wires use twisted pairs.

Breaking Down a UTP Ethernet Link

The term *Ethernet link* refers to any physical cable between two Ethernet nodes. To learn about how a UTP Ethernet link works, it helps to break down the physical link into those basic pieces, as shown in Figure 2-6: the cable itself, the connectors on the ends of the cable, and the matching ports on the devices into which the connectors will be inserted.

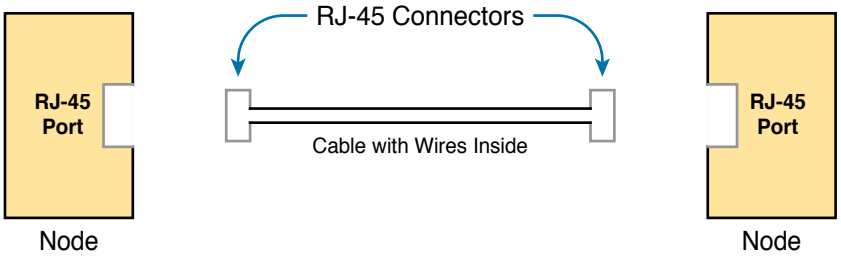


Figure 2-6 *Basic Components of an Ethernet Link*

First, think about the UTP cable itself. The cable holds some copper wires, grouped as twisted pairs. The 10BASE-T and 100BASE-T standards require two pairs of wires, while the 1000BASE-T standard requires four pairs. Each wire has a color-coded plastic coating, with the wires in a pair having a color scheme. For example, for the blue wire pair, one wire's coating is all blue, while the other wire's coating is blue-and-white striped.

Many Ethernet UTP cables use an RJ-45 connector on both ends. The RJ-45 connector has eight physical locations into which the eight wires in the cable can be inserted, called *pin positions*, or simply *pins*. These pins create a place where the ends of the copper wires can touch the electronics inside the nodes at the end of the physical link so that electricity can flow.

NOTE If available, find a nearby Ethernet UTP cable and examine the connectors closely. Look for the pin positions and the colors of the wires in the connector.

To complete the physical link, the nodes each need an RJ-45 *Ethernet port* that matches the RJ-45 connectors on the cable so that the connectors on the ends of the cable can connect to each node. PCs often include this RJ-45 Ethernet port as part of a network interface card (NIC), which can be an expansion card on the PC or can be built in to the system itself.

Switches typically have many RJ-45 ports because switches give user devices a place to connect to the Ethernet LAN. Figure 2-7 shows photos of the cables, connectors, and ports.

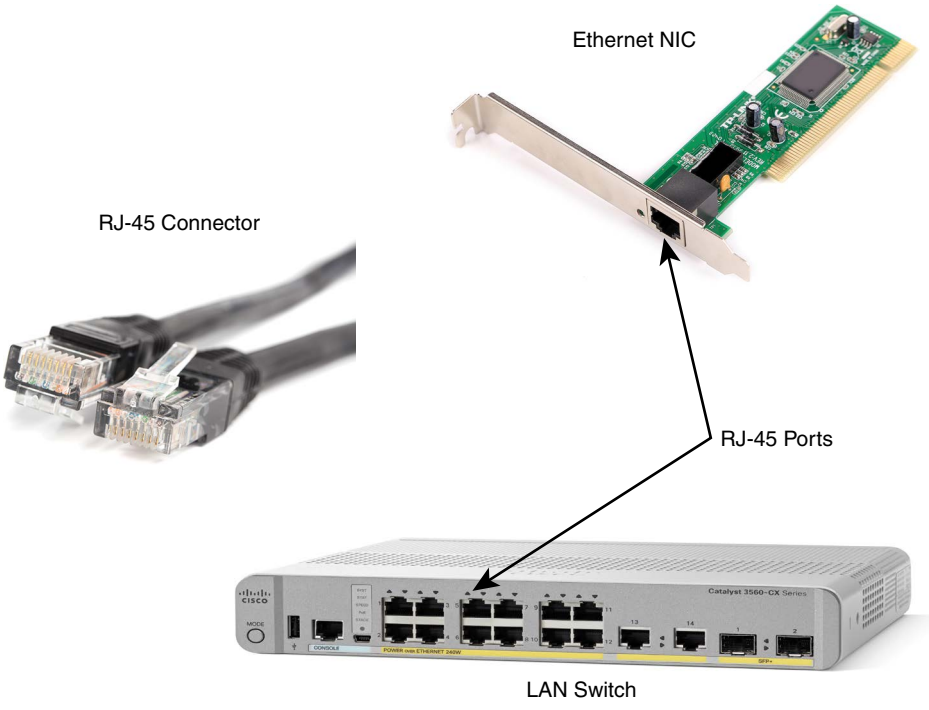


Figure 2-7 *RJ-45 Connectors and Ports (Ethernet NIC © Oleg Begunenko/123RF, RJ-45 Connector © Anton Samsonov/123RF)*

The figure shows a connector on the left and ports on the right. The left shows the eight pin positions in the end of the RJ-45 connector. The upper right shows an Ethernet NIC that is not yet installed in a computer. The lower-right part of the figure shows the side of a Cisco switch, with multiple RJ-45 ports, allowing multiple devices to easily connect to the Ethernet network.

Finally, while RJ-45 connectors with UTP cabling can be common, Cisco LAN switches often support other types of connectors as well. When you buy one of the many models of Cisco switches, you need to think about the mix and numbers of each type of physical ports you want on the switch.

To give its customers flexibility as to the type of Ethernet links, even after the customer has bought the switch, Cisco switches include some physical ports whose port hardware (the transceiver) can be changed later, after you purchase the switch.

For example, Figure 2-8 shows a photo of a Cisco switch with one of the swappable transceivers. In this case, the figure shows an enhanced small form-factor pluggable (SFP+) transceiver, which runs at 10 Gbps, just outside two SFP+ slots on a Cisco 3560CX switch. The SFP+ itself is the silver-colored part below the switch, with a black cable connected to it.

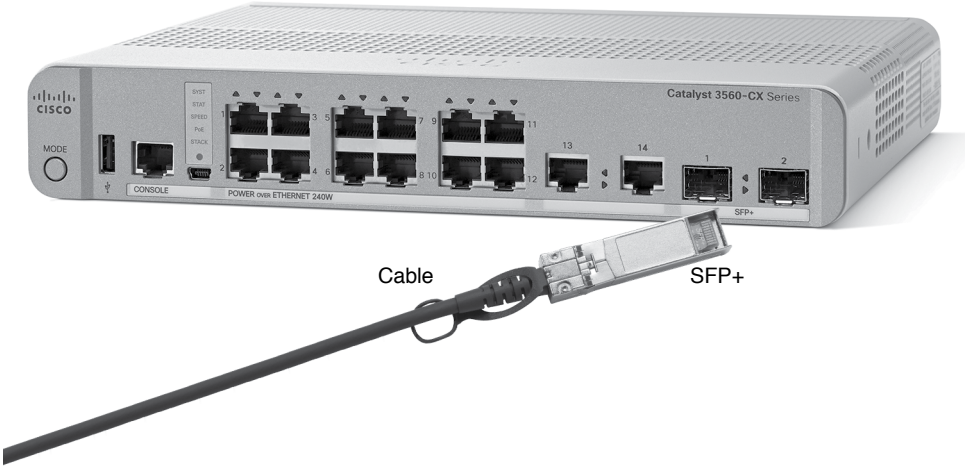


Figure 2-8 *10-Gbps SFP+ with Cable Sitting Just Outside a Catalyst 3560CX Switch*

Gigabit Ethernet Interface Converter (GBIC): The original form factor for a removable transceiver for Gigabit interfaces; larger than SFPs

Small Form Pluggable (SFP): The replacement for GBICs, used on Gigabit interfaces, with a smaller size, taking less space on the side of the networking card or switch.

Small Form Pluggable Plus (SFP+): Same size as the SFP, but used on 10-Gbps interfaces. (The Plus refers to the increase in speed compared to SFPs.)

UTP Cabling Pinouts for 10BASE-T and 100BASE-T

So far in this section, you have learned about the equivalent of how to drive a truck on a 1000-acre ranch: You could drive the truck all over the ranch, any place you wanted to go, and the police would not mind. However, as soon as you get on the public roads, the police want you to behave and follow the rules. Similarly, so far this chapter has discussed the general principles of how to send data, but it has not yet detailed some important rules for Ethernet cabling: the rules of the road so that all the devices send data using the right wires inside the cable.

This next topic discusses some of those rules, specifically for the 10-Mbps 10BASE-T and the 100-Mbps 100BASE-T. Both use UTP cabling in similar ways (including the use of only two wire pairs). A short comparison of the wiring for 1000BASE-T (Gigabit Ethernet), which uses four pairs, follows.

Straight-Through Cable Pinout

10BASE-T and 100BASE-T use two pairs of wires in a UTP cable, one for each direction, as shown in Figure 2-9. The figure shows four wires, all of which sit inside a single UTP cable that connects a PC and a LAN switch. In this example, the PC on the left transmits using the top pair, and the switch on the right transmits using the bottom pair.

Key Topic

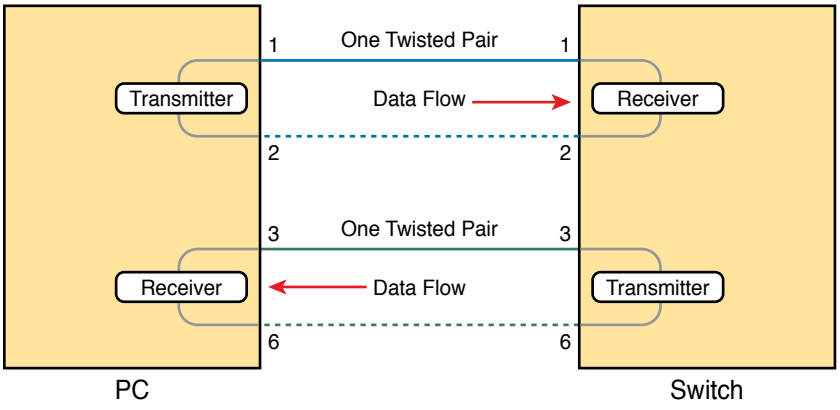


Figure 2-9 *Using One Pair for Each Transmission Direction with 10- and 100-Mbps Ethernet*

For correct transmission over the link, the wires in the UTP cable must be connected to the correct pin positions in the RJ-45 connectors. For example, in Figure 2-9, the transmitter on the PC on the left must know the pin positions of the two wires it should use to transmit. Those two wires must be connected to the correct pins in the RJ-45 connector on the switch so that the switch's receiver logic can use the correct wires.

To understand the wiring of the cable—which wires need to be in which pin positions on both ends of the cable—you need to first understand how the NICs and switches work. As a rule, Ethernet NIC transmitters use the pair connected to pins 1 and 2; the NIC receivers use a pair of wires at pin positions 3 and 6. LAN switches, knowing those facts about what Ethernet NICs do, do the opposite: Their receivers use the wire pair at pins 1 and 2, and their transmitters use the wire pair at pins 3 and 6.

To allow a PC NIC to communicate with a switch, the UTP cable must also use a *straight-through cable pinout*. The term *pinout* refers to the wiring of which color wire is placed in each of the eight numbered pin positions in the RJ-45 connector. An Ethernet straight-through cable connects the wire at pin 1 on one end of the cable to pin 1 at the other end of the cable; the wire at pin 2 needs to connect to pin 2 on the other end of the cable; pin 3 on one end connects to pin 3 on the other, and so on, as seen in Figure 2-10. Also, it uses the wires in one wire pair at pins 1 and 2, and another pair at pins 3 and 6.

Key Topic

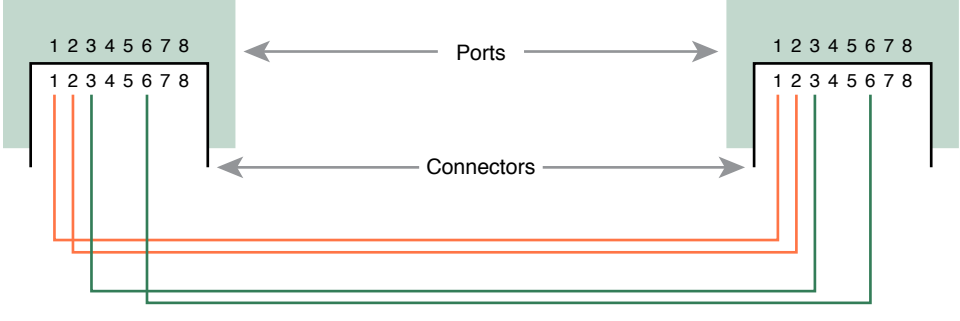


Figure 2-10 *10BASE-T and 100BASE-T Straight-Through Cable Pinout*

Figure 2-11 shows one final perspective on the straight-through cable pinout. In this case, PC Larry connects to a LAN switch. Note that the figure again does not show the UTP cable, but instead shows the wires that sit inside the cable, to emphasize the idea of wire pairs and pins.

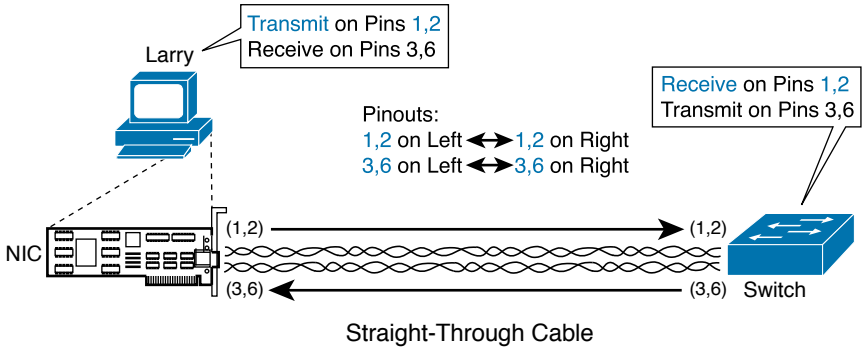


Figure 2-11 *Ethernet Straight-Through Cable Concept*

A straight-through cable works correctly when the nodes use opposite pairs for transmitting data. However, when two like devices connect to an Ethernet link, they both transmit on the same pins. In that case, you then need another type of cabling pinout called a *crossover cable*. The crossover cable pinout crosses the pair at the transmit pins on each device to the receive pins on the opposite device.

While that previous sentence is true, this concept is much clearer with a figure such as Figure 2-12. The figure shows what happens on a link between two switches. The two switches both transmit on the pair at pins 3 and 6, and they both receive on the pair at pins 1 and 2. So, the cable must connect a pair at pins 3 and 6 on each side to pins 1 and 2 on the other side, connecting to the other node's receiver logic. The top of the figure shows the literal pinouts, and the bottom half shows a conceptual diagram.

Key Topic

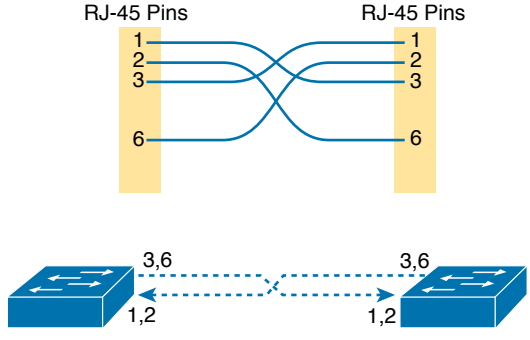


Figure 2-12 *Crossover Ethernet Cable*

Choosing the Right Cable Pinouts

For the exam, you should be well prepared to choose which type of cable (straight-through or crossover) is needed in each part of the network. The key is to know whether a device

acts like a PC NIC, transmitting at pins 1 and 2, or like a switch, transmitting at pins 3 and 6. Then, just apply the following logic:

Crossover cable: If the endpoints transmit on the same pin pair

Straight-through cable: If the endpoints transmit on different pin pairs

Table 2-3 lists the devices and the pin pairs they use, assuming that they use 10BASE-T and 100BASE-T.

Key Topic

Table 2-3 10BASE-T and 100BASE-T Pin Pairs Used

Transmits on Pins 1,2	Transmits on Pins 3,6
PC NICs	Hubs
Routers	Switches
Wireless access point (Ethernet interface)	—

For example, Figure 2-13 shows a campus LAN in a single building. In this case, several straight-through cables are used to connect PCs to switches. In addition, the cables connecting the switches require crossover cables.

Key Topic

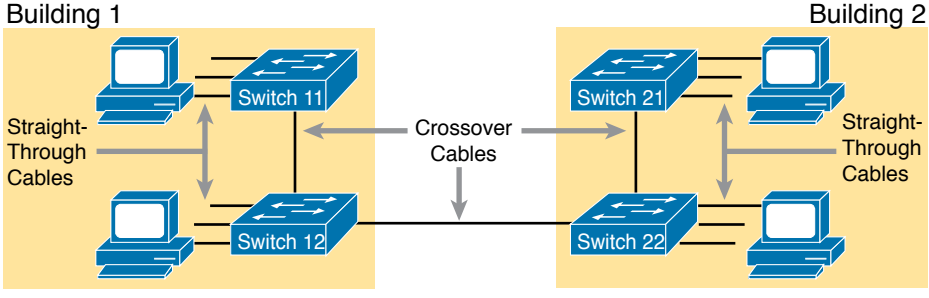


Figure 2-13 *Typical Uses for Straight-Through and Crossover Ethernet Cables*

NOTE If you have some experience with installing LANs, you might be thinking that you have used the wrong cable before (straight-through or crossover), but the cable worked. Cisco switches have a feature called *auto-mdix* that notices when the wrong cable is used and automatically changes its logic to make the link work. However, for the exams, be ready to identify whether the correct cable is shown in the figures.

UTP Cabling Pinouts for 1000BASE-T

1000BASE-T (Gigabit Ethernet) differs from 10BASE-T and 100BASE-T as far as the cabling and pinouts. First, 1000BASE-T requires four wire pairs. Second, it uses more advanced electronics that allow both ends to transmit and receive simultaneously on each wire pair. However, the wiring pinouts for 1000BASE-T work almost identically to the earlier standards, adding details for the additional two pairs.

The straight-through cable for 1000BASE-T uses the four wire pairs to create four circuits, but the pins need to match. It uses the same pinouts for two pairs as do the 10BASE-T and

100BASE-T standards, and it adds a pair at pins 4 and 5 and the final pair at pins 7 and 8, as shown in Figure 2-14.

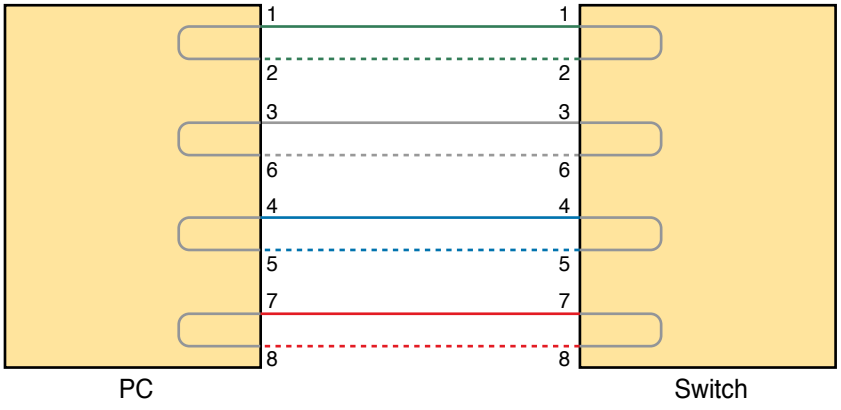


Figure 2-14 *Four-Pair Straight-Through Cable to 1000BASE-T*

The Gigabit Ethernet crossover cable crosses the same two-wire pairs as the crossover cable for the other types of Ethernet (the pairs at pins 1,2 and 3,6). It also crosses the two new pairs as well (the pair at pins 4,5 with the pair at pins 7,8).

Building Physical Ethernet LANs with Fiber

The capability of many UTP-based Ethernet standards to use a cable length up to 100 meters means that the majority of Ethernet cabling in an enterprise uses UTP cables. The distance from an Ethernet switch to every endpoint on the floor of a building will likely be less than 100m. In some cases, however, an engineer might prefer to use fiber cabling for some links in an Ethernet LAN, first to reach greater distances, but for other reasons as well. This next section examines a few of the tradeoffs after discussing the basics of how to transmit data over fiber cabling.

Fiber Cabling Transmission Concepts

Fiber-optic cabling uses glass as the medium through which light passes, varying that light over time to encode 0s and 1s. It might seem strange at first to use glass given that most of us think of glass in windows. Window glass is hard, unbending, and if you hit or bend it enough, the glass will probably shatter—all bad characteristics for a cabling material.

Instead, fiber-optic cables use fiberglass, which allows a manufacturer to spin a long thin string (fiber) of flexible glass. A fiber-optic cable holds the fiber in the middle of the cable, allowing the light to pass through the glass—which is a very important attribute for the purposes of sending data.

Although sending data through a glass fiber works well, the glass fiber by itself needs some help. The glass could break, so the glass fiber needs some protection and strengthening. Figure 2-15 shows a cutout with the components of a fiber cable for perspective.

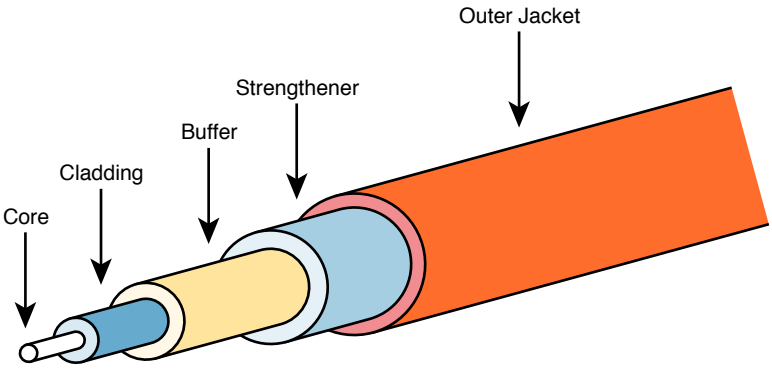


Figure 2-15 *Components of a Fiber-Optic Cable*

The three outer layers of the cable protect the interior of the cable and make the cables easier to install and manage, while the inner *cladding* and *core* work together to create the environment to allow transmission of light over the cable. A light source, called the *optical transmitter*, shines a light into the core. Light can pass through the core; however, light reflects off the cladding back into the core. Figure 2-16 shows an example with a light emitting diode (LED) transmitter. You can see how the cladding reflects the light back into the core as it travels through the core.

Key Topic

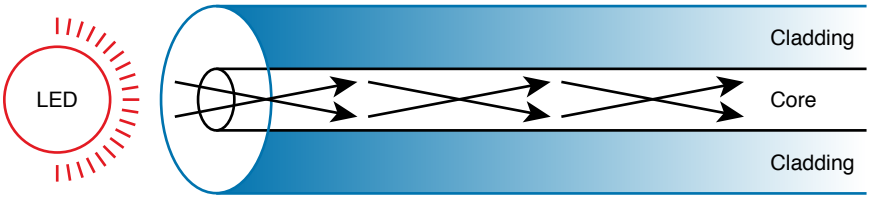


Figure 2-16 *Transmission on Multimode Fiber with Internal Reflection*

The figure shows the normal operation of a multimode fiber, characterized by the fact that the cable allows for multiple angles (modes) of light waves entering the core.

In contrast, single-mode fiber uses a smaller-diameter core, around one-fifth the diameter of common multimode cables (see Figure 2-17). To transmit light into a much smaller core, a laser-based transmitter sends light at a single angle (hence the name *single-mode*).

Figure 2-17 *Transmission on Single-Mode Fiber with Laser Transmitter*

Both multimode and single-mode cabling have important roles in Ethernet and meet different needs. Multimode improves the maximum distances over UTP, and it uses less expensive

transmitters as compared with single-mode. Standards do vary; for instance, the standards for 10 Gigabit Ethernet over Fiber allow for distances up to 400m, which would often allow for connection of devices in different buildings in the same office park. Single-mode allows distances into the tens of kilometers, but with slightly more expensive SFP/SFP+ hardware.

To transmit between two devices, you need two cables, one for each direction, as shown in Figure 2-18. The concept works much like having two electrical circuits with the original UTP Ethernet standards. Note that the transmit port on one device connects to a cable that connects to a receive port on the other device, and vice versa with the other cable.

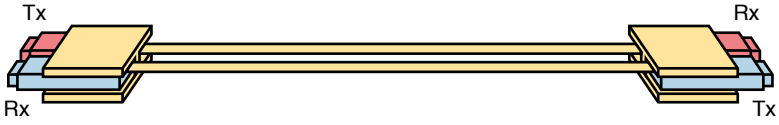


Figure 2-18 *Two Fiber Cables with Tx Connected to Rx on Each Cable*

Using Fiber with Ethernet

To use fiber with Ethernet switches, you need to use a switch with either built-in ports that support a particular optical Ethernet standard, or a switch with modular ports that allow you to change the Ethernet standard used on the port. Refer back to Figure 2-8, which shows a photo of a switch with two SFP+ ports, into which you could insert any of the supported SFP+ modules. Those SFP+ ports support a variety of 10-Gbps standards like those listed in Table 2-4.

Table 2-4 A Sampling of IEEE 802.3 10-Gbps Fiber Standards

Standard	Cable Type	Max Distance*
10GBASE-S	MM	400m
10GBASE-LX4	MM	300m
10GBASE-LR	SM	10km
10GBASE-E	SM	30km

* The maximum distances are based on the IEEE standards with no repeaters.

For instance, to build an Ethernet LAN in an office park, you might need to use some multimode and single-mode fiber links. In fact, many office parks might already have fiber cabling installed for the expected future use by the tenants in the buildings. If each building was within a few hundred meters of at least one other building, you could use multimode fiber between the buildings and connect switches to create your LAN.

NOTE Outside the need to study for CCNA, if you need to look more deeply at fiber Ethernet and SFP/SFP+, check out tmgmatrix.cisco.com as a place to search for and learn about compatible SFP/SFP+ hardware from Cisco.

Although distance might be the first criterion to consider when thinking about whether to use UTP or fiber cabling, a few other tradeoffs exist as well. UTP wins again on cost,

because the cost goes up as you move from UTP, to multimode, and then to single-mode, due to the extra cost for the transmitters like the SFP and SFP+ modules. UTP has some negatives, however. First, UTP might work poorly in some electrically noisy environments such as factories, because UTP can be affected by electromagnetic interference (EMI). Also, UTP cables emit a faint signal outside the cable, so highly secure networks may choose to use fiber, which does not create similar emissions, to make the network more secure. Table 2-5 summarizes these tradeoffs.

Key Topic

Table 2-5 Comparisons Between UTP, MM, and SM Ethernet Cabling

Criteria	UTP	Multimode	Single-Mode
Relative Cost of Cabling	Low	Medium	Medium
Relative Cost of a Switch Port	Low	Medium	High
Approximate Max Distance	100m	500m	40km
Relative Susceptibility to Interference	Some	None	None
Relative Risk of Copying from Cable Emissions	Some	None	None

Sending Data in Ethernet Networks

Although physical layer standards vary quite a bit, other parts of the Ethernet standards work the same regardless of the type of physical Ethernet link. Next, this final major section of this chapter looks at several protocols and rules that Ethernet uses regardless of the type of link. In particular, this section examines the details of the Ethernet data-link layer protocol, plus how Ethernet nodes, switches, and hubs forward Ethernet frames through an Ethernet LAN.

Ethernet Data-Link Protocols

One of the most significant strengths of the Ethernet family of protocols is that these protocols use the same data-link standard. In fact, the core parts of the data-link standard date back to the original Ethernet standards.

The Ethernet data-link protocol defines the Ethernet frame: an Ethernet header at the front, the encapsulated data in the middle, and an Ethernet trailer at the end. Ethernet actually defines a few alternate formats for the header, with the frame format shown in Figure 2-19 being commonly used today.

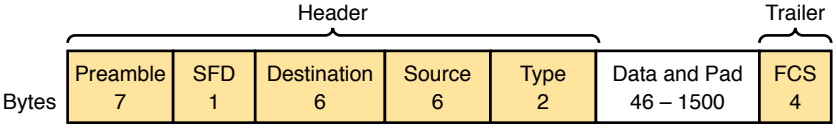


Figure 2-19 *Commonly Used Ethernet Frame Format*

While all the fields in the frame matter, some matter more to the topics discussed in this book. Table 2-6 lists the fields in the header and trailer and a brief description for reference, with the upcoming pages including more detail about a few of these fields.

Table 2-6 IEEE 802.3 Ethernet Header and Trailer Fields

Field	Bytes	Description
Preamble	7	Synchronization.
Start Frame Delimiter (SFD)	1	Signifies that the next byte begins the Destination MAC Address field.
Destination MAC Address	6	Identifies the intended recipient of this frame.
Source MAC Address	6	Identifies the sender of this frame.
Type	2	Defines the type of protocol listed inside the frame; today, most likely identifies IP version 4 (IPv4) or IP version 6 (IPv6).
Data and Pad*	46– 1500	Holds data from a higher layer, typically an L3PDU (usually an IPv4 or IPv6 packet). The sender adds padding to meet the minimum length requirement for this field (46 bytes).
Frame Check Sequence (FCS)	4	Provides a method for the receiving NIC to determine whether the frame experienced transmission errors.

* The IEEE 802.3 specification limits the data portion of the 802.3 frame to a minimum of 46 and a maximum of 1500 bytes. The *term maximum transmission unit* (MTU) defines the maximum Layer 3 packet that can be sent over a medium. Because the Layer 3 packet rests inside the data portion of an Ethernet frame, 1500 bytes is the largest IP MTU allowed over an Ethernet.

Ethernet Addressing

The source and destination Ethernet address fields play a huge role in how Ethernet LANs work. The general idea for each is relatively simple: the sending node puts its own address in the source address field and the intended Ethernet destination device's address in the destination address field. The sender transmits the frame, expecting that the Ethernet LAN, as a whole, will deliver the frame to that correct destination.

Ethernet addresses, also called Media Access Control (MAC) addresses, are 6-byte-long (48-bit-long) binary numbers. For convenience, most computers list MAC addresses as 12-digit hexadecimal numbers. Cisco devices typically add some periods to the number for easier readability as well; for example, a Cisco switch might list a MAC address as 0000.0C12.3456.

Most MAC addresses represent a single NIC or other Ethernet port, so these addresses are often called a *unicast* Ethernet address. The term *unicast* is simply a formal way to refer to the fact that the address represents one interface to the Ethernet LAN. (This term also contrasts with two other types of Ethernet addresses, *broadcast* and *multicast*, which will be defined later in this section.)

The entire idea of sending data to a destination unicast MAC address works well, but it works only if all the unicast MAC addresses are unique. If two NICs tried to use the same MAC address, there could be confusion. (The problem would be like the confusion caused to the postal service if you and I both tried to use the same mailing address—would the postal service deliver mail to your house or mine?) If two PCs on the same Ethernet tried to use the same MAC address, to which PC should frames sent to that MAC address be delivered?

Ethernet solves this problem using an administrative process so that, at the time of manufacture, all Ethernet devices are assigned a universally unique MAC address. Before a manufacturer can build Ethernet products, it must ask the IEEE to assign the manufacturer a universally unique 3-byte code, called the organizationally unique identifier (OUI). The manufacturer agrees to give all NICs (and other Ethernet products) a MAC address that begins with its assigned 3-byte OUI. The manufacturer also assigns a unique value for the last 3 bytes, a number that manufacturer has never used with that OUI. As a result, the MAC address of every device in the universe is unique.

NOTE The IEEE also calls these universal MAC addresses global MAC addresses.

Figure 2-20 shows the structure of the unicast MAC address, with the OUI.

Key Topic

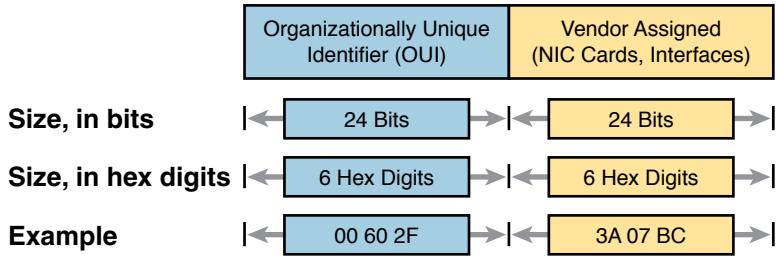


Figure 2-20 *Structure of Unicast Ethernet Addresses*

Ethernet addresses go by many names: LAN address, Ethernet address, hardware address, burned-in address, physical address, universal address, or MAC address. For example, the term burned-in address (BIA) refers to the idea that a permanent MAC address has been encoded (burned into) the ROM chip on the NIC. As another example, the IEEE uses the term *universal address* to emphasize the fact that the address assigned to a NIC by a manufacturer should be unique among all MAC addresses in the universe.

In addition to unicast addresses, Ethernet also uses group addresses. *Group addresses* identify more than one LAN interface card. A frame sent to a group address might be delivered to a small set of devices on the LAN, or even to all devices on the LAN. In fact, the IEEE defines two general categories of group addresses for Ethernet:

Broadcast address: Frames sent to this address should be delivered to all devices on the Ethernet LAN. It has a value of FFFF.FFFF.FFFF.

Multicast addresses: Frames sent to a multicast Ethernet address will be copied and forwarded to a subset of the devices on the LAN that volunteers to receive frames sent to a specific multicast address.

Table 2-7 summarizes most of the details about MAC addresses.

Table 2-7 LAN MAC Address Terminology and Features

LAN Addressing Term or Feature	Description
MAC	Media Access Control. 802.3 (Ethernet) defines the MAC sublayer of IEEE Ethernet.
Ethernet address, NIC address, LAN address	Other names often used instead of MAC address. These terms describe the 6-byte address of the LAN interface card.
Burned-in address	The 6-byte address assigned by the vendor making the card.
Unicast address	A term for a MAC address that represents a single LAN interface.
Broadcast address	An address that means "all devices that reside on this LAN right now."
Multicast address	On Ethernet, a multicast address implies some subset of all devices currently on the Ethernet LAN.

Identifying Network Layer Protocols with the Ethernet Type Field

While the Ethernet header's address fields play an important and more obvious role in Ethernet LANs, the Ethernet Type field plays a much less obvious role. The Ethernet Type field, or EtherType, sits in the Ethernet data-link layer header, but its purpose is to directly help the network processing on routers and hosts. Basically, the Type field identifies the type of network layer (Layer 3) packet that sits inside the Ethernet frame.

First, think about what sits inside the data part of the Ethernet frame shown earlier in Figure 2-14. Typically, it holds the network layer packet created by the network layer protocol on some device in the network. Over the years, those protocols have included IBM Systems Network Architecture (SNA), Novell NetWare, Digital Equipment Corporation's DECnet, and Apple Computer's AppleTalk. Today, the most common network layer protocols are both from TCP/IP: IP version 4 (IPv4) and IP version 6 (IPv6).

The original host has a place to insert a value (a hexadecimal number) to identify the type of packet encapsulated inside the Ethernet frame. However, what number should the sender put in the header to identify an IPv4 packet as the type? Or an IPv6 packet? As it turns out, the IEEE manages a list of EtherType values, so that every network layer protocol that needs a unique EtherType value can have a number. The sender just has to know the list. (Anyone can view the list; just go to www.ieee.org and search for *EtherType*.)

For example, a host can send one Ethernet frame with an IPv4 packet and the next Ethernet frame with an IPv6 packet. Each frame would have a different Ethernet Type field value, using the values reserved by the IEEE, as shown in Figure 2-21.

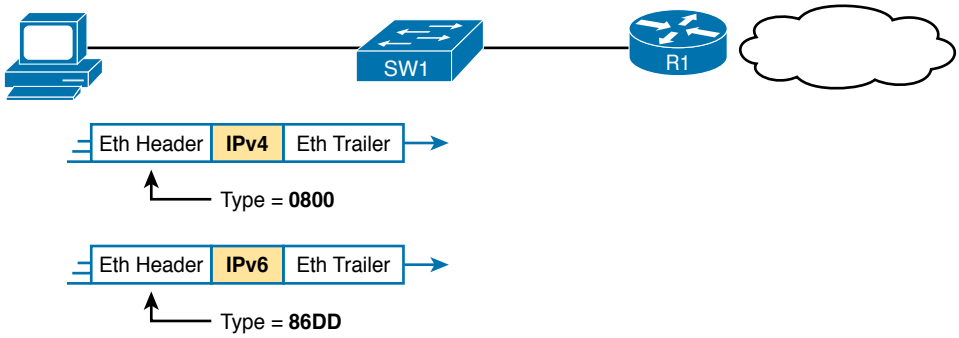


Figure 2-21 *Use of Ethernet Type Field*

Error Detection with FCS

Ethernet also defines a way for nodes to find out whether a frame's bits changed while crossing over an Ethernet link. (Usually, the bits could change because of some kind of electrical interference, or a bad NIC.) Ethernet, like most data-link protocols, uses a field in the data-link trailer for the purpose of error detection.

The Ethernet Frame Check Sequence (FCS) field in the Ethernet trailer—the only field in the Ethernet trailer—gives the receiving node a way to compare results with the sender, to discover whether errors occurred in the frame. The sender applies a complex math formula to the frame before sending it, storing the result of the formula in the FCS field. The receiver applies the same math formula to the received frame. The receiver then compares its own results with the sender's results. If the results are the same, the frame did not change; otherwise, an error occurred, and the receiver discards the frame.

Note that *error detection* does not also mean *error recovery*. Ethernet defines that the errored frame should be discarded, but Ethernet does not attempt to recover the lost frame. Other protocols, notably TCP, recover the lost data by noticing that it is lost and sending the data again.

Sending Ethernet Frames with Switches and Hubs

Ethernet LANs behave slightly differently depending on whether the LAN has mostly modern devices, in particular, LAN switches instead of some older LAN devices called LAN hubs. Basically, the use of more modern switches allows the use of full-duplex logic, which is much faster and simpler than half-duplex logic, which is required when using hubs. The final topic in this chapter looks at these basic differences.

Sending in Modern Ethernet LANs Using Full Duplex

Modern Ethernet LANs use a variety of Ethernet physical standards, but with standard Ethernet frames that can flow over any of these types of physical links. Each individual link can run at a different speed, but each link allows the attached nodes to send the bits in the frame to the next node. They must work together to deliver the data from the sending Ethernet node to the destination node.

The process is relatively simple, on purpose; the simplicity lets each device send a large number of frames per second. Figure 2-22 shows an example in which PC1 sends an Ethernet frame to PC2.

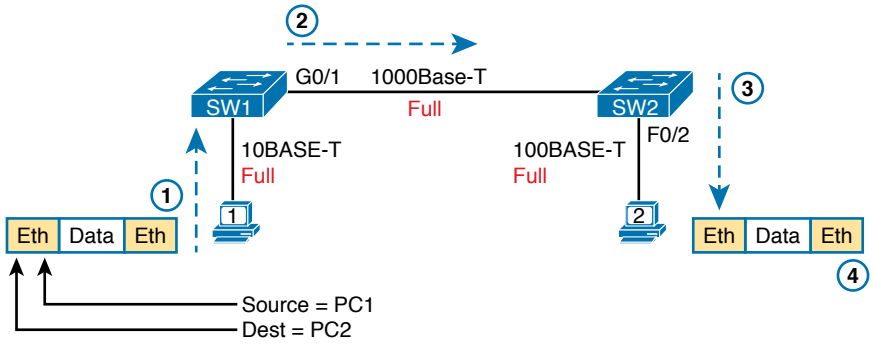


Figure 2-22 *Example of Sending Data in a Modern Ethernet LAN*

Following the steps in the figure:

1. PC1 builds and sends the original Ethernet frame, using its own MAC address as the source address and PC2's MAC address as the destination address.
2. Switch SW1 receives and forwards the Ethernet frame out its G0/1 interface (short for Gigabit interface 0/1) to SW2.
3. Switch SW2 receives and forwards the Ethernet frame out its F0/2 interface (short for Fast Ethernet interface 0/2) to PC2.
4. PC2 receives the frame, recognizes the destination MAC address as its own, and processes the frame.

The Ethernet network in Figure 2-22 uses full duplex on each link, but the concept might be difficult to see.

Full duplex means that that the NIC or switch port has no half-duplex restrictions. So, to understand full duplex, you need to understand half duplex, as follows:

Key Topic

Half duplex: The device must wait to send if it is currently receiving a frame; in other words, it cannot send and receive at the same time.

Full duplex: The device does not have to wait before sending; it can send and receive at the same time.

So, with all PCs and LAN switches, and no LAN hubs, all the nodes can use full duplex. All nodes can send and receive on their port at the same instant in time. For example, in Figure 2-22, PC1 and PC2 could send frames to each other simultaneously, in both directions, without any half-duplex restrictions.

Using Half Duplex with LAN Hubs

To understand the need for half-duplex logic in some cases, you have to understand a little about an older type of networking device called a LAN hub. When the IEEE first introduced 10BASE-T in 1990, Ethernet switches did not exist yet; instead, networks used a device called a LAN hub. Like a switch, a LAN hub provided a number of RJ-45 ports as a place to connect links to PCs; however, hubs used different rules for forwarding data.

LAN hubs forward data using physical layer standards rather than data-link standards and are therefore considered to be Layer 1 devices. When an electrical signal comes in one hub port, the hub repeats that electrical signal out all other ports (except the incoming port). By doing

so, the data reaches all the rest of the nodes connected to the hub, so the data hopefully reaches the correct destination. The hub has no concept of Ethernet frames, of addresses, making decisions based on those addresses, and so on.

The downside of using LAN hubs is that if two or more devices transmitted a signal at the same instant, the electrical signal collides and becomes garbled. The hub repeats all received electrical signals, even if it receives multiple signals at the same time. For example, Figure 2-23 shows the idea, with PCs Archie and Bob sending an electrical signal at the same instant of time (at Steps 1A and 1B) and the hub repeating both electrical signals out toward Larry on the left (Step 2).

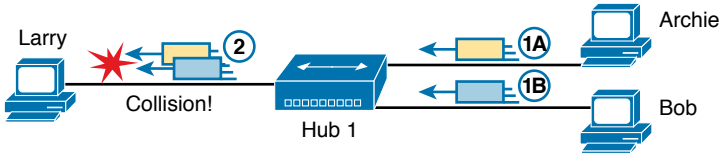


Figure 2-23 *Collision Occurring Because of LAN Hub Behavior*

NOTE For completeness, note that the hub floods each frame out all other ports (except the incoming port). So, Archie's frame goes to both Larry and Bob; Bob's frame goes to Larry and Archie.

If you replace the hub in Figure 2-23 with a LAN switch, the switch prevents the collision on the left. The switch operates as a Layer 2 device, meaning that it looks at the data-link header and trailer. A switch would look at the MAC addresses, and even if the switch needed to forward both frames to Larry on the left, the switch would send one frame and queue the other frame until the first frame was finished.

Now back to the issue created by the hub's logic: collisions. To prevent these collisions, the Ethernet nodes must use half-duplex logic instead of full-duplex logic. A problem occurs only when two or more devices send at the same time; half-duplex logic tells the nodes that if someone else is sending, wait before sending.

For example, back in Figure 2-23, imagine that Archie began sending his frame early enough so that Bob received the first bits of that frame before Bob tried to send his own frame. Bob, at Step 1B, would notice that he was receiving a frame from someone else, and using half-duplex logic, would simply wait to send the frame listed at Step 1B.

Nodes that use half-duplex logic actually use a relatively well-known algorithm called carrier sense multiple access with collision detection (CSMA/CD). The algorithm takes care of the obvious cases but also the cases caused by unfortunate timing. For example, two nodes could check for an incoming frame at the exact same instant, both realize that no other node is sending, and both send their frames at the exact same instant, causing a collision. CSMA/CD covers these cases as well, as follows:

Step 1. A device with a frame to send listens until the Ethernet is not busy.

Step 2. When the Ethernet is not busy, the sender begins sending the frame.

Step 3. The sender listens while sending to discover whether a collision occurs; collisions might be caused by many reasons, including unfortunate timing. If a collision occurs, all currently sending nodes do the following:

A. They send a jamming signal that tells all nodes that a collision happened.

B. They independently choose a random time to wait before trying again, to avoid unfortunate timing.

C. The next attempt starts again at Step 1.

Although most modern LANs do not often use hubs and therefore do not need to use half duplex, enough old hubs still exist in enterprise networks so that you need to be ready to understand duplex issues. Each NIC and switch port has a duplex setting. For all links between PCs and switches, or between switches, use full duplex. However, for any link connected to a LAN hub, the connected LAN switch and NIC port should use half duplex. Note that the hub itself does not use half-duplex logic, instead just repeating incoming signals out every other port.

Figure 2-24 shows an example, with full-duplex links on the left and a single LAN hub on the right. The hub then requires SW2's F0/2 interface to use half-duplex logic, along with the PCs connected to the hub.

Key Topic

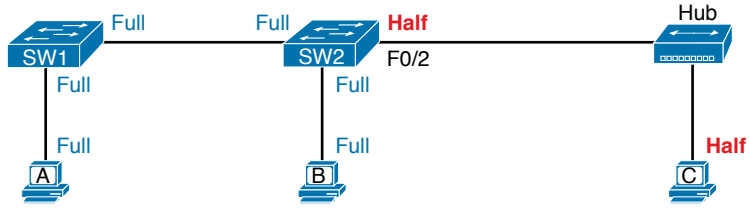


Figure 2-24 *Full and Half Duplex in an Ethernet LAN*

Before closing the chapter, note that the discussion of full and half duplex connects to two specific terms from CCNA exam topic 1.3.b, but those connections may not be obvious. First, the term *Ethernet shared media* (from the exam topic) refers to designs that use hubs, require CSMA/CD, and therefore share the bandwidth. The idea behind the term comes from the fact that the devices connected to the hub share the network because they must use CSMA/CD, and CSMA/CD enforces rules that allow only one device to successfully send a frame at any point in time.

By contrast, the term *Ethernet point-to-point* in that same exam topic emphasizes the fact that in a network built with switches, each (point-to-point) link works independently of the others. Because of the full-duplex logic discussed in this section, a frame can be sent on every point-to-point link in an Ethernet at the same time.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for

more details. Table 2-8 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 2-8 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website

Review All the Key Topics

Key Topic

Table 2-9 Key Topics for Chapter 2

Key Topic Element	Description	Page Number
Figure 2-3	Drawing of a typical wired and wireless enterprise LAN	36
Table 2-2	Several types of Ethernet LANs and some details about each	37
Figure 2-9	Conceptual drawing of transmitting in one direction each over two different electrical circuits between two Ethernet nodes	43
Figure 2-10	10- and 100-Mbps Ethernet straight-through cable pinouts	43
Figure 2-12	10- and 100-Mbps Ethernet crossover cable pinouts	44
Table 2-3	List of devices that transmit on wire pair 1,2 and pair 3,6	45
Figure 2-13	Typical uses for straight-through and crossover Ethernet cables	45
Figure 2-16	Physical transmission concepts in a multimode cable	47
Table 2-5	Comparison between UTP, MM, and SM Ethernet Cabling	49
Figure 2-20	Format of Ethernet MAC addresses	51
List	Definitions of half duplex and full duplex	54
Figure 2-24	Examples of which interfaces use full duplex and which interfaces use half duplex	56

Key Terms You Should Know

Ethernet, IEEE, wired LAN, wireless LAN, Ethernet frame, 10BASE-T, 100BASE-T, 1000BASE-T, Fast Ethernet, Gigabit Ethernet, Ethernet link, RJ-45, Ethernet port, network interface card (NIC), straight-through cable, crossover cable, Ethernet address, MAC address, unicast address, broadcast address, Frame Check Sequence, transceiver, Multimode (MM), single-mode (SM), electromagnetic Interference (EMI), core, cladding, fiber-optic cable

CHAPTER 3

Fundamentals of WANs and IP Routing

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.a Routers

1.2 Describe characteristics of network topology architectures

1.2.d WAN

This chapter introduces WANs and the various features of the TCP/IP network layer.

First, for WANs, note that the current CCNA blueprint does not examine WANs in detail as an end to themselves. However, to understand IP routing, you need to understand the basics of the two types of WAN links introduced in the first major section of this chapter: serial links and Ethernet WAN links. In their most basic form, these WAN links connect routers that sit at sites that can be miles to hundreds of miles apart, allowing communications between remote sites.

The rest of the chapter then turns to the TCP/IP Network layer, with IP as the center of the discussion. The second section of the chapter discusses the major features of IP: routing, addressing, and routing protocols. The final section of the chapter examines a few protocols other than IP that also help the TCP/IP Network layer create a network that allows end-to-end communication between endpoints.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 3-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Wide-Area Networks	1, 2
IP Routing	3–6
Other Network Layer Functions	7

1. Which of the following fields in the HDLC header used by Cisco routers does Cisco add, beyond the ISO standard HDLC?
   a. Flag
   b. Type
   c. Address
   d. FCS
2. Two routers, R1 and R2, connect using an Ethernet over MPLS service. The service provides point-to-point service between these two routers only, as a Layer 2 Ethernet service. Which of the following are the most likely to be true about this WAN? (Choose two answers.)
   a. R1 will connect to a physical Ethernet link, with the other end of the cable connected to R2.
   b. R1 will connect to a physical Ethernet link, with the other end of the cable connected to a device at the WAN service provider point of presence.
   c. R1 will forward data-link frames to R2 using an HDLC header/trailer.
   d. R1 will forward data-link frames to R2 using an Ethernet header/trailer.
3. Imagine a network with two routers that are connected with a point-to-point HDLC serial link. Each router has an Ethernet, with PC1 sharing the Ethernet with Router1 and PC2 sharing the Ethernet with Router2. When PC1 sends data to PC2, which of the following is true?
   a. Router1 strips the Ethernet header and trailer off the frame received from PC1, never to be used again.
   b. Router1 encapsulates the Ethernet frame inside an HDLC header and sends the frame to Router2, which extracts the Ethernet frame for forwarding to PC2.
   c. Router1 strips the Ethernet header and trailer off the frame received from PC1, which is exactly re-created by Router2 before forwarding data to PC2.
   d. Router1 removes the Ethernet, IP, and TCP headers and rebuilds the appropriate headers before forwarding the packet to Router2.
4. Which of the following does a router normally use when making a decision about routing TCP/IP packets?
   a. Destination MAC address
   b. Source MAC address
   c. Destination IP address
   d. Source IP address
   e. Destination MAC and IP addresses

5. Which of the following are true about a LAN-connected TCP/IP host and its IP routing (forwarding) choices?
   a. The host always sends packets to its default gateway.
   b. The host never sends packets to its default gateway.
   c. The host sends packets to its default gateway if the destination IP address is in a different subnet than the host.
   d. The host sends packets to its default gateway if the destination IP address is in the same subnet as the host.
6. Which of the following are functions of a routing protocol? (Choose two answers.)
   a. Advertising known routes to neighboring routers
   b. Learning routes for subnets directly connected to the router
   c. Learning routes and putting those routes into the routing table for routes advertised to the router by its neighboring routers
   d. Forwarding IP packets based on a packet's destination IP address
7. A company implements a TCP/IP network, with PC1 sitting on an Ethernet LAN. Which of the following protocols and features requires PC1 to learn information from some other server device?
   a. ARP
   b. ping
   c. DNS
   d. None of these answers is correct.

Foundation Topics

Wide-Area Networks

Imagine a typical day at the branch office at some enterprise. The user sits at some endpoint device: a PC, tablet, phone, and so on. It connects to a LAN, either via an Ethernet cable or using a wireless LAN. However, the user happens to be checking information on a website, and that web server sits at the home office of the company. To make that work, the data travels over one or more wide-area network (WAN) links.

WAN technologies define the physical (Layer 1) standards and data-link (Layer 2) protocols used to communicate long distances. This first section examines two such technologies: leased-line WANs and Ethernet WANs. Leased-line WANs have been an option for networks for half a century, are becoming much less common today, but you may still see some leased-line WAN links in the exam. Ethernet WAN links do use the same data-link protocols as Ethernet LANs, but they use additional features to make the links work over the much longer distances required for WANs. The next few pages examine leased-line WANs first, followed by Ethernet WANs.

Leased-Line WANs

To connect LANs using a WAN, the internetwork uses a router connected to each LAN, with a WAN link between the routers. First, the enterprise's network engineer would order some kind of WAN link. A router at each site connects to both the WAN link and the LAN, as shown in Figure 3-1. Note that a crooked line between the routers is the common way to represent a leased line when the drawing does not need to show any of the physical details of the line.

Figure 3-1 *Small Enterprise Network with One Leased Line*

This section begins by examining the physical details of leased lines, followed by a discussion of the default data-link protocol for leased lines (HDLC).

Physical Details of Leased Lines

The leased line service delivers bits in both directions, at a predetermined speed, using full-duplex logic. In fact, conceptually it acts as if you had a full-duplex crossover Ethernet link between two routers, as shown in Figure 3-2. The leased line uses two pairs of wires, one pair for each direction of sending data, which allows full-duplex operation.

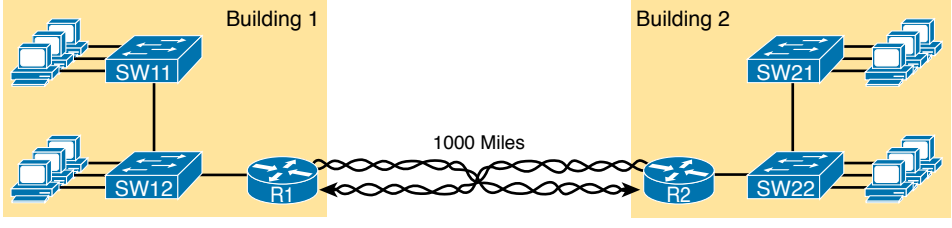


Figure 3-2 *Conceptual View of the Leased-Line Service*

Of course, leased lines have many differences compared to an Ethernet crossover cable. To create such possibly long links, or circuits, a leased line does not actually exist as a single long cable between the two sites. Instead, the telephone company (telco) that creates the leased line installs a large network of cables and specialized switching devices to create its own computer network. The telco network creates a service that acts like a crossover cable between two points, but the physical reality is hidden from the customer.

Leased lines come with their own set of terminology as well. First, the term *leased line* refers to the fact that the company using the leased line does not own the line but instead pays a monthly lease fee to use it. Table 3-2 lists some of the many names for leased lines, mainly so that in a networking job, you have a chance to translate from the terms each person uses with a basic description as to the meaning of the name.

Table 3-2 Different Names for a Leased Line

Name	Meaning or Reference
Leased circuit, Circuit	The words *line* and *circuit* are often used as synonyms in telco terminology; *circuit* makes reference to the electrical circuit between the two endpoints.
Serial link, Serial line	The words *link* and *line* are also often used as synonyms. *Serial* in this case refers to the fact that the bits flow serially and that routers use serial interfaces.
Point-to-point link, Point-to-point line	These terms refer to the fact that the topology stretches between two points, and two points only. (Some older leased lines allowed more than two devices.)
T1	This specific type of leased line transmits data at 1.544 megabits per second (1.544 Mbps).
WAN link, Link	Both of these terms are very general, with no reference to any specific technology.
Private line	This term refers to the fact that the data sent over the line cannot be copied by other telco customers, so the data is private.

To create a leased line, some physical path must exist between the two routers on the ends of the link. The physical cabling must leave the customer buildings where each router sits. However, the telco does not simply install one cable between the two buildings. Instead, it uses what is typically a large and complex network that creates the appearance of a cable between the two routers.

Figure 3-3 gives a little insight into the cabling that could exist inside the telco for a short leased line. Telcos put their equipment in buildings called central offices (CO). The telco installs cables from the CO to most every other building in the city, expecting to sell services to the people in those buildings one day. The telco would then configure its switches to use some of the capacity on each cable to send data in both directions, creating the equivalent of a crossover cable between the two routers.

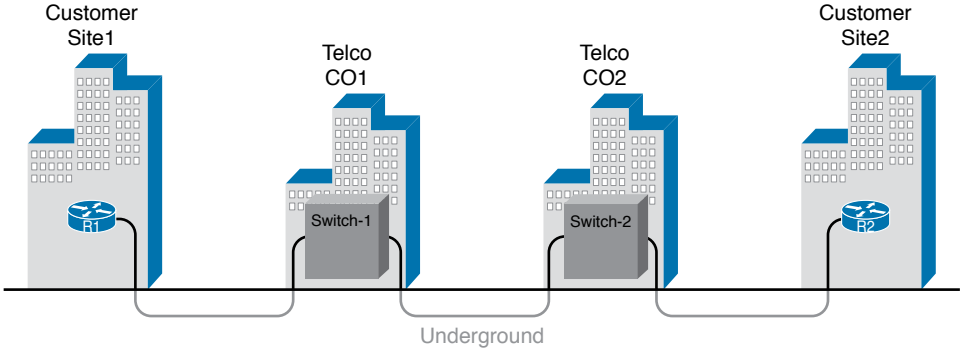


Figure 3-3 *Possible Cabling Inside a Telco for a Short Leased Line*

Answers to the "Do I Know This Already?" quiz:

1 B **2** B, D **3** A **4** C **5** C **6** A, C **7** C

Although the customer does not need to know all the details of how a telco creates a particular leased line, enterprise engineers do need to know about the parts of the link that exist inside the customer's building at the router. However, for the purposes of CCNA, you can think of any serial link as a point-to-point connection between two routers.

HDLC Data-Link Details of Leased Lines

A leased line provides a Layer 1 service. In other words, it promises to deliver bits between the devices connected to the leased line. However, the leased line itself does not define a data-link layer protocol to be used on the leased line.

Because leased lines define only the Layer 1 transmission service, many companies and standards organizations have created data-link protocols to control and use leased lines. Today, the two most popular data-link layer protocols used for leased lines between two routers are High-Level Data Link Control (HDLC) and Point-to-Point Protocol (PPP).

All data-link protocols perform a similar role: to control the correct delivery of data over a physical link of a particular type. For example, the Ethernet data-link protocol uses a destination address field to identify the correct device that should receive the data and an FCS field that allows the receiving device to determine whether the data arrived correctly. HDLC provides similar functions.

HDLC has less work to do than Ethernet because of the simple point-to-point topology of a leased line. When one router sends an HDLC frame, the frame can go only one place: to the other end of the link. So, while HDLC has an address field, the destination is implied, and the actual address is unimportant. The idea is sort of like when I have lunch with my friend Gary, and only Gary. I do not need to start every sentence with "Hey, Gary"—he knows I am talking to him.

HDLC has other fields and functions similar to Ethernet as well. Table 3-3 lists the HDLC fields, with the similar Ethernet header/trailer field, just for the sake of learning HDLC based on something you have already learned about (Ethernet).

Table 3-3 Comparing HDLC Header Fields to Ethernet

HDLC Field	Ethernet Equivalent	Description
Flag	Preamble, SFD	Lists a recognizable bit pattern so that the receiving nodes realize that a new frame is arriving.
Address	Destination Address	Identifies the destination device.
Control	N/A	Mostly used for purposes no longer in use today for links between routers.
Type	Type	Identifies the type of Layer 3 packet encapsulated inside the frame.
FCS	FCS	Identifies a field used by the error detection process. (It is the only trailer field in this table.)

HDLC exists today as a standard of the International Organization for Standardization (ISO), the same organization that brought us the OSI model. However, ISO standard HDLC does not have a Type field, and routers need to know the type of packet inside the frame. So, Cisco routers use a Cisco-proprietary variation of HDLC that adds a Type field, as shown in Figure 3-4.

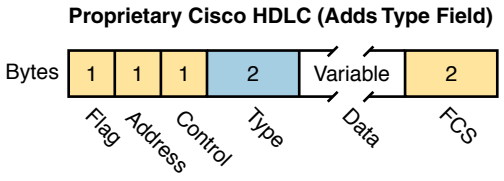


Figure 3-4 *HDLC Framing*

How Routers Use a WAN Data Link

Leased lines connect to routers, and routers focus on delivering packets to a destination host. However, routers physically connect to both LANs and WANs, with those LANs and WANs requiring that data be sent inside data-link frames. So, now that you know a little about HDLC, it helps to think about how routers use the HDLC protocol when sending data.

First, the TCP/IP network layer focuses on forwarding IP packets from the sending host to the destination host. The underlying LANs and WANs just act as a way to move the packets to the next router or end-user device. Figure 3-5 shows that network layer perspective.

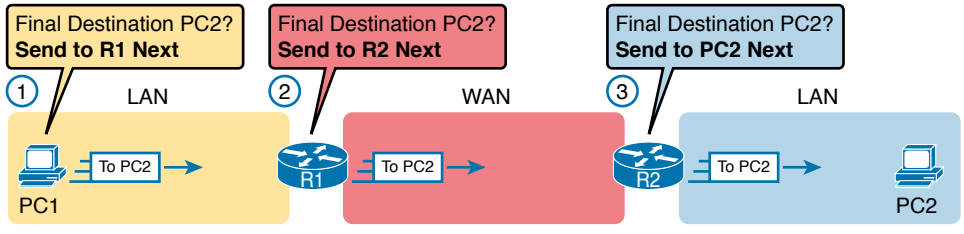


Figure 3-5 *IP Routing Logic over LANs and WANs*

Following the steps in the figure, for a packet sent by PC1 to PC2's IP address:

1. PC1's network layer (IP) logic tells it to send the packet to a nearby router (R1).
2. Router R1's network layer logic tells it to forward (route) the packet out the leased line to Router R2 next.
3. Router R2's network layer logic tells it to forward (route) the packet out the LAN link to PC2 next.

While Figure 3-5 shows the network layer logic, the PCs and routers must rely on the LANs and WANs in the figure to actually move the bits in the packet. Figure 3-6 shows the same figure, with the same packet, but this time showing some of the data-link layer logic used by the hosts and routers. Basically, three separate data-link layer steps encapsulate the packet, inside a data-link frame, over three hops through the internetwork: from PC1 to R1, from R1 to R2, and from R2 to PC2.

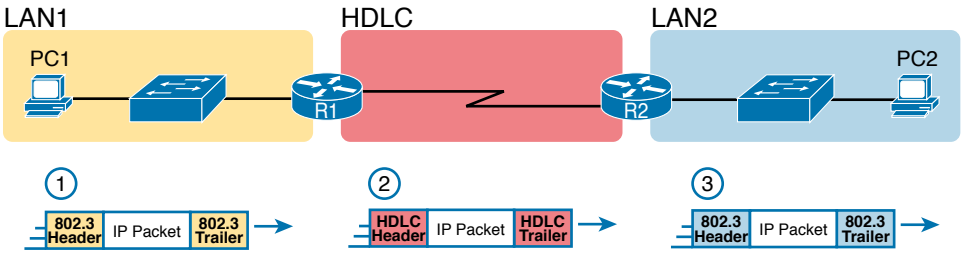


Figure 3-6 *General Concept of Routers De-encapsulating and Re-encapsulating IP Packets*

Following the steps in the figure, again for a packet sent by PC1 to PC2's IP address:

1. To send the IP packet to Router R1 next, PC1 encapsulates the IP packet in an Ethernet frame that has the destination MAC address of R1.
2. Router R1 de-encapsulates (removes) the IP packet from the Ethernet frame, encapsulates the packet into an HDLC frame using an HDLC header and trailer, and forwards the HDLC frame to Router R2 next.
3. Router R2 de-encapsulates (removes) the IP packet from the HDLC frame, encapsulates the packet into an Ethernet frame that has the destination MAC address of PC2, and forwards the Ethernet frame to PC2.

In summary, a leased line with HDLC creates a WAN link between two routers so that they can forward packets for the devices on the attached LANs. The leased line itself provides the physical means to transmit the bits, in both directions. The HDLC frames provide the means to encapsulate the network layer packet correctly so that it crosses the link between routers.

Leased lines have many benefits that have led to their relatively long life in the WAN marketplace. These lines are simple for the customer, are widely available, are of high quality, and are private. However, they do have some negatives as well compared to newer WAN technologies, including a higher cost and typically longer lead times to get the service installed. Additionally, by today's standards, leased-line LANs are slow, with faster speeds in the tens of megabits per second (Mbps). New faster WAN technology has been replacing leased lines for a long time, including the second WAN technology discussed in this book: Ethernet.

Ethernet as a WAN Technology

For the first several decades of the existence of Ethernet, Ethernet was only appropriate for LANs. The restrictions on cable lengths and devices might allow a LAN that stretched a kilometer or two, to support a campus LAN, but that was the limit.

As time passed, the IEEE improved Ethernet standards in ways that made Ethernet a reasonable WAN technology. For example, the 1000BASE-LX standard uses single-mode fiber cabling, with support for a 5-km cable length; the 1000BASE-ZX standard supports an even longer 70-km cable length. As time went by, and as the IEEE improved cabling distances for fiber Ethernet links, Ethernet became a reasonable WAN technology.

Today, many WAN service providers (SP) offer WAN services that take advantage of Ethernet. SPs offer a wide variety of these Ethernet WAN services, with many different names. But all of them use a similar model, with Ethernet used between the customer site and the SP's network, as shown in Figure 3-7.

Key Topic

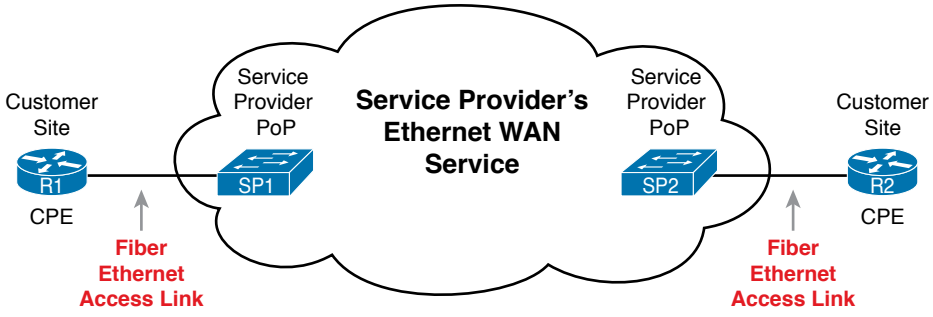


Figure 3-7 *Fiber Ethernet Link to Connect a CPE Router to a Service Provider's WAN*

The model shown in Figure 3-7 has many of the same ideas of how a telco creates a leased line, as shown earlier in Figure 3-3, but now with Ethernet links and devices. The customer connects to an Ethernet link using a router interface. The (fiber) Ethernet link leaves the customer building and connects to some nearby SP location called a point of presence (PoP). Instead of a telco switch as shown in Figure 3-3, the SP uses an Ethernet switch. Inside the SP's network, the SP uses any technology that it wants to create the specific Ethernet WAN services.

Ethernet WANs That Create a Layer 2 Service

Ethernet WAN services include a variety of specific services that vary in ways that change how routers use those services. However, for the purposes of CCNA, you just need to understand the most basic Ethernet WAN service, one that works much like an Ethernet crossover cable—just over a WAN. In other words:

- Logically, behaves like a point-to-point connection between two routers
- Physically, behaves as if a physical fiber Ethernet link existed between the two routers

NOTE For perspective about the broad world of the service provider network shown in Figure 3-7, look for more information about the Cisco CCNA, CCNP Service Provider, and CCIE Service Provider certifications. See www.cisco.com/go/certificationsfor more details.

This book refers to this particular Ethernet WAN service with a couple of the common names:

Ethernet WAN: A generic name to differentiate it from an Ethernet LAN.

Ethernet Line Service (E-Line): A term from the Metro Ethernet Forum (MEF) for the kind of point-to-point Ethernet WAN service shown throughout this book.

Ethernet emulation: A term emphasizing that the link is not a literal Ethernet link from end to end.

Ethernet over MPLS (EoMPLS): A term that refers to Multiprotocol Label Switching (MPLS), a technology that can be used to create the Ethernet service for the customer.

So, if you can imagine two routers, with a single Ethernet link between the two routers, you understand what this particular EoMPLS service does, as shown in Figure 3-8. In this case, the two routers, R1 and R2, connect with an EoMPLS service instead of a serial link. The routers use Ethernet interfaces, and they can send data in both directions at the same time. Physically, each router actually connects to some SP PoP, as shown earlier in Figure 3-7, but logically, the two routers can send Ethernet frames to each other over the link.

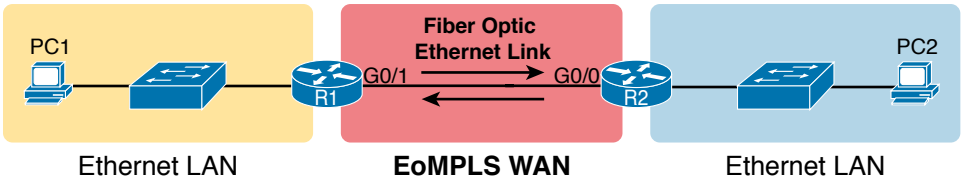


Figure 3-8 *EoMPLS Acting Like a Simple Ethernet Link Between Two Routers*

How Routers Route IP Packets Using Ethernet Emulation

WANs, by their very nature, give IP routers a way to forward IP packets from a LAN at one site, over the WAN, and to another LAN at another site. Routing over an EoMPLS WAN link still uses the WAN like a WAN, as a way to forward IP packets from one site to another. However, the WAN link happens to use the same Ethernet protocols as the Ethernet LAN links at each site.

The EoMPLS link uses Ethernet for both Layer 1 and Layer 2 functions. That means the link uses the same familiar Ethernet header and trailer, as shown in the middle of Figure 3-9. Note that the figure shows a small cloud over the Ethernet link as a way to tell us that the link is an Ethernet WAN link, rather than an Ethernet LAN link.

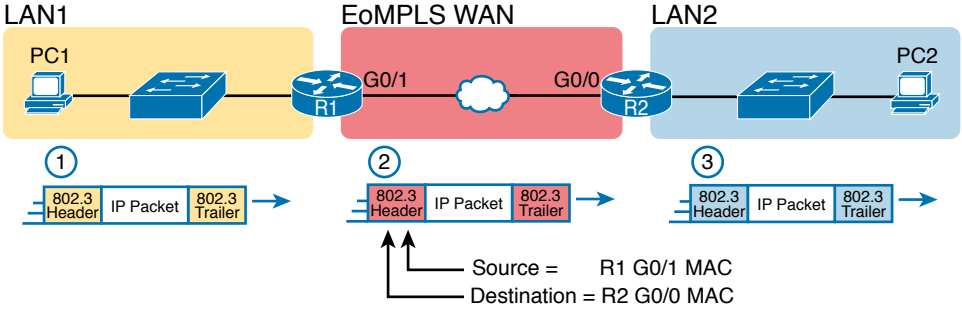


Figure 3-9 *Routing over an EoMPLS Link*

NOTE The 802.3 headers/trailers in the figure are different at each stage! Make sure to notice the reasons in the step-by-step explanations that follow.

The figure shows the same three routing steps as shown with the serial link in the earlier Figure 3-6. In this case, all three routing steps use the same Ethernet (802.3) protocol. However, note that each frame's data-link header and trailer are different. Each router

discards the old data-link header/trailer and adds a new set, as described in these steps. Focus mainly on Step 2, because compared to the similar example shown in Figure 3-6, Steps 1 and 3 are unchanged:

1. To send the IP packet to Router R1 next, PC1 encapsulates the IP packet in an Ethernet frame that has the destination MAC address of R1.
2. Router R1 de-encapsulates (removes) the IP packet from the Ethernet frame and encapsulates the packet into a new Ethernet frame, with a new Ethernet header and trailer. The destination MAC address is R2's G0/0 MAC address, and the source MAC address is R1's G0/1 MAC address. R1 forwards this frame over the EoMPLS service to R2 next.
3. Router R2 de-encapsulates (removes) the IP packet from the Ethernet frame, encapsulates the packet into an Ethernet frame that has the destination MAC address of PC2, and forwards the Ethernet frame to PC2.

Throughout this book, the WAN links (serial and Ethernet) will connect routers as shown here, with the focus being on the LANs and IP routing. The rest of the chapter turns our attention to a closer look at IP routing.

IP Routing

Many protocol models have existed over the years, but today the TCP/IP model dominates. And at the network layer of TCP/IP, two options exist for the main protocol around which all other network layer functions revolve: IP version 4 (IPv4) and IP version 6 (IPv6). Both IPv4 and IPv6 define the same kinds of network layer functions, but with different details. This chapter introduces these network layer functions for IPv4.

NOTE All references to IP in this chapter refer to the older and more established IPv4.

Internet Protocol (IP) focuses on the job of routing data, in the form of IP packets, from the source host to the destination host. IP does not concern itself with the physical transmission of data, instead relying on the lower TCP/IP layers to do the physical transmission of the data. Instead, IP concerns itself with the logical details, rather than physical details, of delivering data. In particular, the network layer specifies how packets travel end to end over a TCP/IP network, even when the packet crosses many different types of LAN and WAN links.

This next major section of the chapter examines IP routing in more depth. First, IP defines what it means to route an IP packet from sending host to destination host, while using successive data-link protocols. This section then examines how IP addressing rules help to make IP routing much more efficient by grouping addresses into subnets. This section closes by looking at the role of IP routing protocols, which give routers a means by which to learn routes to all the IP subnets in an internetwork.

Network Layer Routing (Forwarding) Logic

Routers and end-user computers (called *hosts* in a TCP/IP network) work together to perform IP routing. The host operating system (OS) has TCP/IP software, including the software that implements the network layer. Hosts use that software to choose where to send IP packets,

often to a nearby router. Those routers make choices of where to send the IP packet next. Together, the hosts and routers deliver the IP packet to the correct destination, as shown in the example in Figure 3-10.

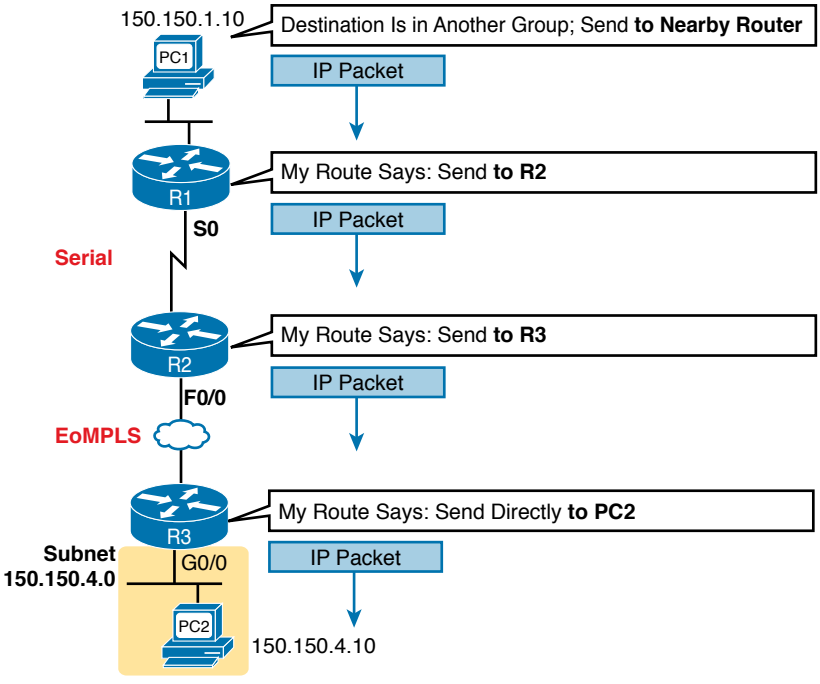


Figure 3-10 *Routing Logic: PC1 Sending an IP Packet to PC2*

The IP packet, created by PC1, goes from the top of the figure all the way to PC2 at the bottom of the figure. The next few pages discuss the network layer routing logic used by each device along the path.

NOTE The term *path selection* is sometimes used to refer to the routing process shown in Figure 3-10. At other times, it refers to routing protocols, specifically how routing protocols select the best route among the competing routes to the same destination.

Host Forwarding Logic: Send the Packet to the Default Router

In this example, PC1 does some basic analysis and then chooses to send the IP packet to the router so that the router will forward the packet. PC1 analyzes the destination address and realizes that PC2's address (150.150.4.10) is not on the same LAN as PC1. So PC1's logic tells it to send the packet to a device whose job it is to know where to route data: a nearby router, on the same LAN, called PC1's default router.

To send the IP packet to the default router, the sender sends a data-link frame across the medium to the nearby router; this frame includes the packet in the data portion of the frame. That frame uses data-link layer (Layer 2) addressing in the data-link header to ensure that the nearby router receives the frame.

NOTE The *default router* is also referred to as the *default gateway*.

R1 and R2's Logic: Routing Data Across the Network

All routers use the same general process to route the packet. Each router keeps an *IP routing table*. This table lists IP address *groupings*, called *IP networks* and *IP subnets*. When a router receives a packet, it compares the packet's destination IP address to the entries in the routing table and makes a match. This matching entry also lists directions that tell the router where to forward the packet next.

In Figure 3-10, R1 would have matched the destination address (150.150.4.10) to a routing table entry, which in turn told R1 to send the packet to R2 next. Similarly, R2 would have matched a routing table entry that told R2 to send the packet, over an Ethernet WAN link, to R3 next.

The routing concept works a little like driving down the freeway when approaching a big interchange. You look up and see signs for nearby towns, telling you which exits to take to go to each town. Similarly, the router looks at the IP routing table (the equivalent of the road signs) and directs each packet over the correct next LAN or WAN link (the equivalent of a road).

R3's Logic: Delivering Data to the End Destination

The final router in the path, R3, uses almost the same logic as R1 and R2, but with one minor difference. R3 needs to forward the packet directly to PC2, not to some other router. On the surface, that difference seems insignificant. In the next section, when you read about how the network layer uses LANs and WANs, the significance of the difference will become obvious.

How Network Layer Routing Uses LANs and WANs

While the network layer routing logic ignores the physical transmission details, the bits still have to be transmitted. To do that work, the network layer logic in a host or router must hand off the packet to the data-link layer protocols, which, in turn, ask the physical layer to actually send the data. The data-link layer adds the appropriate header and trailer to the packet, creating a frame, before sending the frames over each physical network.

The routing process forwards the network layer packet from end to end through the network, while each data-link frame only takes a smaller part of the trip. Each successive data-link layer frame moves the packet to the next device that thinks about network layer logic. In short, the network layer thinks about the bigger view of the goal, like "Send this packet to the specified next router or host…," while the data-link layer thinks about the specifics, like "Encapsulate the packet in a data-link frame and transmit it." The following list summarizes the major steps in a router's internal network layer routing for each packet beginning with the a frame arriving in a router interface:

Key Topic

Step 1. Use the data-link Frame Check Sequence (FCS) field to ensure that the frame had no errors; if errors occurred, discard the frame.

Step 2. Assuming that the frame was not discarded at Step 1, discard the old data-link header and trailer, leaving the IP packet.

Step 3. Compare the IP packet's destination IP address to the routing table, and find the route that best matches the destination address. This route identifies the outgoing interface of the router and possibly the next-hop router IP address.

Step 4. Encapsulate the IP packet inside a new data-link header and trailer, appropriate for the outgoing interface, and forward the frame.

Figure 3-11 works through a repeat example of a packet sent by PC1 to PC2, followed by a detailed analysis of each device's routing logic. Each explanation includes the details about how PC1 and each of the three routers builds the appropriate new data-link headers.

Key Topic

R1 Routing Table

Subnet	Interface	Next Hop
150.150.4.0	Serial0	150.150.2.7

R2 Routing Table

Subnet	Interface	Next Hop
150.150.4.0	FastEth0/0	150.150.3.1

R3 Routing Table

Subnet	Interface	Next Hop
150.150.4.0	Gigabit0/0	N/A

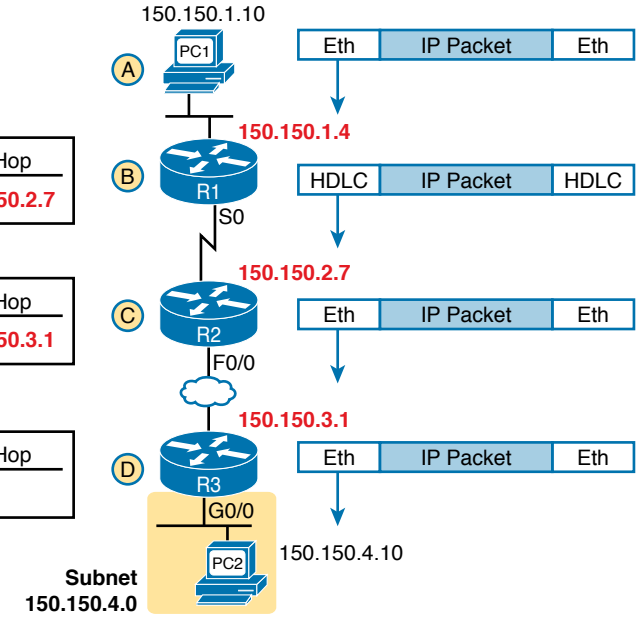


Figure 3-11 *Network Layer and Data-Link Layer Encapsulation*

The following list explains the forwarding logic at each router, focusing on how the routing integrates with the data link.

Step A. **PC1 sends the packet to its default router.** PC1's network layer logic builds the IP packet, with a destination address of PC2's IP address (150.150.4.10). The network layer also performs the analysis to decide that 150.150.4.10 is not in the local IP subnet, so PC1 needs to send the packet to R1 (PC1's default router). PC1 places the IP packet into an Ethernet data-link frame, with a destination Ethernet address of R1's Ethernet address. PC1 sends the frame on to the Ethernet.

Step B. **R1 processes the incoming frame and forwards the packet to R2.** Because the incoming Ethernet frame has a destination MAC of R1's Ethernet MAC, R1 decides to process the frame. R1 checks the frame's FCS for errors, and if none, R1 discards the Ethernet header and trailer. Next, R1 compares the packet's

destination address (150.150.4.10) to its routing table and finds the entry for subnet 150.150.4.0. Because the destination address of 150.150.4.10 is in that subnet, R1 forwards the packet out the interface listed in that matching route (Serial0) to next-hop Router R2 (150.150.2.7). R1 must first encapsulate the IP packet into an HDLC frame.

Step C. **R2 processes the incoming frame and forwards the packet to R3.** R2 repeats the same general process as R1 when R2 receives the HDLC frame. R2 checks the FCS field and finds that no errors occurred and then discards the HDLC header and trailer. Next, R2 compares the packet's destination address (150.150.4.10) to its routing table and finds the entry for subnet 150.150.4.0, a route that directs R2 to send the packet out interface Fast Ethernet 0/0 to next-hop router 150.150.3.1 (R3). But first, R2 must encapsulate the packet in an Ethernet header. That header uses R2's MAC address and R3's MAC address on the Ethernet WAN link as the source and destination MAC address, respectively.

Step D. **R3 processes the incoming frame and forwards the packet to PC2.** Like R1 and R2, R3 checks the FCS, discards the old data-link header and trailer, and matches its own route for subnet 150.150.4.0. R3's routing table entry for 150.150.4.0 shows that the outgoing interface is R3's Ethernet interface, but there is no next-hop router because R3 is connected directly to subnet 150.150.4.0. All R3 has to do is encapsulate the packet inside a new Ethernet header and trailer, but with a destination Ethernet address of PC2's MAC address.

Because the routers build new data-link headers and trailers, and because the new headers contain data-link addresses, the PCs and routers must have some way to decide what data-link addresses to use. An example of how the router determines which data-link address to use is the IP Address Resolution Protocol (ARP). *ARP dynamically learns the data-link address of an IP host connected to a LAN*. For example, at the last step, at the bottom of Figure 3-11, Router R3 would use ARP once to learn PC2's MAC address before sending any packets to PC2.

How IP Addressing Helps IP Routing

IP defines network layer addresses that identify any host or router interface that connects to a TCP/IP network. The idea basically works like a postal address: Any interface that expects to receive IP packets needs an IP address, just like you need a postal address before receiving mail from the postal service. This next short topic introduces the idea of IP networks and subnets, which are the groups of addresses defined by IP.

NOTE IP defines the word *network* to mean a very specific concept. To avoid confusion when writing about IP addressing, this book (and others) often avoids using the term *network* for other uses. In particular, this book uses the term *internetwork* to refer more generally to a network made up of routers, switches, cables, and other equipment.

Rules for Groups of IP Addresses (Networks and Subnets)

TCP/IP groups IP addresses together so that IP addresses used on the same physical network are part of the same group. IP calls these address groups an *IP network* or an *IP subnet*. Using that same postal service analogy, each IP network and IP subnet works like a postal code (or in the United States, a ZIP code). All nearby postal addresses are in the same postal code (ZIP code), while all nearby IP addresses must be in the same IP network or IP subnet.

IP defines specific rules about which IP address should be in the same IP network or IP subnet. Numerically, the addresses in the same group have the same value in the first part of the addresses. For example, Figures 3-10 and 3-11 could have used the following conventions:

- Hosts on the top Ethernet: Addresses start with 150.150.1
- Hosts on the R1–R2 serial link: Addresses start with 150.150.2
- Hosts on the R2–R3 EoMPLS link: Addresses start with 150.150.3
- Hosts on the bottom Ethernet: Addresses start with 150.150.4

From the perspective of IP routing, the grouping of IP addresses means that the routing table can be much smaller. A router can list one routing table entry for each IP network or subnet, instead of one entry for every single IP address.

While the list shows just one example of how IP addresses may be grouped, the rules for how to group addresses using subnets will require some work to master the concepts and math. Part III of this book details IP addressing and subnetting, and you can find other subnetting video and practice products listed in the Introduction to the book. However, the brief version of two of the foundational rules of subnetting can be summarized as follows:

Key Topic

- Two IP addresses, not separated from each other by a router, must be in the same group (subnet).
- Two IP addresses, separated from each other by at least one router, must be in different groups (subnets).

It's similar to the USPS ZIP code system and how it requires local governments to assign addresses to new buildings. It would be ridiculous to have two houses next door to each other, whose addresses had different postal/ZIP codes. Similarly, it would be silly to have people who live on opposite sides of the country to have addresses with the same postal/ZIP code.

The IP Header

The routing process also makes use of the IPv4 header, as shown in Figure 3-12. The header lists a 32-bit source IP address, as well as a 32-bit destination IP address. The header, of course, has other fields, a few of which matter for other discussions in this book. The book will refer to this figure as needed, but otherwise, be aware of the 20-byte IP header and the existence of the source and destination IP address fields. Note that in the examples so far in this chapter, while routers remove and add data-link headers each time it routes a packet, the IP header remains, with the IP addresses unchanged by the IP routing process.

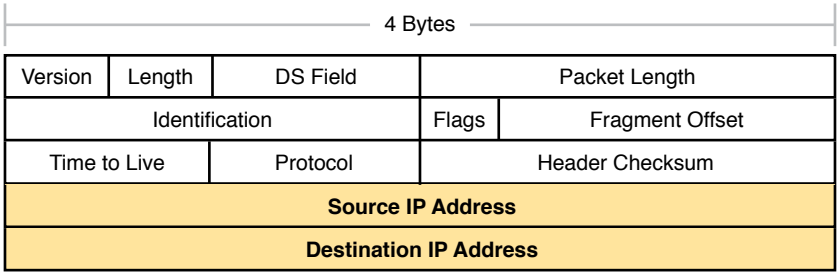


Figure 3-12 *IPv4 Header, Organized as 4 Bytes Wide for a Total of 20 Bytes*

How IP Routing Protocols Help IP Routing

For routing logic to work on both hosts and routers, each host and router needs to know something about the TCP/IP internetwork. Hosts need to know the IP address of their default router so that hosts can send packets to remote destinations. Routers, however, need to know routes so they forward packets to each and every reachable IP network and IP subnet.

The best method for routers to know all the useful routes is to configure the routers to use the same IP routing protocol. Alternately, a network engineer could configure (type) all the required routes, on every router. However, if you enable the same routing protocol on all the routers in a TCP/IP internetwork, with the correct settings, the routers will send routing protocol messages to each other. As a result, all the routers will learn routes for all the IP networks and subnets in the TCP/IP internetwork.

IP supports a small number of different IP routing protocols. All use some similar ideas and processes to learn IP routes, but different routing protocols do have some internal differences; otherwise, you would not need more than one routing protocol. However, many routing protocols use the same general steps for learning routes:

Key Topic

Step 1. Each router, independent of the routing protocol, adds a route to its routing table for each subnet directly connected to the router.

Step 2. Each router's routing protocol tells its neighbors about the routes in its routing table, including the directly connected routes and routes learned from other routers.

Step 3. After learning a new route from a neighbor, the router's routing protocol adds a route to its IP routing table, with the next-hop router of that route typically being the neighbor from which the route was learned.

Also, note that at the final step, routers may have to choose between multiple routes to reach a single subnet. When that happens, routers place the best currently available route to reach a subnet (based on a measurement called a metric) into the routing table.

Figure 3-13 shows an example of how a routing protocol works, using the same diagram as in Figures 3-10 and 3-11. In this case, IP subnet 150.150.4.0, which consists of all addresses that begin with 150.150.4.0, sits on the Ethernet at the bottom of the figure. The figure shows the advertisement of routes for subnet 150.150.4.0 from bottom to top, as described in detail following the figure.

Key Topic

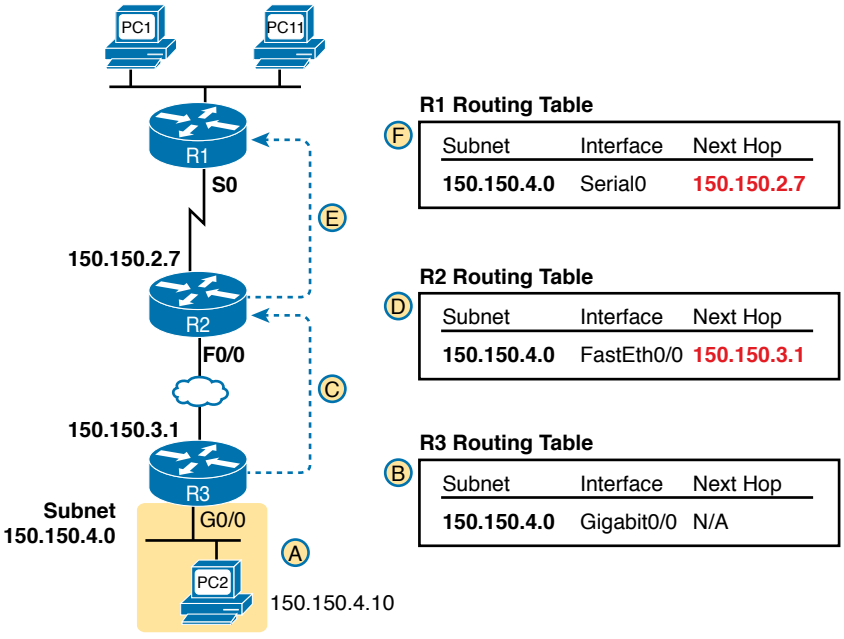


Figure 3-13 *Example of How Routing Protocols Advertise About Networks and Subnets*

Follow items A through F shown in the figure to see how each router learns its route to 150.150.4.0.

Step A. Subnet 150.150.4.0 exists as a subnet at the bottom of the figure, connected to Router R3.

Step B. R3 adds a connected route for 150.150.4.0 to its IP routing table; this happens without help from the routing protocol.

Step C. R3 sends a routing protocol message, called a *routing update*, to R2, causing R2 to learn about subnet 150.150.4.0.

Step D. R2 adds a route for subnet 150.150.4.0 to its routing table.

Step E. R2 sends a similar routing update to R1, causing R1 to learn about subnet 150.150.4.0.

Step F. R1 adds a route for subnet 150.150.4.0 to its routing table. The route lists R1's own Serial0 as the outgoing interface and R2 as the next-hop router IP address (150.150.2.7).

Other Network Layer Features

The TCP/IP network layer defines many functions beyond IP. Sure, IP plays a huge role in networking today, defining IP addressing and IP routing. However, other protocols and standards, defined in other Requests For Comments (RFC), play an important role for network layer functions as well. For example, routing protocols like Open Shortest Path First (OSPF) exist as separate protocols, defined in separate RFCs.

This last short section of the chapter introduces three other network layer features that should be helpful to you when reading through the rest of this book. These last three topics just help fill in a few holes, helping to give you some perspective and helping you make sense of later discussions as well. The three topics are

- Domain Name System (DNS)
- Address Resolution Protocol (ARP)
- Ping

Using Names and the Domain Name System

Can you imagine a world in which every time you used an application, you had to refer to it by IP address? Instead of using easy names like google.com or facebook.com, you would have to remember and type IP addresses, like 64.233.177.100. (At press time, 64.233.177.100 was an address used by Google, and you could reach Google's website by typing that address in a browser.) Certainly, asking users to remember IP addresses would not be user friendly and could drive some people away from using computers at all.

Thankfully, TCP/IP defines a way to use *hostnames* to identify other computers. The user either never thinks about the other computer or refers to the other computer by name. Then, protocols dynamically discover all the necessary information to allow communications based on that name.

For example, when you open a web browser and type in the hostname **www.google.com**, your computer does not send an IP packet with destination IP address www.google.com; it sends an IP packet to an IP address used by the web server for Google. TCP/IP needs a way to let a computer find the IP address used by the listed hostname, and that method uses the Domain Name System (DNS).

Enterprises use the DNS process to resolve names into the matching IP address, as shown in the example in Figure 3-14. In this case, PC11, on the left, needs to connect to a server named Server1. At some point, the user either types in the name Server1 or some application on PC11 refers to that server by name. At Step 1, PC11 sends a DNS message—a DNS query—to the DNS server. At Step 2, the DNS server sends back a DNS reply that lists Server1's IP address. At Step 3, PC11 can now send an IP packet to destination address 10.1.2.3, the address used by Server1.

Key Topic

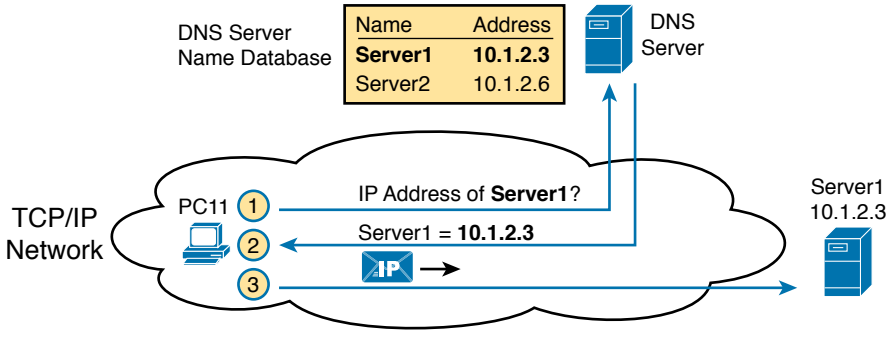


Figure 3-14 *Basic DNS Name Resolution Request*

Note that the example in Figure 3-14 shows a cloud for the TCP/IP network because the details of the network, including routers, do not matter to the name resolution process. Routers treat the DNS messages just like any other IP packet, routing them based on the destination IP address. For example, at Step 1 in the figure, the DNS query will list the DNS server's IP address as the destination address, which any routers will use to forward the packet.

Finally, DNS defines much more than just a few messages. DNS defines protocols, as well as standards for the text names used throughout the world, and a worldwide set of distributed DNS servers. The domain names that people use every day when web browsing, which look like www.example.com,follow the DNS naming standards. Also, no single DNS server knows all the names and matching IP addresses, but the information is distributed across many DNS servers. So, the DNS servers of the world work together, forwarding queries to each other, until the server that knows the answer supplies the desired IP address information.

The Address Resolution Protocol

As discussed in depth throughout this chapter, IP routing logic requires that hosts and routers encapsulate IP packets inside data-link layer frames. For Ethernet interfaces, how does a router know what MAC address to use for the destination? It uses ARP.

On Ethernet LANs, whenever a host or router needs to encapsulate an IP packet in a new Ethernet frame, the host or router knows all the important facts to build that header—except for the destination MAC address. The host knows the IP address of the next device, either another host IP address or the default router IP address. A router knows the IP route used for forwarding the IP packet, which lists the next router's IP address. However, the hosts and routers do not know those neighboring devices' MAC addresses beforehand.

TCP/IP defines the Address Resolution Protocol (ARP) as the method by which any host or router on a LAN can dynamically learn the MAC address of another IP host or router on the same LAN. ARP defines a protocol that includes the *ARP Request*, which is a message that makes the simple request "if this is your IP address, please reply with your MAC address." ARP also defines the *ARP Reply* message, which indeed lists both the original IP address and the matching MAC address.

Figure 3-15 shows an example that uses the same router and host from the bottom part of the earlier Figure 3-13. The figure shows the ARP Request sent by router R3, on the left of the figure, as a LAN broadcast. All devices on the LAN will then process the received frame. On the right, at Step 2, host PC2 sends back an ARP Reply, identifying PC2's MAC address. The text beside each message shows the contents inside the ARP message itself, which lets PC2 learn R3's IP address and matching MAC address, and R3 learn PC2's IP address and matching MAC address.

Note that hosts and routers remember the ARP results, keeping the information in their *ARP cache* or *ARP table*. A host or router only needs to use ARP occasionally, to build the ARP cache the first time. Each time a host or router needs to send a packet encapsulated in an Ethernet frame, it first checks its ARP cache for the correct IP address and matching MAC address. Hosts and routers will let ARP cache entries time out to clean up the table, so occasional ARP Requests can be seen.

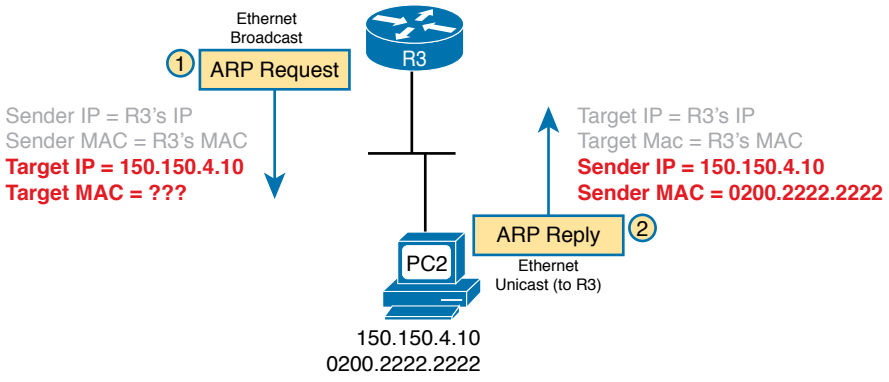


Figure 3-15 *Sample ARP Process*

NOTE You can see the contents of the ARP cache on most PC operating systems by using the **arp -a** command from a command prompt.

ICMP Echo and the ping Command

After you have implemented a TCP/IP internetwork, you need a way to test basic IP connectivity without relying on any applications to be working. The primary tool for testing basic network connectivity is the **ping** command.

Ping (Packet Internet Groper) uses the Internet Control Message Protocol (ICMP), sending a message called an *ICMP echo request* to another IP address. The computer with that IP address should reply with an *ICMP echo reply*. If that works, you successfully have tested the IP network. In other words, you know that the network can deliver a packet from one host to the other and back. ICMP does not rely on any application, so it really just tests basic IP connectivity—Layers 1, 2, and 3 of the OSI model. Figure 3-16 outlines the basic process.

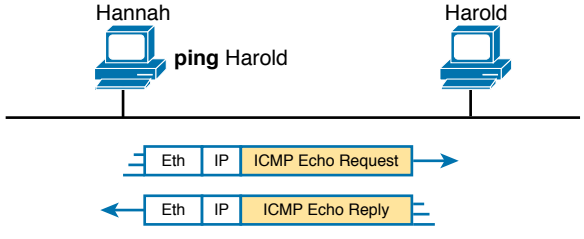


Figure 3-16 *Sample Network, ping Command*

Note that while the **ping** command uses ICMP, ICMP does much more. ICMP defines many messages that devices can use to help manage and control the IP network.

Chapter Review

The "Your Study Plan" element, just before Chapter 1, discusses how you should study and practice the content and skills for each chapter before moving on to the next chapter. That element introduces the tools used here at the end of each chapter. If you haven't already done so, take a few minutes to read that section. Then come back here and do the useful work of reviewing the chapter to help lock into memory what you just read.

Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Table 3-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 3-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website

Review All the Key Topics

Key Topic

Table 3-5 Key Topics for Chapter 3

Key Topic Element	Description	Page Number
Figure 3-7	Ethernet over MPLS—physical connections	66
List	Four-step process of how routers route (forward) packets	70
Figure 3-11	IP Routing and Encapsulation	71
List	Two statements about how IP expects IP addresses to be grouped into networks or subnets	73
List	Three-step process of how routing protocols learn routes	74
Figure 3-13	IP Routing Protocol Basic Process	75
Figure 3-14	Example that shows the purpose and process of DNS name resolution	76
Figure 3-15	Example of the purpose and process of ARP	78

Key Terms You Should Know

leased line, wide-area network (WAN), telco, serial interface, HDLC, Ethernet over MPLS, Ethernet Line Service (E-Line), default router (default gateway), routing table, IP network, IP subnet, IP packet, routing protocol, dotted-decimal notation (DDN), IPv4 address, unicast IP address, subnetting, hostname, DNS, ARP, ping

Part I Review

Keep track of your part review progress with the checklist shown in Table P1-1. Details on each task follow the table.

Table P1-1 Part I Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book, using the PTP software. Refer to the Introduction to this book, the section titled "How to View Only DIKTA Questions by Chapter or Part," for help with how to make the PTP software show you DIKTA questions for this part only.

Answer Part Review Questions

For this task, answer the Part Review questions for this part of the book, using the PTP software. Refer to the Introduction to this book, the section titled "How to View Part Review Questions," for help with how to make the PTP software show you Part Review questions for this part only. (Note that if you use the questions but then want even more, get the Premium Edition of the book, as detailed in the Introduction, in the section "Other Features," under the item labeled "eBook.")

Review Key Topics

Browse back through the chapters and look for the Key Topic icons. If you do not remember some details, take the time to reread those topics, or use the Key Topics application(s) found on the companion website.

Use Per-Chapter Interactive Review Elements

Using the companion website, browse through the interactive review elements, like memory tables and key term flashcards, to review the content from each chapter.

This page intentionally left blank

Part I provided a broad look at the fundamentals of all parts of networking, focusing on Ethernet LANs, WANs, and IP routing. Parts II and III now drill into depth about the details of Ethernet, which was introduced in Chapter 2, "Fundamentals of Ethernet LANs."

Part II begins that journey by discussing the basics of building a small Ethernet LAN with Cisco Catalyst switches. The journey begins by showing how to access the user interface of a Cisco switch so that you can see evidence of what the switch is doing and configure the switch to act in the ways you want it to act. At this point, you should start using whatever lab practice option you chose in the "Your Study Plan" section that preceded Chapter 1, "Introduction to TCP/IP Networking." (And if you have not yet finalized your plan for how to practice your hands-on skills, now is the time.)

After you complete Chapter 4 and see how to get into the command-line interface (CLI) of a switch, the next three chapters step through some important foundations of how to implement LANs—foundations used by every company that builds LANs with Cisco gear. Chapter 5 takes a close look at Ethernet switching—that is, the logic used by a switch—and how to know what a particular switch is doing. Chapter 6 shows the ways to configure a switch for remote access with Telnet and Secure Shell (SSH), along with a variety of other useful commands that will help you when you work with any real lab gear, simulator, or any other practice tools. Chapter 7, the final chapter in Part II, shows how to configure and verify the operation of switch interfaces for several important features, including speed, duplex, and autonegotiation.

Part II

Implementing Ethernet LANs

CHAPTER 4

Using the Command-Line Interface

This chapter covers the following exam topics:

None

This chapter explains foundational skills required before you can learn about the roughly 15 exam topics that use the verbs *configure* and *verify*. However, Cisco does not list the foundational skills described in this chapter as a separate exam topic, so there are no specific exam topics included in this chapter.

To create an Ethernet LAN, network engineers start by planning. They consider the requirements, create a design, buy the switches, contract to install cables, and configure the switches to use the right features.

The CCNA exam focuses on skills like understanding how LANs work, configuring different switch features, verifying that those features work correctly, and finding the root cause of the problem when a feature is not working correctly. The first skill you need to learn before doing all the configuration and verification tasks is to learn how to access and use the user interface of the switch, called the command-line interface (CLI).

This chapter begins that process by showing the basics of how to access the switch's CLI. These skills include how to access the CLI and how to issue verification commands to check on the status of the LAN. This chapter also includes the processes of how to configure the switch and how to save that configuration.

Note that this chapter focuses on processes that provide a foundation for most every exam topic that includes the verbs *configure* and/or *verify*. Most of the rest of the chapters in Parts II and III of this book then go on to include details of the particular commands you can use to verify and configure different switch features.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 4-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Accessing the Cisco Catalyst Switch CLI	1–3
Configuring Cisco IOS Software	4–6

1. In what modes can you type the command **show mac address-table** and expect to get a response with MAC table entries? (Choose two answers.)
   a. User mode
   b. Enable mode
   c. Global configuration mode
   d. Interface configuration mode
2. In which of the following modes of the CLI could you type the command **reload** and expect the switch to reboot?
   a. User mode
   b. Enable mode
   c. Global configuration mode
   d. Interface configuration mode
3. Which of the following is a difference between Telnet and SSH as supported by a Cisco switch?
   a. SSH encrypts the passwords used at login, but not other traffic; Telnet encrypts nothing.
   b. SSH encrypts all data exchange, including login passwords; Telnet encrypts nothing.
   c. Telnet is used from Microsoft operating systems, and SSH is used from UNIX and Linux operating systems.
   d. Telnet encrypts only password exchanges; SSH encrypts all data exchanges.
4. What type of switch memory is used to store the configuration used by the switch when it is up and working?
   a. RAM
   b. ROM
   c. Flash
   d. NVRAM
   e. Bubble
5. What command copies the configuration from RAM into NVRAM?
   a. **copy running-config tftp**
   b. **copy tftp running-config**
   c. **copy running-config start-up-config**
   d. **copy start-up-config running-config**
   e. **copy startup-config running-config**
   f. **copy running-config startup-config**

6. A switch user is currently in console line configuration mode. Which of the following would place the user in enable mode? (Choose two answers.)

   a. Using the **exit** command once
   b. Using the **end** command once
   c. Pressing the Ctrl+Z key sequence once
   d. Using the **quit** command

Foundation Topics

Accessing the Cisco Catalyst Switch CLI

Cisco uses the concept of a command-line interface (CLI) with its router products and most of its Catalyst LAN switch products. The CLI is a text-based interface in which the user, typically a network engineer, enters a text command and presses Enter. Pressing Enter sends the command to the switch, which tells the device to do something. The switch does what the command says, and in some cases, the switch replies with some messages stating the results of the command.

Cisco Catalyst switches also support other methods to both monitor and configure a switch. For example, a switch can provide a web interface so that an engineer can open a web browser to connect to a web server running in the switch. Switches also can be controlled and operated using network management software.

This book discusses only Cisco Catalyst enterprise-class switches, and in particular, how to use the Cisco CLI to monitor and control these switches. This first major section of the chapter first examines these Catalyst switches in more detail and then explains how a network engineer can get access to the CLI to issue commands.

Cisco Catalyst Switches

Within the Cisco Catalyst brand of LAN switches, Cisco produces a wide variety of switch series or families. Each switch series includes several specific models of switches that have similar features, similar price-versus-performance tradeoffs, and similar internal components.

For example, at the time this book was published, the Cisco 2960-XR series of switches was a current switch model series. Cisco positions the 2960-XR series (family) of switches as full-featured, low-cost wiring closet switches for enterprises. That means that you would expect to use 2960-XR switches as access switches in a typical campus LAN design.

Figure 4-1 shows a photo of 10 different models from the 2960-XR switch model series from Cisco. Each switch series includes several models, with a mix of features. For example, some of the switches have 48 RJ-45 unshielded twisted-pair (UTP) 10/100/1000 ports, meaning that these ports can autonegotiate the use of 10BASE-T (10 Mbps), 100BASE-T (100 Mbps), or 1000BASE-T (1 Gbps) Ethernet.

Figure 4-1 *Cisco 2960-XR Catalyst Switch Series*

Cisco refers to a switch's physical connectors as either *interfaces* or *ports*, with an interface type and interface number. The interface type, as used in commands on the switch, is either Ethernet, Fast Ethernet, Gigabit Ethernet, and so on for faster speeds. For Ethernet interfaces that support running at multiple speeds, the permanent name for the interface refers to the fastest supported speed. For example, a 10/100/1000 interface (that is, an interface that runs at 10 Mbps, 100 Mbps, or 1000 Mbps) would be called Gigabit Ethernet no matter what speed is currently in use.

To uniquely number each different interface, some Catalyst switches use a two-digit interface number (*x*/*y*), while others have a three-digit number (*x*/*y*/*z*). For instance, two 10/100/1000 ports on many older Cisco Catalyst switches would be called GigabitEthernet 0/0 and GigabitEthernet 0/1, while on the newer 2960-XR series, two interfaces would be GigabitEthernet 1/0/1 and GigabitEthernet 1/0/2.

Accessing the Cisco IOS CLI

Like any other piece of computer hardware, Cisco switches need some kind of operating system software. Cisco calls this OS the Internetwork Operating System (IOS).

Cisco IOS Software for Catalyst switches implements and controls logic and functions performed by a Cisco switch. Besides controlling the switch's performance and behavior, Cisco IOS also defines an interface for humans called the CLI. The Cisco IOS CLI allows the user to use a terminal emulation program, which accepts text entered by the user. When the user presses Enter, the terminal emulator sends that text to the switch. The switch processes the text as if it is a command, does what the command says, and sends text back to the terminal emulator.

The switch CLI can be accessed through three popular methods—the console, Telnet, and Secure Shell (SSH). Two of these methods (Telnet and SSH) use the IP network in which the switch resides to reach the switch. The console is a physical port built specifically to allow access to the CLI. Figure 4-2 depicts the options.

Key Topic

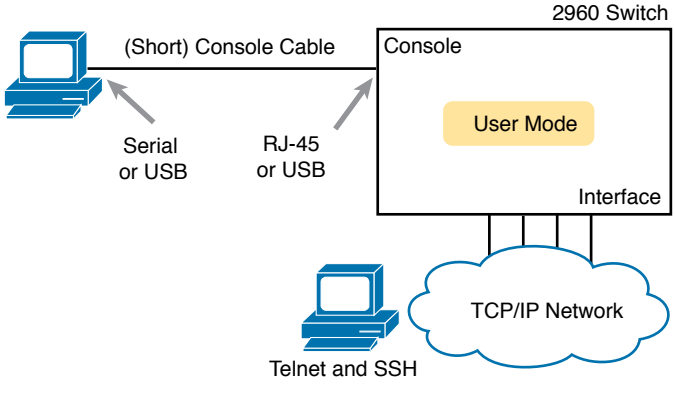


Figure 4-2 *CLI Access Options*

Console access requires both a physical connection between a PC (or other user device) and the switch's console port, as well as some software on the PC. Telnet and SSH require software on the user's device, but they rely on the existing TCP/IP network to transmit data. The next few pages detail how to connect the console and set up the software for each method to access the CLI.

Cabling the Console Connection

The physical console connection, both old and new, uses three main components: the physical console port on the switch, a physical serial port on the PC, and a cable that works with the console and serial ports. However, the physical cabling details have changed slowly over time, mainly because of advances and changes with serial interfaces on PC hardware. For this next topic, the text looks at three cases: newer connectors on both the PC and the switch, older connectors on both, and a third case with the newer (USB) connector on the PC but with an older connector on the switch.

Most PCs today use a familiar standard USB cable for the console connection. Cisco has been including USB ports as console ports in newer routers and switches as well. All you have to do is look at the switch to make sure you have the correct style of USB cable end to match the USB console port. In the simplest form, you can use any USB port on the PC, with a USB cable, connected to the USB console port on the switch or router, as shown on the far right side of Figure 4-3.

Key Topic

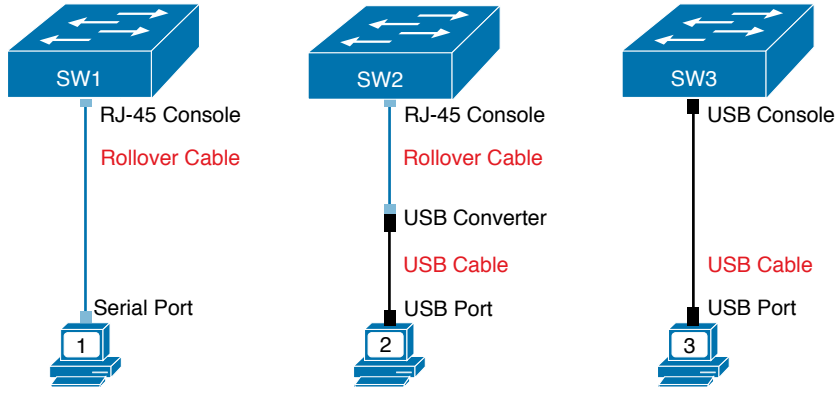


Figure 4-3 *Console Connection to a Switch*

Older console connections use a PC serial port that pre-dates USB, a UTP cable, and an RJ-45 console port on the switch, as shown on the left side of Figure 4-3. The PC serial port typically has a D-shell connector (roughly rectangular) with nine pins (often called a DB-9). The console port looks like any Ethernet RJ-45 port (but is typically colored in blue and with the word *console* beside it on the switch).

The cabling for this older-style console connection can be simple or require some effort, depending on what cable you use. You can use the purpose-built console cable that ships with new Cisco switches and routers and not think about the details. However, you can make

---

Answers to the "Do I Know This Already?" quiz:

1 A, B **2** B **3** B **4** A **5** F **6** B, C

your own cable with a standard serial cable (with a connector that matches the PC), a standard RJ-45 to DB-9 converter plug, and a UTP cable. However, the UTP cable does not use the same pinouts as Ethernet; instead, the cable uses rollover cable pinouts rather than any of the standard Ethernet cabling pinouts. The rollover pinout uses eight wires, rolling the wire at pin 1 to pin 8, pin 2 to pin 7, pin 3 to pin 6, and so on.

As it turns out, USB ports became common on PCs before Cisco began commonly using USB for its console ports. So, you also have to be ready to use a PC that has only a USB port and not an old serial port, but a router or switch that has the older RJ-45 console port (and no USB console port). The center of Figure 4-3 shows that case. To connect such a PC to a router or switch console, you need a USB converter that converts from the older console cable to a USB connector, and a rollover UTP cable, as shown in the middle of Figure 4-3.

NOTE When using the USB options, you typically also need to install a software driver so that your PC's OS knows that the device on the other end of the USB connection is the console of a Cisco device. Also, you can easily find photos of these cables and components online, with searches like "cisco console cable," "cisco usb console cable," or "console cable converter."

The 2960-XR series, for instance, supports both the older RJ-45 console port and a USB console port. Figure 4-4 points to the two console ports; you would use only one or the other. Note that the USB console port uses a mini-B port rather than the more commonly seen rectangular standard USB Type A port.

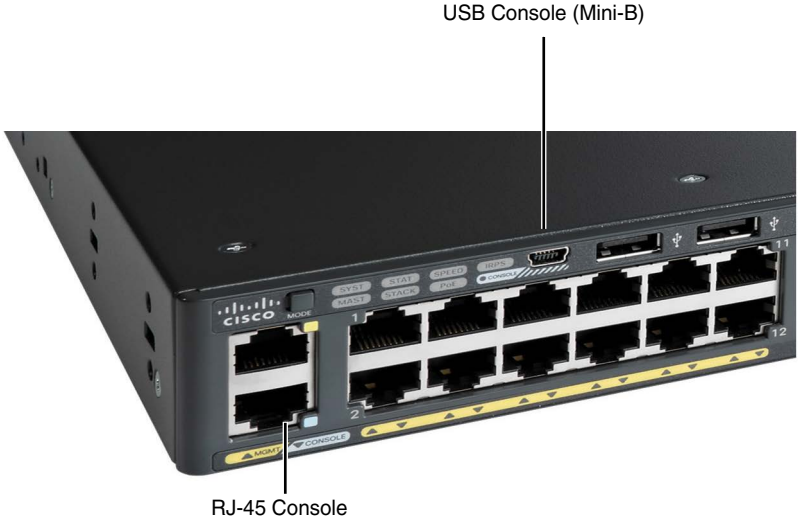


Figure 4-4 *A Part of a 2960-XR Switch with Console Ports Shown*

After the PC is physically connected to the console port, a terminal emulator software package must be installed and configured on the PC. The terminal emulator software treats all data as text. It accepts the text typed by the user and sends it over the console connection to the switch. Similarly, any bits coming into the PC over the console connection are displayed as text for the user to read.

The emulator must be configured to use the PC's serial port to match the settings on the switch's console port settings. The default console port settings on a switch are as follows. Note that the last three parameters are referred to collectively as 8N1:

Key Topic

- 9600 bits/second
- No hardware flow control
- 8-bit ASCII
- No parity bits
- 1 stop bit

Figure 4-5 shows one such terminal emulator. The image shows the window created by the emulator software in the background, with some output of a **show** command. The foreground, in the upper right, shows a settings window that lists the default console settings as listed just before this paragraph.

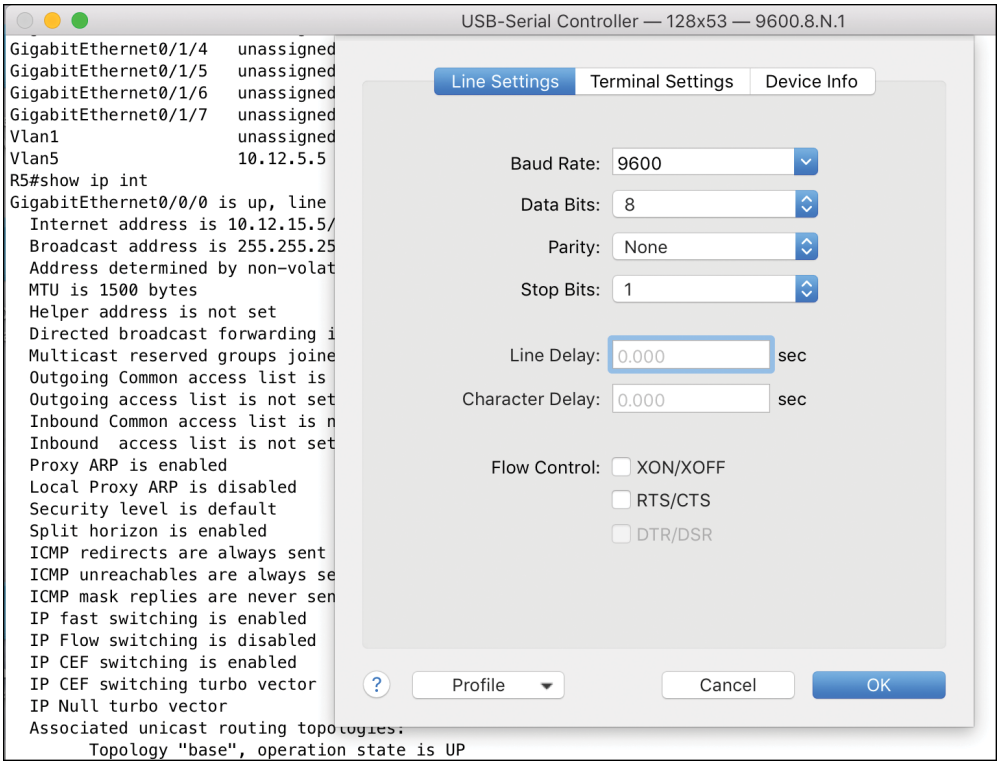


Figure 4-5 *Terminal Settings for Console Access*

Accessing the CLI with Telnet and SSH

For many years, terminal emulator applications have supported far more than the ability to communicate over a serial port to a local device (like a switch's console). Terminal emulators support a variety of TCP/IP applications as well, including Telnet and SSH. Telnet and SSH both allow the user to connect to another device's CLI, but instead of connecting through

a console cable to the console port, the traffic flows over the same IP network that the networking devices are helping to create.

Telnet uses the concept of a Telnet client (the terminal application) and a Telnet server (the switch in this case). A *Telnet client*, the device that sits in front of the user, accepts keyboard input and sends those commands to the *Telnet server*. The Telnet server accepts the text, interprets the text as a command, and replies back.

Cisco Catalyst switches enable a Telnet server by default, but switches need a few more configuration settings before you can successfully use Telnet to connect to a switch. Chapter 6, "Configuring Basic Switch Management," covers switch configuration to support Telnet and SSH in detail.

Using Telnet in a lab today makes sense, but Telnet poses a significant security risk in production networks. Telnet sends all data (including any username and password for login to the switch) as clear-text data. SSH gives us a much better option.

Think of SSH as the much more secure Telnet cousin. Outwardly, you still open a terminal emulator, connect to the switch's IP address, and see the switch CLI, no matter whether you use Telnet or SSH. The differences exist behind the scenes: SSH encrypts the contents of all messages, including the passwords, avoiding the possibility of someone capturing packets in the network and stealing the password to network devices.

User and Enable (Privileged) Modes

All three CLI access methods covered so far (console, Telnet, and SSH) place the user in an area of the CLI called *user EXEC mode*. User EXEC mode, sometimes also called *user mode*, allows the user to look around but not break anything. The "EXEC mode" part of the name refers to the fact that in this mode, when you enter a command, the switch executes the command and then displays messages that describe the command's results.

NOTE If you have not used the CLI before, you might want to experiment with the CLI from the Sim Lite product, or view the video about CLI basics. You can find these resources on the companion website as mentioned in the Introduction.

Cisco IOS supports a more powerful EXEC mode called *enable* mode (also known as *privileged* mode or *privileged EXEC* mode). Enable mode gets its name from the **enable** command, which moves the user from user mode to enable mode, as shown in Figure 4-6. The other name for this mode, *privileged mode*, refers to the fact that powerful (or privileged) commands can be executed there. For example, you can use the **reload** command, which tells the switch to reinitialize or reboot Cisco IOS, only from enable mode.

NOTE If the command prompt lists the hostname followed by a >, the user is in user mode; if it is the hostname followed by the #, the user is in enable mode.

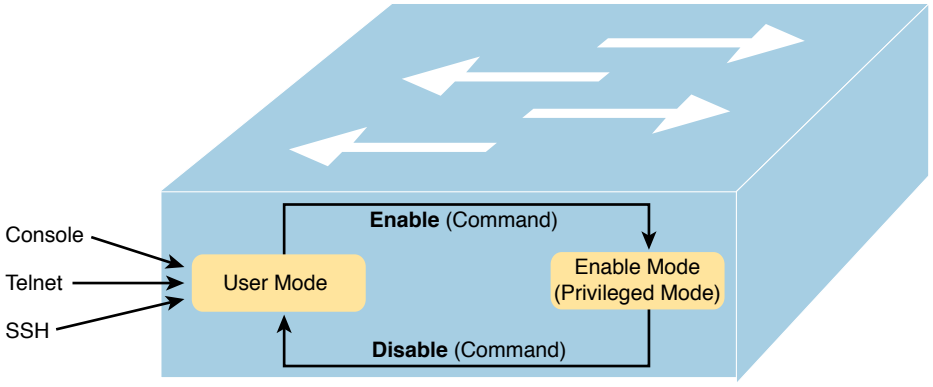


Figure 4-6 *User and Privileged Modes*

Example 4-1 demonstrates the differences between user and enable modes. The example shows the output that you could see in a terminal emulator window, for instance, when connecting from the console. In this case, the user sits at the user mode prompt ("Certskills1>") and tries the **reload** command. The **reload** command tells the switch to reinitialize or reboot Cisco IOS, so IOS allows this powerful command to be used only from enable mode. IOS rejects the **reload** command when used in user mode. Then the user moves to enable mode—also called privileged mode—(using the **enable** EXEC command). At that point, IOS accepts the **reload** command now that the user is in enable mode.

Example 4-1 *Example of Privileged Mode Commands Being Rejected in User Mode*

```
Press RETURN to get started.

User Access Verification

Password:
Certskills1>
Certskills1> reload
Translating "reload"
% Unknown command or computer name, or unable to find computer address
Certskills1> enable
Password:
Certskills1#
Certskills1# reload

Proceed with reload? [confirm] y
00:08:42: %SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload Command.
```

NOTE The commands that can be used in either user (EXEC) mode or enable (EXEC) mode are called EXEC commands.

This example is the first instance of this book showing you the output from the CLI, so it is worth noting a few conventions. The bold text represents what the user typed, and the non-bold text is what the switch sent back to the terminal emulator. Also, the typed passwords do not show up on the screen for security purposes. Finally, note that this switch has been preconfigured with a hostname of Certskills1, so the command prompt on the left shows that hostname on each line.

Password Security for CLI Access from the Console

A Cisco switch, with default settings, remains relatively secure when locked inside a wiring closet, because by default, a switch allows console access only. By default, the console requires no password at all, and no password to reach enable mode for users that happened to connect from the console. The reason is that if you have access to the physical console port of the switch, you already have pretty much complete control over the switch. You could literally get out your screwdriver and walk off with it, or you could unplug the power, or follow well-published procedures to go through password recovery to break into the CLI and then configure anything you want to configure.

However, many people go ahead and set up simple password protection for console users. Simple passwords can be configured at two points in the login process from the console: when the user connects from the console, and when any user moves to enable mode (using the **enable** EXEC command). You may have noticed that back in Example 4-1, the user saw a password prompt at both points.

Example 4-2 shows the additional configuration commands that were configured prior to collecting the output in Example 4-1. The output holds an excerpt from the EXEC command **show running-config**, which lists the current configuration in the switch.

Example 4-2 *Nondefault Basic Configuration*

```
Certskills1# show running-config
! Output has been formatted to show only the parts relevant to this discussion
hostname Certskills1
!
enable secret love
!
line console 0
 login
 password faith
! The rest of the output has been omitted
Certskills1#
```

Working from top to bottom, note that the first configuration command listed by the **show running-config** command sets the switch's hostname to Certskills1. You might have noticed that the command prompts in Example 4-1 all began with Certskills1, and that's why the command prompt begins with the hostname of the switch.

Next, note that the lines with a **!** in them are comment lines, both in the text of this book and in the real switch CLI.

The **enable secret love** configuration command defines the password that all users must use to reach enable mode. So, no matter whether users connect from the console, Telnet, or SSH, they would use the password love when prompted for a password after typing the **enable** EXEC command.

Finally, the last three lines configure the console password. The first line (**line console 0**) is the command that identifies the console, basically meaning "these next commands apply to the console only." The **login** command tells IOS to perform simple password checking (at the console). Remember, by default, the switch does not ask for a password for console users. Finally, the **password faith** command defines the password the console user must type when prompted.

This example just scratches the surface of the kinds of security configuration you might choose to configure on a switch, but it does give you enough detail to configure switches in your lab and get started (which is the reason I put these details in this first chapter of Part II). Note that Chapter 6 shows the configuration steps to add support for Telnet and SSH (including password security), and Chapter 5 of the *CCNA 200-301 Official Cert Guide, Volume 2*, "Securing Network Devices," shows additional security configuration as well.

CLI Help Features

If you printed the Cisco IOS Command Reference documents, you would end up with a stack of paper several feet tall. No one should expect to memorize all the commands—and no one does. You can use several very easy, convenient tools to help remember commands and save time typing. As you progress through your Cisco certifications, the exams will cover progressively more commands. However, you should know the methods of getting command help.

Table 4-2 summarizes command-recall help options available at the CLI. Note that, in the first column, *command* represents any command. Likewise, *parm* represents a command's parameter. For example, the second row lists *command* **?**, which means that commands such as **show ?** and **copy ?** would list help for the **show** and **copy** commands, respectively.

Table 4-2 Cisco IOS Software Command Help

What You Enter	What Help You Get
?	Provides help for all commands available in this mode.
command **?**	With a space between the command and the **?**, the switch lists text to describe all the first parameter options for the command.
*com***?**	Lists commands that start with **com**.
*command parm***?**	Lists all parameters beginning with the **parameter typed so far**. (Notice that there is no space between *parm* and the **?**.)
command parm<Tab>	Pressing the Tab key causes IOS to spell out the rest of the word, assuming that you have typed enough of the word so there is only one option that begins with that string of characters.
command parm1 **?**	If a space is inserted before the question mark, the CLI lists all the next parameters and gives a brief explanation of each.

When you enter the **?**, the Cisco IOS CLI reacts immediately; that is, you don't need to press the Enter key or any other keys. The device running Cisco IOS also redisplays what you entered before the **?** to save you some keystrokes. If you press Enter immediately after the **?**, Cisco IOS tries to execute the command with only the parameters you have entered so far.

The information supplied by using help depends on the CLI mode. For example, when **?** is entered in user mode, the commands allowed in user mode are displayed, but commands available only in enable mode (not in user mode) are not displayed. Also, help is available in configuration mode, which is the mode used to configure the switch. In fact, configuration mode has many different subconfiguration modes, as explained in the section "Configuration Submodes and Contexts," later in this chapter. So, you can get help for the commands available in each configuration submode as well. (Note that this might be a good time to use the free Sim Lite product on the companion website—open any lab, use the question mark, and try some commands.)

Cisco IOS stores the commands that you enter in a history buffer, storing ten commands by default. The CLI allows you to move backward and forward in the historical list of commands and then edit the command before reissuing it. These key sequences can help you use the CLI more quickly on the exams. Table 4-3 lists the commands used to manipulate previously entered commands.

Table 4-3 Key Sequences for Command Edit and Recall

Keyboard Command	What Happens
Up arrow or Ctrl+P	This displays the most recently used command. If you press it again, the next most recent command appears, until the history buffer is exhausted. (The *P* stands for previous.)
Down arrow or Ctrl+N	If you have gone too far back into the history buffer, these keys take you forward to the more recently entered commands. (The *N* stands for next.)
Left arrow or Ctrl+B	This moves the cursor backward in the currently displayed command without deleting characters. (The *B* stands for back.)
Right arrow or Ctrl+F	This moves the cursor forward in the currently displayed command without deleting characters. (The *F* stands for forward.)
Backspace	This moves the cursor backward in the currently displayed command, deleting characters.

The debug and show Commands

By far, the single most popular Cisco IOS command is the **show** command. The **show** command has a large variety of options, and with those options, you can find the status of almost every feature of Cisco IOS. Essentially, the **show** command lists the currently known facts about the switch's operational status. The only work the switch does in reaction to **show** commands is to find the current status and list the information in messages sent to the user.

For example, consider the output from the **show mac address-table dynamic** command listed in Example 4-3. This **show** command, issued from user mode, lists the table the switch uses to make forwarding decisions. A switch's MAC address table basically lists the data a switch uses to do its primary job.

Example 4-3 *Nondefault Basic Configuration*

```
Certskills1> show mac address-table dynamic
 Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  31    0200.1111.1111    DYNAMIC     Gi0/1
  31    0200.3333.3333    DYNAMIC     Fa0/3
  31    1833.9d7b.0e9a    DYNAMIC     Gi0/1
  10    1833.9d7b.0e9a    DYNAMIC     Gi0/1
  10    30f7.0d29.8561    DYNAMIC     Gi0/1
   1    1833.9d7b.0e9a    DYNAMIC     Gi0/1
  12    1833.9d7b.0e9a    DYNAMIC     Gi0/1
Total Mac Addresses for this criterion: 7
Certskills1>
```

The **debug** command also tells the user details about the operation of the switch. However, while the **show** command lists status information at one instant of time—more like a photograph—the **debug** command acts more like a live video camera feed. Once you issue a **debug** command, IOS remembers, issuing messages that any switch user can choose to see. The console sees these messages by default. Most of the commands used throughout this book to verify operation of switches and routers are **show** commands.

Configuring Cisco IOS Software

You will want to configure every switch in an Enterprise network, even though the switches will forward traffic even with default configuration. This section covers the basic configuration processes, including the concept of a configuration file and the locations in which the configuration files can be stored. Although this section focuses on the configuration process, and not on the configuration commands themselves, you should know all the commands covered in this chapter for the exams, in addition to the configuration processes.

C*onfiguration mode* is another mode for the Cisco CLI, similar to user mode and privileged mode. User mode lets you issue nondisruptive commands and displays some information. Privileged mode supports a superset of commands compared to user mode, including commands that might disrupt switch operations. However, not one of the commands in user or privileged mode changes the switch's configuration. Configuration mode accepts *configuration commands*—commands that tell the switch the details of what to do and how to do it. Figure 4-7 illustrates the relationships among configuration mode, user EXEC mode, and privileged EXEC mode.

Key Topic

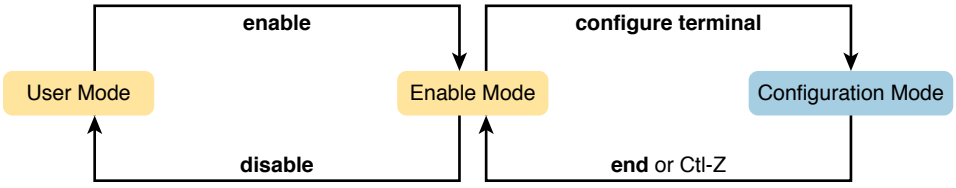


Figure 4-7 *CLI Configuration Mode Versus EXEC Modes*

Commands entered in configuration mode update the active configuration file. *These changes to the configuration occur immediately each time you press the Enter key at the end of a command.* Be careful when you enter a configuration command!

Configuration Submodes and Contexts

Configuration mode itself contains a multitude of commands. To help organize the configuration, IOS groups some kinds of configuration commands together. To do that, when using configuration mode, you move from the initial mode—global configuration mode—into subcommand modes. *Context-setting commands* move you from one configuration subcommand mode, or context, to another. These context-setting commands tell the switch the topic about which you will enter the next few configuration commands. More importantly, the context tells the switch the topic you care about right now, so when you use the **?** to get help, the switch gives you help about that topic only.

NOTE *Context-setting* is not a Cisco term. It is just a description used here to help make sense of configuration mode.

The best way to learn about configuration submodes is to use them, but first, take a look at these upcoming examples. For instance, the **interface** command is one of the most commonly used context-setting configuration commands. For example, the CLI user could enter interface configuration mode by entering the **interface FastEthernet 0/1** configuration command. Asking for help in interface configuration mode displays only commands that are useful when configuring Ethernet interfaces. Commands used in this context are called *subcommands*—or, in this specific case, *interface subcommands*. When you begin practicing with the CLI with real equipment, the navigation between modes can become natural. For now, consider Example 4-4, which shows the following:

- Movement from enable mode to global configuration mode by using the **configure terminal** EXEC command
- Using a **hostname Fred** global configuration command to configure the switch's name
- Movement from global configuration mode to console line configuration mode (using the **line console 0** command)
- Setting the console's simple password to **hope** (using the **password hope** line subcommand)
- Movement from console configuration mode to interface configuration mode (using the **interface** *type number* command)

- Setting the speed to 100 Mbps for interface Fa0/1 (using the **speed 100** interface subcommand)
- Movement from interface configuration mode back to global configuration mode (using the **exit** command)

Example 4-4 *Navigating Between Different Configuration Modes*

```
Switch# configure terminal
Switch(config)# hostname Fred
Fred(config)# line console 0
Fred(config-line)# password hope
Fred(config-line)# interface FastEthernet 0/1
Fred(config-if)# speed 100
Fred(config-if)# exit
Fred(config)#
```

The text inside parentheses in the command prompt identifies the configuration mode. For example, the first command prompt after you enter configuration mode lists (config), meaning global configuration mode. After the **line console 0** command, the text expands to (config-line), meaning line configuration mode. Each time the command prompt changes within config mode, you have moved to another configuration mode.

Table 4-4 shows the most common command prompts in configuration mode, the names of those modes, and the context-setting commands used to reach those modes.

Key Topic

Table 4-4 Common Switch Configuration Modes

Prompt	Name of Mode	Context-Setting Command(s) to Reach This Mode
hostname(config)#	Global	None—first mode after **configure terminal**
hostname(config-line)#	Line	**line console 0** **line vty 0 15**
hostname(config-if)#	Interface	**interface** *type number*
hostname(vlan)#	VLAN	**vlan** *number*

You should practice until you become comfortable moving between the different configuration modes, back to enable mode, and then back into the configuration modes. However, you can learn these skills just doing labs about the topics in later chapters of the book. For now, Figure 4-8 shows most of the navigation between global configuration mode and the four configuration submodes listed in Table 4-4.

NOTE You can also move directly from one configuration submode to another, without first using the **exit** command to move back to global configuration mode. Just use the commands listed in bold in the center of the figure.

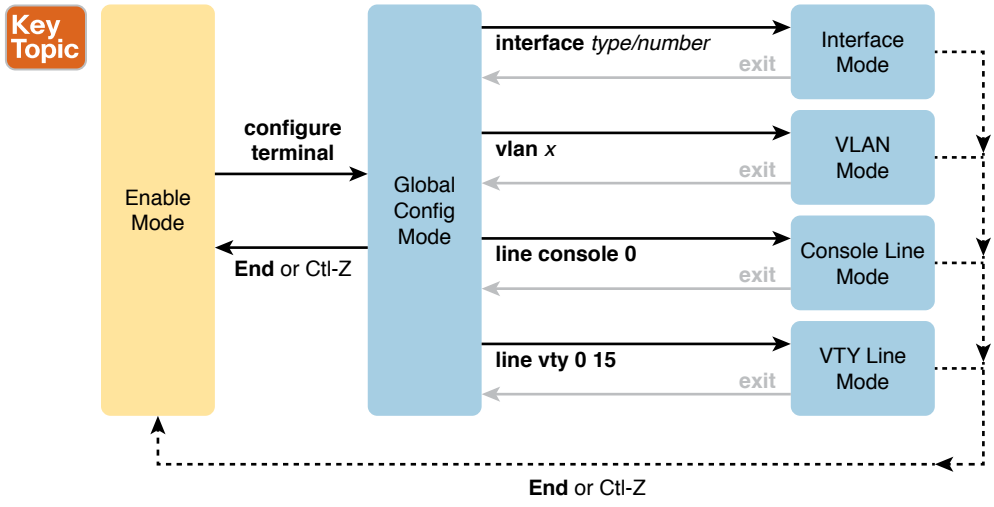


Figure 4-8 *Navigation In and Out of Switch Configuration Modes*

You really should stop and try navigating around these configuration modes. If you have not yet decided on a lab strategy, install the Pearson Sim Lite software from the companion website. It includes the simulator and a couple of lab exercises. Start any lab, ignore the instructions, and just get into configuration mode and move around between the configuration modes shown in Figure 4-8.

No set rules exist for what commands are global commands or subcommands. Generally, however, when multiple instances of a parameter can be set in a single switch, the command used to set the parameter is likely a configuration subcommand. Items that are set once for the entire switch are likely global commands. For example, the **hostname** command is a global command because there is only one hostname per switch. Conversely, the **speed** command is an interface subcommand that applies to each switch interface that can run at different speeds, so it is a subcommand, applying to the particular interface under which it is configured.

Storing Switch Configuration Files

When you configure a switch, it needs to use the configuration. It also needs to be able to retain the configuration in case the switch loses power. Cisco switches contain random-access memory (RAM) to store data while Cisco IOS is using it, but RAM loses its contents when the switch loses power or is reloaded. To store information that must be retained when the switch loses power or is reloaded, Cisco switches use several types of more permanent memory, none of which has any moving parts. By avoiding components with moving parts (such as traditional disk drives), switches can maintain better uptime and availability.

The following list details the four main types of memory found in Cisco switches, as well as the most common use of each type:

- **RAM:** Sometimes called DRAM, for dynamic random-access memory, RAM is used by the switch just as it is used by any other computer: for working storage. The running (active) configuration file is stored here.

- **Flash memory:** Either a chip inside the switch or a removable memory card, flash memory stores fully functional Cisco IOS images and is the default location where the switch gets its Cisco IOS at boot time. Flash memory also can be used to store any other files, including backup copies of configuration files.
- **ROM:** Read-only memory (ROM) stores a bootstrap (or boothelper) program that is loaded when the switch first powers on. This bootstrap program then finds the full Cisco IOS image and manages the process of loading Cisco IOS into RAM, at which point Cisco IOS takes over operation of the switch.
- **NVRAM:** Nonvolatile RAM (NVRAM) stores the initial or startup configuration file that is used when the switch is first powered on and when the switch is reloaded.

Figure 4-9 summarizes this same information in a briefer and more convenient form for memorization and study.

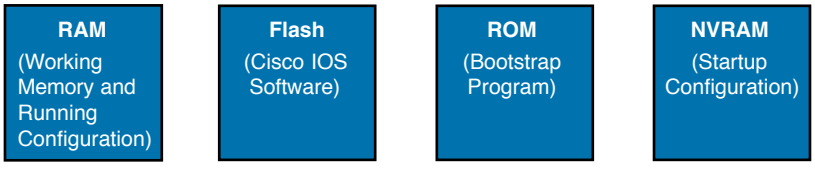


Figure 4-9 *Cisco Switch Memory Types*

Cisco IOS stores the collection of configuration commands in a *configuration file*. In fact, switches use multiple configuration files—one file for the initial configuration used when powering on, and another configuration file for the active, currently used running configuration as stored in RAM. Table 4-5 lists the names of these two files, their purpose, and their storage location.

Key Topic

Table 4-5 Names and Purposes of the Two Main Cisco IOS Configuration Files

Configuration Filename	Purpose	Where It Is Stored
startup-config	Stores the initial configuration used anytime the switch reloads Cisco IOS.	NVRAM
running-config	Stores the currently used configuration commands. This file changes dynamically when someone enters commands in configuration mode.	RAM

Essentially, when you use configuration mode, you change only the running-config file. This means that the configuration example earlier in this chapter (Example 4-4) updates only the running-config file. However, if the switch lost power right after that example, all that configuration would be lost. If you want to keep that configuration, you have to copy the running-config file into NVRAM, overwriting the old startup-config file.

Example 4-5 demonstrates that commands used in configuration mode change only the running configuration in RAM. The example shows the following concepts and steps:

Step 1. The example begins with both the running and startup-config having the same hostname, per the **hostname hannah** command.

Step 2. The hostname is changed in configuration mode using the **hostname harold** command.

Step 3. The **show running-config** and **show startup-config** commands show the fact that the hostnames are now different, with the **hostname harold** command found only in the running-config.

Example 4-5 *How Configuration Mode Commands Change the Running-Config File, Not the Startup-Config File*

```
! Step 1 next (two commands)
!
hannah# show running-config
! (lines omitted)
hostname hannah
! (rest of lines omitted)

hannah# show startup-config
! (lines omitted)
hostname hannah
! (rest of lines omitted)
! Step 2 next. Notice that the command prompt changes immediately after
! the hostname command.

hannah# configure terminal
hannah(config)# hostname harold
harold(config)# exit
! Step 3 next (two commands)
!
harold# show running-config
! (lines omitted) - just showing the part with the hostname command
hostname harold
!
harold# show startup-config
! (lines omitted) - just showing the part with the hostname command
hostname hannah
```

Copying and Erasing Configuration Files

The configuration process updates the running-config file, which is lost if the router loses power or is reloaded. Clearly, IOS needs to provide us a way to copy the running configuration so that it will not be lost, so it will be used the next time the switch reloads or powers on. For instance, Example 4-5 ended with a different running configuration (with the **hostname harold** command) versus the startup configuration.

In short, the EXEC command **copy running-config startup-config** backs up the running-config to the startup-config file. This command overwrites the current startup-config file with what is currently in the running-configuration file.

In addition, in the lab, you may want to just get rid of all existing configuration and start over with a clean configuration. To do that, you can erase the startup-config file using three different commands:

```
write erase
erase startup-config
erase nvram:
```

Once the startup-config file is erased, you can reload or power off/on the switch, and it will boot with the now-empty startup configuration.

Note that Cisco IOS does not have a command that erases the contents of the running-config file. To clear out the running-config file, simply erase the startup-config file, and then **reload** the switch, and the running-config will be empty at the end of the process.

NOTE Cisco uses the term *reload* to refer to what most PC operating systems call rebooting or restarting. In each case, it is a re-initialization of the software. The **reload** EXEC command causes a switch to reload.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or on the book's companion website. Refer to the "Your Study Plan" element section titled "Step 2: Build Your Study Habits Around the Chapter" for more details. Table 4-6 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 4-6 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review memory tables		Book, website
Review command tables		Book

Review All the Key Topics

Key Topic

Table 4-7 Key Topics for Chapter 4

Key Topic Element	Description	Page Number
Figure 4-2	Three methods to access a switch CLI	87
Figure 4-3	Cabling options for a console connection	88
List	A Cisco switch's default console port settings	90
Figure 4-7	Navigation between user, enable, and global config modes	97
Table 4-4	A list of configuration mode prompts, the name of the configuration mode, and the command used to reach each mode	98
Figure 4-8	Configuration mode context-setting commands	99
Table 4-5	The names and purposes of the two configuration files in a switch or router	100

Key Terms You Should Know

command-line interface (CLI), Telnet, Secure Shell (SSH), enable mode, user mode, configuration mode, startup-config file, running-config file

Command References

Tables 4-8 and 4-9 list configuration and verification commands used in this chapter, respectively. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 4-8 Chapter 4 Configuration Commands

Command	Mode and Purpose
line console 0	Global command that changes the context to console configuration mode.
login	Line (console and vty) configuration mode. Tells IOS to prompt for a password (no username).
password *pass-value*	Line (console and vty) configuration mode. Sets the password required on that line for login if the **login** command (with no other parameters) is also configured.
interface *type port-number*	Global command that changes the context to interface mode—for example, **interface FastEthernet 0/1**.
hostname *name*	Global command that sets this switch's hostname, which is also used as the first part of the switch's command prompt.
exit	Moves back to the next higher mode in configuration mode.

Command	Mode and Purpose
end	Exits configuration mode and goes back to enable mode from any of the configuration submodes.
Ctrl+Z	This is not a command, but rather a two-key combination (pressing the Ctrl key and the letter Z) that together do the same thing as the **end** command.

Table 4-9 Chapter 4 EXEC Command Reference

Command	Purpose
no debug all **undebug all**	Enable mode EXEC command to disable all currently enabled debugs.
reload	Enable mode EXEC command that reboots the switch or router.
copy running-config startup-config	Enable mode EXEC command that saves the active config, replacing the startup-config file used when the switch initializes.
copy startup-config running-config	Enable mode EXEC command that merges the startup-config file with the currently active config file in RAM.
show running-config	Lists the contents of the running-config file.
write erase **erase startup-config** **erase nvram:**	These enable mode EXEC commands erase the startup-config file.
quit	EXEC command that disconnects the user from the CLI session.
show startup-config	Lists the contents of the startup-config (initial config) file.
enable	Moves the user from user mode to enable (privileged) mode and prompts for a password if one is configured.
disable	Moves the user from enable mode to user mode.
configure terminal	Enable mode command that moves the user into configuration mode.

This page intentionally left blank

CHAPTER 5

Analyzing Ethernet LAN Switching

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.b L2 and L3 Switches

1.13 Describe switching concepts

1.13.a MAC learning and aging

1.13.b Frame switching

1.13.c Frame flooding

1.13.d MAC address table

2.0 Network Access

2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations

When you buy a Cisco Catalyst Ethernet switch, the switch is ready to work. All you have to do is take it out of the box, power on the switch by connecting the power cable to the switch and a power outlet, and connect hosts to the switch using the correct unshielded twisted-pair (UTP) cables. You do not have to configure anything else, or connect to the console and login, or do anything: the switch just starts forwarding Ethernet frames.

In Part II of this book, you will learn how to build, configure, and verify the operation of Ethernet LANs. In Chapter 4, "Using the Command-Line Interface," you learned how to move around in the CLI, issue commands, and configure the switch. This chapter takes a short but important step in that journey by explaining the logic a switch uses when forwarding Ethernet frames.

This chapter breaks the content into two major sections. The first reviews and then further develops the concepts behind LAN switching, which were first introduced back in Chapter 2, "Fundamentals of Ethernet LANs." The second section then uses IOS show commands to verify that Cisco switches actually learned the MAC addresses, built the MAC address table, and forwarded frames.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 5-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
LAN Switching Concepts	1–4
Verifying and Analyzing Ethernet Switching	5–6

1. Which of the following statements describes part of the process of how a switch decides to forward a frame destined for a known unicast MAC address?
   a. It compares the unicast destination address to the bridging, or MAC address, table.
   b. It compares the unicast source address to the bridging, or MAC address, table.
   c. It forwards the frame out all interfaces in the same VLAN except for the incoming interface.
   d. It compares the destination IP address to the destination MAC address.
   e. It compares the frame's incoming interface to the source MAC entry in the MAC address table.
2. Which of the following statements describes part of the process of how a LAN switch decides to forward a frame destined for a broadcast MAC address?
   a. It compares the unicast destination address to the bridging, or MAC address, table.
   b. It compares the unicast source address to the bridging, or MAC address, table.
   c. It forwards the frame out all interfaces in the same VLAN except for the incoming interface.
   d. It compares the destination IP address to the destination MAC address.
   e. It compares the frame's incoming interface to the source MAC entry in the MAC address table.
3. Which of the following statements best describes what a switch does with a frame destined for an unknown unicast address?
   a. It forwards out all interfaces in the same VLAN except for the incoming interface.
   b. It forwards the frame out the one interface identified by the matching entry in the MAC address table.
   c. It compares the destination IP address to the destination MAC address.
   d. It compares the frame's incoming interface to the source MAC entry in the MAC address table.
4. Which of the following comparisons does a switch make when deciding whether a new MAC address should be added to its MAC address table?
   a. It compares the unicast destination address to the bridging, or MAC address, table.
   b. It compares the unicast source address to the bridging, or MAC address, table.
   c. It compares the VLAN ID to the bridging, or MAC address, table.
   d. It compares the destination IP address's ARP cache entry to the bridging, or MAC address, table.

5. A Cisco Catalyst switch has 24 10/100 ports, numbered 0/1 through 0/24. Ten PCs connect to the 10 lowest numbered ports, with those PCs working and sending data over the network. The other ports are not connected to any device. Which of the following answers lists facts displayed by the show interfaces status command?
   a. Port Ethernet 0/1 is in a connected state.
   b. Port Fast Ethernet 0/11 is in a connected state.
   c. Port Fast Ethernet 0/5 is in a connected state.
   d. Port Ethernet 0/15 is in a notconnected state.

6. Consider the following output from a Cisco Catalyst switch:

```
SW1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address        Type         Ports
----    -----------        --------     -----
   1    02AA.AAAA.AAAA     DYNAMIC      Gi0/1
   1    02BB.BBBB.BBBB     DYNAMIC      Gi0/2
   1    02CC.CCCC.CCCC     DYNAMIC      Gi0/3
Total Mac Addresses for this criterion: 3
```

   Which of the following answers is true about this switch?
   a. The output proves that port Gi0/2 connects directly to a device that uses address 02BB.BBBB.BBBB.
   b. The switch has learned three MAC addresses since the switch powered on.
   c. The three listed MAC addresses were learned based on the destination MAC address of frames forwarded by the switch.
   d. 02CC.CCCC.CCCC was learned from the source MAC address of a frame that entered port Gi0/3.

Foundation Topics

LAN Switching Concepts

A modern Ethernet LAN connects user devices as well as servers into some switches, with the switches then connecting to each other, sometimes in a design like Figure 5-1. Part of the LAN, called a campus LAN, supports the end-user population as shown on the left of the figure. End-user devices connect to LAN switches, which in turn connect to other switches so that a path exists to the rest of the network. The campus LAN switches sit in wiring closets close to the end users. On the right, the servers used to provide information to the users also connect to the LAN. Those servers and switches often sit in a closed room called a *data center*, with connections to the campus LAN to support traffic to/from the users.

To forward traffic from a user device to a server and back, each switch performs the same kind of logic, independently from each other. The first half of this chapter examines the logic: how a switch chooses to forward an Ethernet frame, when the switch chooses to not forward the frame, and so on.

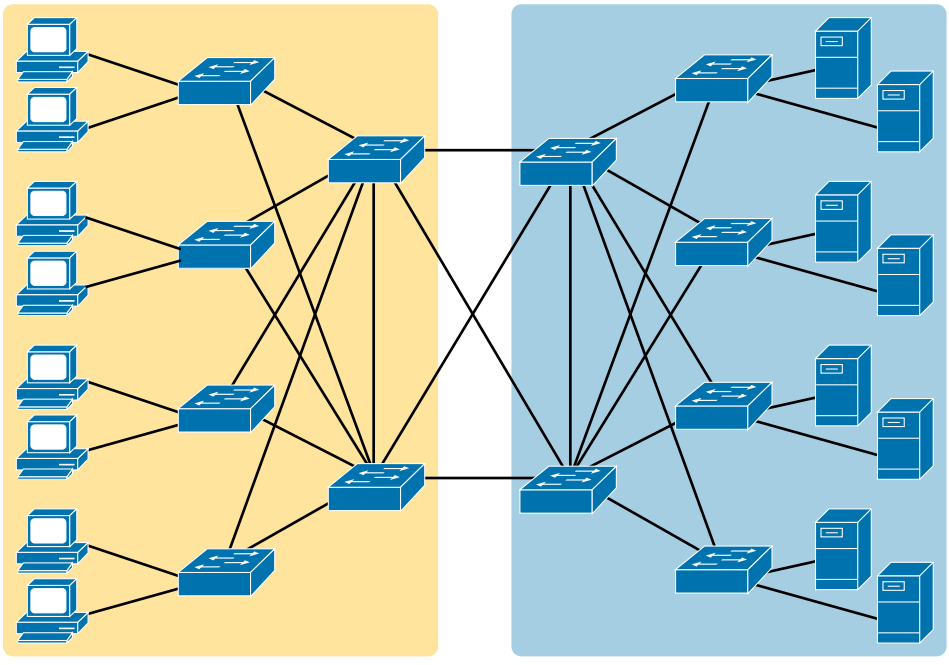


Figure 5-1 *Campus LAN and Data Center LAN, Conceptual Drawing*

Overview of Switching Logic

Ultimately, the role of a LAN switch is to forward Ethernet frames. LANs exist as a set of user devices, servers, and other devices that connect to switches, with the switches connected to each other. The LAN switch has one primary job: to forward frames to the correct destination (MAC) address. And to achieve that goal, switches use logic—logic based on the source and destination MAC address in each frame's Ethernet header.

LAN switches receive Ethernet frames and then make a switching decision: either forward the frame out some other ports or ignore the frame. To accomplish this primary mission, switches perform three actions:

Key Topic

1. Deciding when to forward a frame or when to filter (not forward) a frame, based on the destination MAC address
2. Preparing to forward frames by learning MAC addresses by examining the source MAC address of each frame received by the switch
3. Preparing to forward only one copy of the frame to the destination by creating a (Layer 2) loop-free environment with other switches by using Spanning Tree Protocol (STP)

The first action is the switch's primary job, whereas the other two items are overhead functions.

NOTE Throughout this book's discussion of LAN switches, the terms *switch port* and *switch interface* are synonymous.

Although Chapter 2's section titled "Ethernet Data-Link Protocols" already discussed the frame format, this discussion of Ethernet switching is pretty important, so reviewing the Ethernet frame at this point might be helpful. Figure 5-2 shows one popular format for an Ethernet frame. Basically, a switch would take the frame shown in the figure, make a decision of where to forward the frame, and send the frame out that other interface.

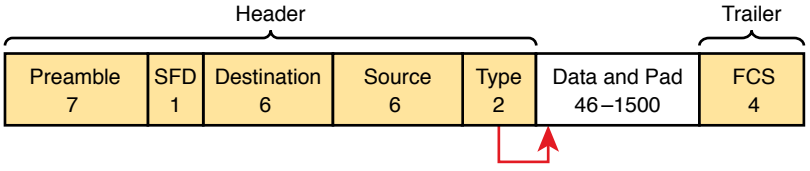


Figure 5-2 *IEEE 802.3 Ethernet Frame (One Variation)*

Most of the upcoming discussions and figures about Ethernet switching focus on the use of the destination and source MAC address fields in the header. All Ethernet frames have both a destination and source MAC address. Both are 6-bytes long (represented as 12 hex digits in the book) and are a key part of the switching logic discussed in this section. Refer back to Chapter 2's discussion of the header in detail for more info on the rest of the Ethernet frame.

NOTE The companion website includes a video that explains the basics of Ethernet switching.

Now on to the details of how Ethernet switching works!

Forwarding Known Unicast Frames

To decide whether to forward a frame, a switch uses a dynamically built table that lists MAC addresses and outgoing interfaces. Switches compare the frame's destination MAC address to this table to decide whether the switch should forward a frame or simply ignore it. For example, consider the simple network shown in Figure 5-3, with Fred sending a frame to Barney.

In this figure, Fred sends a frame with destination address 0200.2222.2222 (Barney's MAC address). The switch compares the destination MAC address (0200.2222.2222) to the MAC address table, matching the bold table entry. That matched table entry tells the switch to forward the frame out port F0/2, and only port F0/2.

Answers to the "Do I Know This Already?" quiz:

1 A **2** C **3** A **4** B **5** C **6** D

NOTE A switch's MAC address table is also called the *switching table*, or *bridging table*, or even the *Content-Addressable Memory (CAM) table*, in reference to the type of physical memory used to store the table.

Key Topic

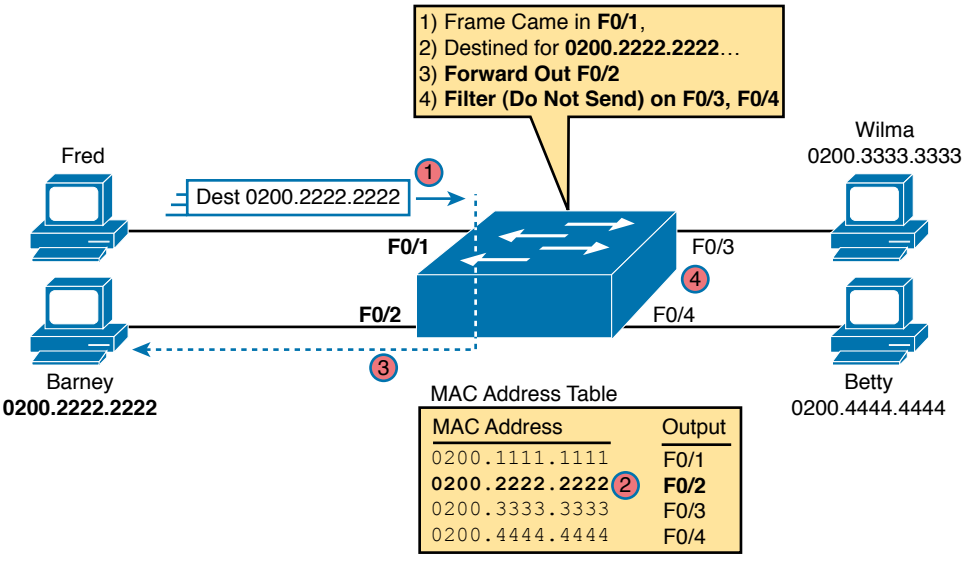


Figure 5-3 *Sample Switch Forwarding and Filtering Decision*

A switch's MAC address table lists the location of each MAC relative to that one switch. In LANs with multiple switches, each switch makes an independent forwarding decision based on its own MAC address table. Together, they forward the frame so that it eventually arrives at the destination.

For example, Figure 5-4 shows the first switching decision in a case in which Fred sends a frame to Wilma, with destination MAC 0200.3333.3333. The topology has changed versus the previous figure, this time with two switches, and Fred and Wilma connected to two different switches. Figure 5-3 shows the first switch's logic, in reaction to Fred sending the original frame. Basically, the switch receives the frame in port F0/1, finds the destination MAC (0200.3333.3333) in the MAC address table, sees the outgoing port of G0/1, so SW1 forwards the frame out its G0/1 port.

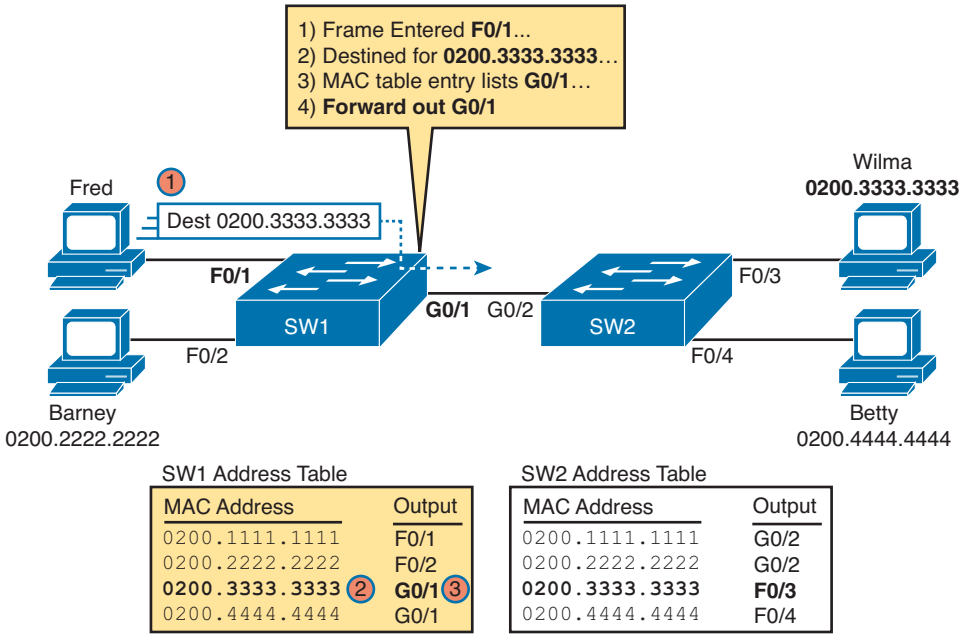


Figure 5-4 *Forwarding Decision with Two Switches: First Switch*

That same frame next arrives at switch SW2, entering SW2's G0/2 interface. As shown in Figure 5-5, SW2 uses the same logic steps, but using SW2's table. The MAC table lists the forwarding instructions for that switch only. In this case, switch SW2 forwards the frame out its F0/3 port, based on SW2's MAC address table.

Key Topic

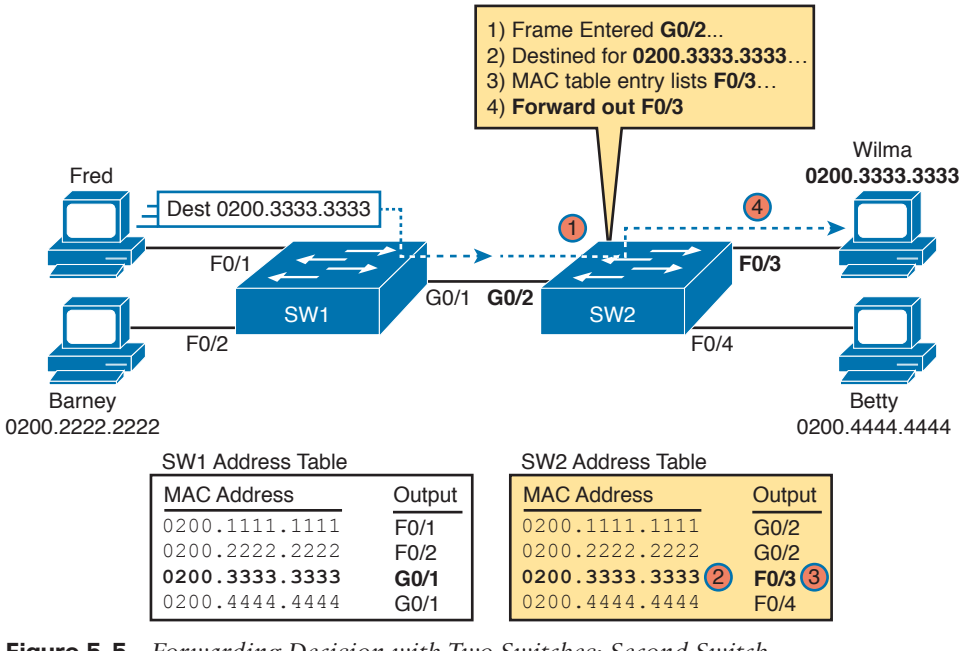


Figure 5-5 *Forwarding Decision with Two Switches: Second Switch*

NOTE The forwarding choice by a switch was formerly called a *forward-versus-filter* decision, because the switch also chooses to not forward (to filter) frames, not sending the frame out some ports.

The examples so far use switches that happen to have a MAC table with all the MAC addresses listed. As a result, the destination MAC address in the frame is known to the switch. The frames are called known unicast frames, or simply known unicasts, because the destination address is a unicast address, and the destination is known. As shown in these examples, switches forward known unicast frames out one port: the port as listed in the MAC table entry for that MAC address.

Learning MAC Addresses

Thankfully, the networking staff does not have to type in all those MAC table entries. Instead, each switch does its second main function: to learn the MAC addresses and interfaces to put into its address table. With a complete MAC address table, the switch can make accurate forwarding and filtering decisions as just discussed.

Switches build the address table by listening to incoming frames and examining the *source MAC address* in the frame. If a frame enters the switch and the source MAC address is not in the MAC address table, the switch creates an entry in the table. That table entry lists the interface from which the frame arrived. Switch learning logic is that simple.

Figure 5-6 depicts the same single-switch topology network as Figure 5-3, but before the switch has built any address table entries. The figure shows the first two frames sent in this network—first a frame from Fred, addressed to Barney, and then Barney's response, addressed to Fred.

Key Topic

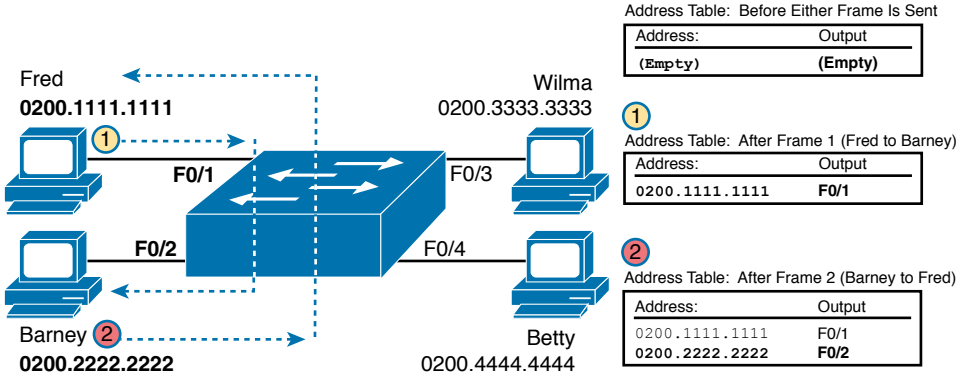


Figure 5-6 *Switch Learning: Empty Table and Adding Two Entries*

(Figure 5-6 depicts the MAC learning process only, and ignores the forwarding process and therefore ignores the destination MAC addresses.)

Focus on the learning process and how the MAC table grows at each step as shown on the right side of the figure. The switch begins with an empty MAC table, as shown in the upper-right part of the figure. Then Fred sends his first frame (labeled "1") to Barney, so the switch

adds an entry for 0200.1111.1111, Fred's MAC address, associated with interface F0/1. Why F0/1? The frame sent by Fred entered the switch's F0/1 port. SW1's logic runs something like this: "The source is MAC 0200.1111.1111, the frame entered F0/1, so from my perspective, 0200.1111.1111 must be reachable out my port F0/1."

Continuing the example, when Barney replies in Step 2, the switch adds a second entry, this one for 0200.2222.2222, Barney's MAC address, along with interface F0/2. Why F0/2? The frame Barney sent entered the switch's F0/2 interface. Learning always occurs by looking at the source MAC address in the frame and adds the incoming interface as the associated port.

Flooding Unknown Unicast and Broadcast Frames

Now again turn your attention to the forwarding process, using the topology in Figure 5-5. What do you suppose the switch does with Fred's first frame, the one that occurred when there were no entries in the MAC address table? As it turns out, when there is no matching entry in the table, switches forward the frame out all interfaces (except the incoming interface) using a process called *flooding*. And the frame whose destination address is unknown to the switch is called an *unknown unicast frame*, or simply an *unknown unicast*.

Switches flood unknown unicast frames. Flooding means that the switch forwards copies of the frame out all ports, except the port on which the frame was received. The idea is simple: if you do not know where to send it, send it everywhere, to deliver the frame. And, by the way, that device will likely then send a reply—and then the switch can learn that device's MAC address and forward future frames out one port as a known unicast frame.

Switches also flood LAN broadcast frames (frames destined to the Ethernet broadcast address of FFFF.FFFF.FFFF) because this process helps deliver a copy of the frame to all devices in the LAN.

For example, Figure 5-7 shows the same first frame sent by Fred, when the switch's MAC table is empty. At step 1, Fred sends the frame. At step 2, the switch sends a copy of the frame out all three of the other interfaces.

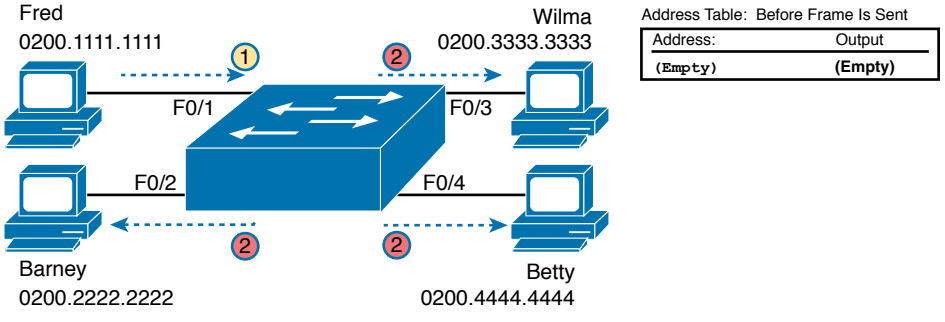


Figure 5-7 *Switch Flooding: Unknown Unicast Arrives, Floods Out Other Ports*

Avoiding Loops Using Spanning Tree Protocol

The third primary feature of LAN switches is loop prevention, as implemented by Spanning Tree Protocol (STP). Without STP, any flooded frames would loop for an indefinite period of

time in Ethernet networks with physically redundant links. To prevent looping frames, STP blocks some ports from forwarding frames so that only one active path exists between any pair of LAN segments.

The result of STP is good: frames do not loop infinitely, which makes the LAN usable. However, STP has negative features as well, including the fact that it takes some work to balance traffic across the redundant alternate links.

A simple example makes the need for STP more obvious. Remember, switches flood unknown unicast frames and broadcast frames. Figure 5-8 shows an unknown unicast frame, sent by Larry to Bob, which loops forever because the network has redundancy but no STP. Note that the figure shows one direction of the looping frame only, just to reduce clutter, but a copy of the frame would also loop the other direction.

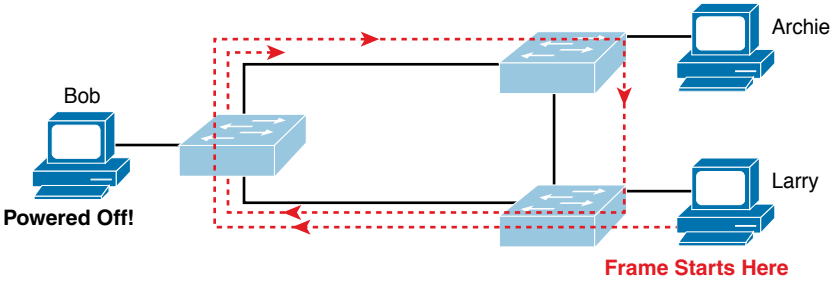


Figure 5-8 *Network with Redundant Links but Without STP: The Frame Loops Forever*

The flooding of this frame would result in the frame repeatedly rotating around the three switches, because none of the switches list Bob's MAC address in their address tables—so each switch floods the frame. And while the flooding process is a good mechanism for forwarding unknown unicasts and broadcasts, the continual flooding of traffic frames as in the figure can completely congest the LAN to the point of making it unusable.

A topology like Figure 5-8, with redundant links, is good, but we need to prevent the bad effect of those looping frames. To avoid Layer 2 loops, all switches need to use STP. STP causes each interface on a switch to settle into either a blocking state or a forwarding state. *Blocking* means that the interface cannot forward or receive data frames, while *forwarding* means that the interface can send and receive data frames. If a correct subset of the interfaces is blocked, only a single currently active logical path exists between each pair of LANs.

NOTE STP behaves identically for a transparent bridge and a switch. Therefore, the terms *bridge*, *switch*, and *bridging device* all are used interchangeably when discussing STP.

Chapter 9 of this book, "Spanning Tree Protocol Concepts," examines STP in depth, including how STP prevents loops.

LAN Switching Summary

Switches use Layer 2 logic, examining the Ethernet data-link header to choose how to process frames. In particular, switches make decisions to forward and filter frames, learn MAC addresses, and use STP to avoid loops, as follows:

Step 1. Switches forward frames based on the destination MAC address:

A. If the destination MAC address is a broadcast, multicast, or unknown destination unicast (a unicast not listed in the MAC table), the switch floods the frame.

B. If the destination MAC address is a known unicast address (a unicast address found in the MAC table):

i. If the outgoing interface listed in the MAC address table is different from the interface in which the frame was received, the switch forwards the frame out the outgoing interface.

ii. If the outgoing interface is the same as the interface in which the frame was received, the switch filters the frame, meaning that the switch simply ignores the frame and does not forward it.

Step 2. Switches use the following logic to learn MAC address table entries:

A. For each received frame, examine the source MAC address and note the interface from which the frame was received.

B. If it is not already in the table, add the MAC address and interface it was learned on.

Step 3. Switches use STP to prevent loops by causing some interfaces to block, meaning that they do not send or receive frames.

Verifying and Analyzing Ethernet Switching

A Cisco Catalyst switch comes from the factory ready to switch frames. All you have to do is connect the power cable, plug in the Ethernet cables, and the switch starts switching incoming frames. Connect multiple switches together, and they are ready to forward frames between the switches as well. And the big reason behind this default behavior has to do with the default settings on the switches.

Cisco Catalyst switches come ready to get busy switching frames because of settings like these:

- The interfaces are enabled by default, ready to start working once a cable is connected.
- All interfaces are assigned to VLAN 1.
- 10/100 and 10/100/1000 interfaces use autonegotiation by default.
- The MAC learning, forwarding, flooding logic all works by default.
- STP is enabled by default.

This second section of the chapter examines how switches will work with these default settings, showing how to verify the Ethernet learning and forwarding process.

Demonstrating MAC Learning

To see a switch's MAC address table, use the **show mac address-table** command. With no additional parameters, this command lists all known MAC addresses in the MAC table, including some overhead static MAC addresses that you can ignore. To see all the dynamically learned MAC addresses only, instead use the **show mac address-table dynamic** command.

The examples in this chapter use almost no configuration, as if you just unboxed the switch when you first purchased it. For the examples, the switches have no configuration other than the **hostname** command to set a meaningful hostname. Note that to do this in lab, all I did was

Key Topic

- Use the **erase startup-config** EXEC command to erase the startup-config file
- Use the **delete vlan.dat** EXEC command to delete the VLAN configuration details
- Use the **reload** EXEC command to reload the switch (thereby using the empty startup-config, with no VLAN information configured)
- Configure the **hostname SW1** command to set the switch hostname

Once done, the switch starts forwarding and learning MAC addresses, as demonstrated in Example 5-1.

Key Topic

Example 5-1 show mac address-table dynamic *for Figure 5-7*

```
SW1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.1111.1111    DYNAMIC     Fa0/1
   1    0200.2222.2222    DYNAMIC     Fa0/2
   1    0200.3333.3333    DYNAMIC     Fa0/3
   1    0200.4444.4444    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 4
SW1#
```

First, focus on two columns of the table: the MAC Address and Ports columns of the table. The values should look familiar: they match the earlier single-switch example, as repeated here as Figure 5-9. Note the four MAC addresses listed, along with their matching ports, as shown in the figure.

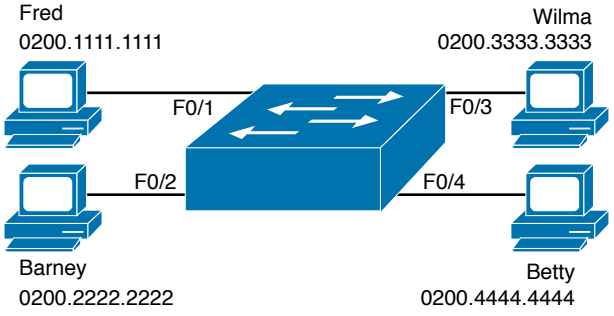


Figure 5-9 *Single Switch Topology Used in Verification Section*

Next, look at the Type field in the heading of the output table. The column tells us how the switch learned the MAC address as described earlier in this chapter; in this case, the switch learned all MAC addresses dynamically. You can also statically predefine MAC table entries using a couple of different features, including port security, and those would appear as Static in the Type column.

Finally, the VLAN column of the output gives us a chance to briefly discuss how VLANs impact switching logic. LAN switches forward Ethernet frames inside a VLAN. What that means is if a frame enters via a port in VLAN 1, then the switch will forward or flood that frame out other ports in VLAN 1 only, and not out any ports that happen to be assigned to another VLAN. Chapter 8, "Implementing Ethernet Virtual LANs," looks at all the details of how switches forward frames when using VLANs.

Switch Interfaces

The first example assumes that you installed the switch and cabling correctly, and that the switch interfaces work. Once you do the installation and connect to the Console, you can easily check the status of those interfaces with the **show interfaces status** command, as shown in Example 5-2.

Example 5-2 **show interfaces status** *on Switch SW1*

```
SW1# show interfaces status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        connected    1          a-full  a-100 10/100BaseTX
Fa0/2                        connected    1          a-full  a-100 10/100BaseTX
Fa0/3                        connected    1          a-full  a-100 10/100BaseTX
Fa0/4                        connected    1          a-full  a-100 10/100BaseTX
Fa0/5                        notconnect   1            auto   auto 10/100BaseTX
Fa0/6                        notconnect   1            auto   auto 10/100BaseTX
Fa0/7                        notconnect   1            auto   auto 10/100BaseTX
Fa0/8                        notconnect   1            auto   auto 10/100BaseTX
Fa0/9                        notconnect   1            auto   auto 10/100BaseTX
Fa0/10                       notconnect   1            auto   auto 10/100BaseTX
Fa0/11                       notconnect   1            auto   auto 10/100BaseTX
```

```
Fa0/12                    notconnect   1            auto   auto 10/100BaseTX
Fa0/13                    notconnect   1            auto   auto 10/100BaseTX
Fa0/14                    notconnect   1            auto   auto 10/100BaseTX
Fa0/15                    notconnect   1            auto   auto 10/100BaseTX
Fa0/16                    notconnect   1            auto   auto 10/100BaseTX
Fa0/17                    notconnect   1            auto   auto 10/100BaseTX
Fa0/18                    notconnect   1            auto   auto 10/100BaseTX
Fa0/19                    notconnect   1            auto   auto 10/100BaseTX
Fa0/20                    notconnect   1            auto   auto 10/100BaseTX
Fa0/21                    notconnect   1            auto   auto 10/100BaseTX
Fa0/22                    notconnect   1            auto   auto 10/100BaseTX
Fa0/23                    notconnect   1            auto   auto 10/100BaseTX
Fa0/24                    notconnect   1            auto   auto 10/100BaseTX
Gi0/1                     notconnect   1            auto   auto 10/100/1000BaseTX
Gi0/2                     notconnect   1            auto   auto 10/100/1000BaseTX
SW1#
```


Focus on the port column for a moment. As a reminder, Cisco Catalyst switches name their ports based on the fastest specification supported, so in this case, the switch has 24 interfaces named FastEthernet, and two named GigabitEthernet. Many commands abbreviate those terms, this time as Fa for FastEthernet and Gi for GigabitEthernet. (The example happens to come from a Cisco Catalyst switch that has 24 10/100 ports and two 10/100/1000 ports.)

The Status column, of course, tells us the status or state of the port. In this case, the lab switch had cables and devices connected to ports F0/1–F0/4 only, with no other cables connected. As a result, those first four ports have a state of connected, meaning that the ports have a cable and are functional. The notconnect state means that the port is not yet functioning. It may mean that there is no cable installed, but other problems may exist as well. (The section "Analyzing Switch Interface Status and Statistics," in Chapter 7, "Configuring and Verifying Switch Interfaces," works through the details of what causes a switch interface to fail.)

NOTE You can see the status for a single interface in a couple of ways. For instance, for F0/1, the command **show interfaces f0/1 status** lists the status in a single line of output as in Example 5-2. The **show interfaces f0/1** command (without the **status** keyword) displays a detailed set of messages about the interface.

The **show interfaces** command has a large number of options. One particular option, the **counters** option, lists statistics about incoming and outgoing frames on the interfaces. In particular, it lists the number of unicast, multicast, and broadcast frames (both the in and out directions), and a total byte count for those frames. Example 5-3 shows an example, again for interface F0/1.

Example 5-3 **show interfaces f0/1 counters** *on Switch SW1*

```
SW1# show interfaces f0/1 counters

Port            InOctets     InUcastPkts     InMcastPkts      InBcastPkts
Fa0/1            1223303           10264             107               18

Port           OutOctets    OutUcastPkts    OutMcastPkts     OutBcastPkts
Fa0/1            3235055           13886           22940              437
```

Finding Entries in the MAC Address Table

With a single switch and only four hosts connected to it, you can just read the details of the MAC address table and find the information you want to see. However, in real networks, with lots of interconnected hosts and switches, just reading the output to find one MAC address can be hard to do. You might have hundreds of entries—page after page of output—with each MAC address looking like a random string of hex characters. (The book uses easy-to-recognize MAC addresses to make it easier to learn.)

Thankfully, Cisco IOS supplies several more options on the **show mac address-table** command to make it easier to find individual entries. First, if you know the MAC address, you can search for it—just type in the MAC address at the end of the command, as shown in Example 5-4. All you have to do is include the **address** keyword, followed by the actual MAC address. If the address exists, the output lists the address. Note that the output lists the exact same information in the exact same format, but it lists only the line for the matching MAC address.

Example 5-4 **show mac address-table dynamic** *with the* **address** *Keyword*

```
SW1# show mac address-table dynamic address 0200.1111.1111
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.1111.1111    DYNAMIC     Fa0/1
Total Mac Addresses for this criterion: 1
```

While this information is useful, often the engineer troubleshooting a problem does not know the MAC addresses of the devices connected to the network. Instead, the engineer has a topology diagram, knowing which switch ports connect to other switches and which connect to endpoint devices.

Sometimes you might be troubleshooting while looking at a network topology diagram and want to look at all the MAC addresses learned off a particular port. IOS supplies that option with the **show mac address-table dynamic interface** command. Example 5-5 shows one example, for switch SW1's F0/1 interface.

Example 5-5 show mac address-table dynamic *with the* interface *Keyword*

```
SW1# show mac address-table dynamic interface fastEthernet 0/1
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.1111.1111    DYNAMIC     Fa0/1
Total Mac Addresses for this criterion: 1
```

Finally, you may also want to find the MAC address table entries for one VLAN. You guessed it—you can add the **vlan** parameter, followed by the VLAN number. Example 5-6 shows two such examples from the same switch SW1 from Figure 5-7—one for VLAN 1, where all four devices reside, and one for a nonexistent VLAN 2.

Example 5-6 *The* show mac address-table vlan *Command*

```
SW1# show mac address-table dynamic vlan 1
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.1111.1111    DYNAMIC     Fa0/1
   1    0200.2222.2222    DYNAMIC     Fa0/2
   1    0200.3333.3333    DYNAMIC     Fa0/3
   1    0200.4444.4444    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 4
SW1#
SW1# show mac address-table dynamic vlan 2
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
SW1#
```

Managing the MAC Address Table (Aging, Clearing)

This chapter closes with a few comments about how switches manage their MAC address tables. Switches do learn MAC addresses, but those MAC addresses do not remain in the table indefinitely. The switch will remove the entries due to age, due to the table filling, and you can remove entries using a command.

First, for aging out MAC table entries, switches remove entries that have not been used for a defined number of seconds (default of 300 seconds on many switches). To do that, switches

look at every incoming frame and every source MAC address, and do something related to learning. If it is a new MAC address, the switch adds the correct entry to the table, of course. However, if that entry already exists, the switch still does something: it resets the inactivity timer back to 0 for that entry. Each entry's timer counts upward over time to measure how long the entry has been in the table. The switch times out (removes) any entries whose timer reaches the defined aging time.

Example 5-7 shows the aging timer setting for the entire switch. The aging time can be configured to a different time, globally and per-VLAN using the **mac address-table aging-time** *time-in-seconds* [**vlan** *vlan-number*] global configuration command. The example shows a case with all defaults, with the global setting of 300 seconds, and no per-VLAN overrides.

Example 5-7 *The MAC Address Default Aging Timer Displayed*

```
SW1# show mac address-table aging-time
Global Aging Time:  300
Vlan    Aging Time
----    ----------
SW1#

SW1# show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count  : 4
Static  Address Count  : 0
Total Mac Addresses    : 4

Total Mac Address Space Available: 7299
```

Each switch also removes the oldest table entries, even if they are younger than the aging time setting, if the table fills. The MAC address table uses content-addressable memory (CAM), a physical memory that has great table lookup capabilities. However, the size of the table depends on the size of the CAM in a particular model of switch and based on some configurable settings in the switch. When a switch tries to add a new MAC table entry and finds the table full, the switch times out (removes) the oldest table entry to make space. For perspective, the end of Example 5-7 lists the size of a Cisco Catalyst switch's MAC table at about 8000 entries—the same four existing entries from the earlier examples, with space for 7299 more.

Finally, you can remove the dynamic entries from the MAC address table with the **clear mac address-table dynamic** command. Note that the **show** commands in this chapter can be executed from user and enable mode, but the **clear** command happens to be an enable mode command. The command also allows parameters to limit the types of entries cleared, as follows:

- By VLAN: **clear mac address-table dynamic vlan** *vlan-number*
- By Interface: **clear mac address-table dynamic interface** *interface-id*
- By MAC address: **clear mac address-table dynamic address** *mac-address*

MAC Address Tables with Multiple Switches

Finally, to complete the discussion, it helps to think about an example with multiple switches, just to emphasize how MAC learning, forwarding, and flooding happen independently on each LAN switch.

Consider the topology in Figure 5-10, and pay close attention to the port numbers. The ports were purposefully chosen so that neither switch used any of the same ports for this example. That is, switch SW2 does have a port F0/1 and F0/2, but I did not plug any devices into those ports when making this example. Also note that all ports are in VLAN 1, and as with the other examples in this chapter, all default configuration is used other than the hostname on the switches.

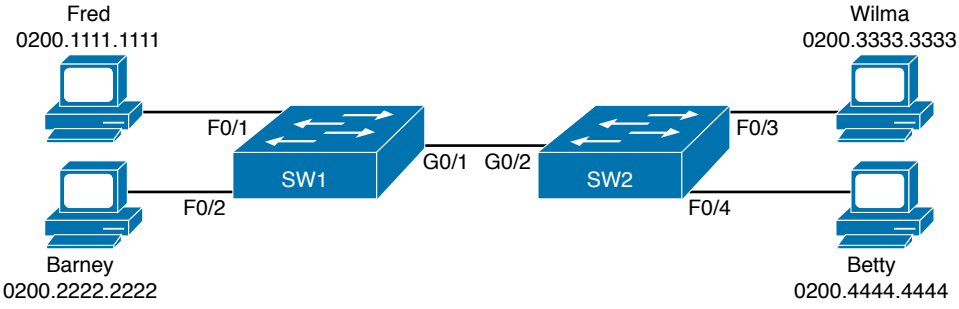


Figure 5-10 *Two-Switch Topology Example*

Think about a case in which both switches learn all four MAC addresses. For instance, that would happen if the hosts on the left communicate with the hosts on the right. SW1's MAC address table would list SW1's own port numbers (F0/1, F0/2, and G0/1) because SW1 uses that information to decide where SW1 should forward frames. Similarly, SW2's MAC table lists SW2's port numbers (F0/3, F0/4, G0/2 in this example). Example 5-8 shows the MAC address tables on both switches for that scenario.

Example 5-8 *The MAC Address Table on Two Switches*

```
SW1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.1111.1111    DYNAMIC     Fa0/1
   1    0200.2222.2222    DYNAMIC     Fa0/2
   1    0200.3333.3333    DYNAMIC     Gi0/1
   1    0200.4444.4444    DYNAMIC     Gi0/1
Total Mac Addresses for this criterion: 4
```

```
! The next output is from switch SW2
SW2# show mac address-table dynamic
   1    0200.1111.1111    DYNAMIC     Gi0/2
```

```
   1    0200.2222.2222    DYNAMIC     Gi0/2
   1    0200.3333.3333    DYNAMIC     Fa0/3
   1    0200.4444.4444    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 4
```

Chapter Review

Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Table 5-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 5-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Do labs		Book, Sim Lite, blog
Review command tables		Book

Review All the Key Topics

Key Topic

Table 5-3 Key Topics for Chapter 5

Key Topic Element	Description	Page Number
List	Three main functions of a LAN switch	109
Figure 5-3	Process to forward a known unicast frame	111
Figure 5-5	Process to forward a known unicast, second switch	112
Figure 5-6	Process to learn MAC addresses	113
List	Summary of switch forwarding logic	117
Example 5-1	The **show mac address-table dynamic** command	117

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included free with this book. The subset of labs mostly relate to this part of the book, so take the time to try some of the labs.

As always, also check the author's blog site pages for configuration exercises (Config Labs) at http://blog.certskills.com.

Key Terms You Should Know

broadcast frame, known unicast frame, Spanning Tree Protocol (STP), unknown unicast frame, MAC address table, forward, flood

Command References

Table 5-4 lists the verification commands used in this chapter. As an easy review exercise, cover the left column, read the right, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 5-4 Chapter 5 EXEC Command Reference

Command	Mode/Purpose/Description
show mac address-table	Shows all MAC table entries of all types
show mac address-table dynamic	Shows all dynamically learned MAC table entries
show mac address-table dynamic vlan *vlan-id*	Shows all dynamically learned MAC table entries in that VLAN
show mac address-table dynamic address *mac-address*	Shows the dynamically learned MAC table entries with that MAC address
show mac address-table dynamic interface *interface-id*	Shows all dynamically learned MAC table entries associated with that interface
show mac address-table count	Shows the number of entries in the MAC table and the total number of remaining empty slots in the MAC table
show mac address-table aging-time	Shows the global and per-VLAN aging timeout for inactive MAC table entries
clear mac address-table dynamic	Empties the MAC table of all dynamic entries
show interfaces status	Lists one line per interface on the switch, with basic status and operating information for each
clear mac address-table dynamic [vlan *vlan-number*] **[interface** *interface-id*] **[address** *mac-address*]	Clears (removes) dynamic MAC table entries: either all (with no parameters), or a subset based on VLAN ID, interface ID, or a specific MAC address

Note that this chapter also includes reference to one configuration command, so it does not call for the use of a separate table. For review, the command is

mac address-table aging-time *time-in-seconds* [**vlan** *vlan-number*]

CHAPTER 6

Configuring Basic Switch Management

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

4.0 IP Services

4.6 Configure and verify DHCP client and relay

4.8 Configure network devices for remote access using SSH

5.0 Security Fundamentals

5.3 Configure device access control using local passwords

The work performed by a networking device can be divided into three broad categories. The first and most obvious, called the data plane, is the work a switch does to forward frames generated by the devices connected to the switch. In other words, the data plane is the main purpose of the switch. Second, the control plane refers to the configuration and processes that control and change the choices made by the switch's data plane. The network engineer can control which interfaces are enabled and disabled, which ports run at which speeds, how Spanning Tree blocks some ports to prevent loops, and so on.

The third category, the management plane, is the topic of this chapter. The management plane deals with managing the device itself, rather than controlling what the device is doing. In particular, this chapter looks at the most basic management features that can be configured in a Cisco switch. The first section of the chapter works through the configuration of different kinds of login security. The second section shows how to configure IPv4 settings on a switch so it can be remotely managed. The last (short) section then explains a few practical matters that can make your life in the lab a little easier.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 6-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Securing the Switch CLI	1–3
Enabling IP for Remote Access	4–5
Miscellaneous Settings Useful in Lab	6

1. Imagine that you have configured the **enable secret** command, followed by the **enable password** command, from the console. You log out of the switch and log back in at the console. Which command defines the password that you had to enter to access privileged mode?
   a. **enable password**
   b. **enable secret**
   c. Neither
   d. The **password** command, if it is configured
2. An engineer wants to set up simple password protection with no usernames for some switches in a lab, for the purpose of keeping curious coworkers from logging in to the lab switches from their desktop PCs. Which of the following commands would be a useful part of that configuration?
   a. A **login** vty mode subcommand
   b. A **password** *password* console subcommand
   c. A **login local** vty subcommand
   d. A **transport input ssh** vty subcommand
3. An engineer had formerly configured a Cisco 2960 switch to allow Telnet access so that the switch expected a password of **mypassword** from the Telnet user. The engineer then changed the configuration to support Secure Shell. Which of the following commands could have been part of the new configuration? (Choose two answers.)
   a. A **username** *name* **secret** *password* vty mode subcommand
   b. A **username** *name* **secret** *password* global configuration command
   c. A **login local** vty mode subcommand
   d. A **transport input ssh** global configuration command

4. An engineer's desktop PC connects to a switch at the main site. A router at the main site connects to each branch office through a serial link, with one small router and switch at each branch. Which of the following commands must be configured on the branch office switches, in the listed configuration mode, to allow the engineer to telnet to the branch office switches and supply only a password to login? (Choose three answers.)
   a. The **ip address** command in interface configuration mode
   b. The **ip address** command in global configuration mode
   c. The **ip default-gateway** command in VLAN configuration mode
   d. The **ip default-gateway** command in global configuration mode
   e. The **password** command in console line configuration mode
   f. The **password** command in vty line configuration mode
5. A Layer 2 switch configuration places all its physical ports into VLAN 2. The IP addressing plan shows that address 172.16.2.250 (with mask 255.255.255.0) is reserved for use by this new LAN switch and that 172.16.2.254 is already configured on the router connected to that same VLAN. The switch needs to support SSH connections into the switch from any subnet in the network. Which of the following commands are part of the required configuration in this case? (Choose two answers.)
   a. The **ip address 172.16.2.250 255.255.255.0** command in interface vlan 1 configuration mode.
   b. The **ip address 172.16.2.250 255.255.255.0** command in interface vlan 2 configuration mode.
   c. The **ip default-gateway 172.16.2.254** command in global configuration mode.
   d. The switch cannot support SSH because all its ports connect to VLAN 2, and the IP address must be configured on interface VLAN 1.
6. Which of the following line subcommands tells a switch to wait until a show command's output has completed before displaying log messages on the screen?
   a. **logging synchronous**
   b. **no ip domain-lookup**
   c. **exec-timeout 0 0**
   d. **history size 15**

Foundation Topics

Securing the Switch CLI

By default, a Cisco Catalyst switch allows anyone to connect to the console port, access user mode, and then move on to enable and configuration modes without any kind of security. That default makes sense, given that if you can get to the console port of the switch, you already have control over the switch physically. However, everyone needs to operate switches remotely, and the first step in that process is to secure the switch so that only the appropriate users can access the switch command-line interface (CLI).

This first topic in the chapter examines how to configure login security for a Cisco Catalyst switch. Securing the CLI includes protecting access to enable mode, because from enable mode, an attacker could reload the switch or change the configuration. Protecting user mode is also important, because attackers can see the status of the switch, learn about the network, and find new ways to attack the network.

Note that all remote access and management protocols require that the switch IP configuration be completed and working. A switch's IPv4 configuration has nothing to do with how a Layer 2 switch forwards Ethernet frames (as discussed in Chapter 5, "Analyzing Ethernet LAN Switching"). Instead, to support Telnet and Secure Shell (SSH) into a switch, the switch needs to be configured with an IP address. This chapter also shows how to configure a switch's IPv4 settings in the upcoming section "Enabling IPv4 for Remote Access."

In particular, this section covers the following login security topics:

- Securing user mode and privileged mode with simple passwords
- Securing user mode access with local usernames
- Securing user mode access with external authentication servers
- Securing remote access with Secure Shell (SSH)

Securing User Mode and Privileged Mode with Simple Passwords

By default, Cisco Catalyst switches allow full access from the console but no access via Telnet or SSH. Using default settings, a console user can move into user mode and then privileged mode with no passwords required; however, default settings prevent remote users from accessing even user mode.

The defaults work great for a brand new switch, but in production, you will want to secure access through the console as well as enable remote login via Telnet and/or SSH so you can sit at your desk and log in to all the switches in the LAN. Keep in mind, however, that you should not open the switch for just anyone to log in and change the configuration, so some type of secure login should be used.

Most people use a simple shared password for access to lab gear. This method uses a password only—with no username—with one password for console users and a different password for Telnet users. Console users must supply the *console password*, as configured in console line configuration mode. Telnet users must supply the *Telnet password*, also called the vty password, so called because the configuration sits in vty line configuration mode. Figure 6-1 summarizes these options for using shared passwords from the perspective of the user logging in to the switch.

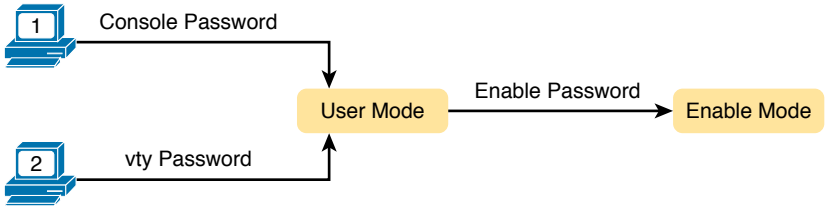


Figure 6-1 *Simple Password Security Concepts*

NOTE This section refers to several passwords as *shared* passwords. Users share these passwords in that all users must know and use that same password. In other words, each user does not have a unique username/password to use, but rather, all the appropriate staff knows and uses the same password.

In addition, Cisco switches protect enable mode (also called privileged mode) with yet another shared password called the *enable password*. From the perspective of the network engineer connecting to the CLI of the switch, once in user mode, the user types the **enable** EXEC command. This command prompts the user for this enable password; if the user types the correct password, IOS moves the user to enable mode.

Example 6-1 shows an example of the user experience of logging in to a switch from the console when the shared console password and the shared enable password have both been set. Note that before this example began, the user started the terminal emulator, physically connected a laptop to the console cable, and then pressed the Return key to make the switch respond as shown at the top of the example.

Example 6-1 *Console Login and Movement to Enable Mode*

```
(User now presses enter now to start the process. This line of text does not appear.)

User Access Verification

Password: faith
Switch> enable
Password: love
Switch#
```

Note that the example shows the password text as if typed (faith and love), along with the **enable** command that moves the user from user mode to enable mode. In reality, the switch hides the passwords when typed, to prevent someone from reading over your shoulder to see the passwords.

To configure the shared passwords for the console, Telnet, and for enable mode, you need to configure several commands. However, the parameters of the commands can be pretty intuitive. Figure 6-2 shows the configuration of all three of these passwords.

The configuration for these three passwords does not require a lot of work. First, the console and vty password configuration sets the password based on the context: console mode for the console (**line con 0**), and vty line configuration mode for the Telnet password (**line vty 0 15**). Then inside console mode and vty mode, respectively, the two commands in each mode are as follows:

password *password-value*: Defines the actual password used on the console or vty

login: Tells IOS to enable the use of a simple shared password (with no username) on this line (console or vty), so that the switch asks the user for a password

---

Answers to the "Do I Know This Already?" quiz:

1 B **2** A **3** B, C **4** A, D, F **5** B, C **6** A

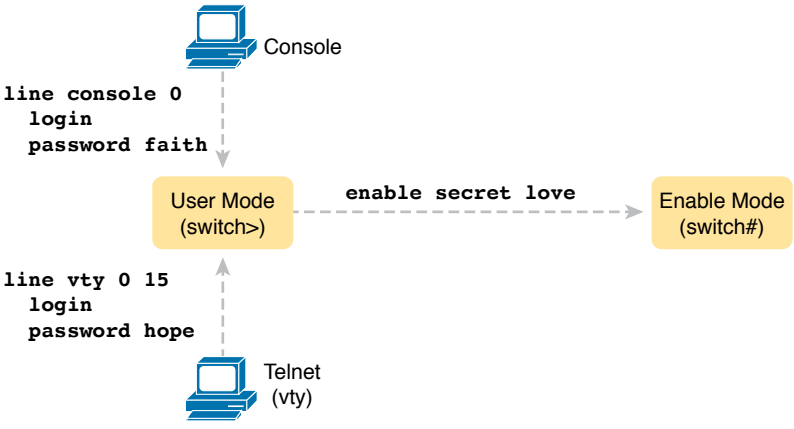


Figure 6-2 *Simple Password Security Configuration*

The configured enable password, shown on the right side of the figure, applies to all users, no matter whether they connect to user mode via the console, Telnet, or otherwise. The command to configure the enable password is a global configuration command: **enable secret** *password-value*.

NOTE Older IOS versions used the command **enable password** *password-value* to set the enable password, and that command still exists in IOS. However, the **enable secret** command is much more secure. In real networks, use **enable secret**. Chapter 5, "Securing Network Devices," in the *CCNA 200-301 Official Cert Guide, Volume 2*, explains more about the security levels of various password mechanisms, including a comparison of the **enable secret** and **enable password** commands.

To help you follow the process, and for easier study later, use the configuration checklist before the example. The configuration checklist collects the required and optional steps to configure a feature as described in this book. The configuration checklist for shared passwords for the console, Telnet, and enable passwords is

Config Checklist

Step 1. Configure the enable password with the **enable secret** *password-value* command.

Step 2. Configure the console password:

A. Use the **line con 0** command to enter console configuration mode.

B. Use the **password** *password-value* subcommand to set the value of the console password.

C. Use the **login** subcommand to enable console password security using a simple password.

Step 3. Configure the Telnet (vty) password:

A. Use the **line vty 0 15** command to enter vty configuration mode for all 16 vty lines (numbered 0 through 15).

B. Use the **password** *password-value* subcommand to set the value of the console password.

C. Use the **login** subcommand to enable console password security using a simple password.

Example 6-2 shows the configuration process as noted in the configuration checklist, along with setting the enable secret password. Note that the lines which begin with a ! are comment lines; they are there to guide you through the configuration.

Key Topic

Example 6-2 *Configuring Basic Passwords*

```
! Enter global configuration mode, set the enable password, and also
! set the hostname (just because it makes sense to do so)
!
Switch# configure terminal
Switch(config)# enable secret love
!
! At Step 2 in the checklist, enter console configuration mode, set the
! password value to "faith" and enable simple passwords for the console.
! The exit command moves the user back to global config mode.
!
Switch#(config)# line console 0
Switch#(config-line)# password faith
Switch#(config-line)# login
Switch#(config-line)# exit
!
! The next few lines do basically the same configuration, except it is
! for the vty lines. Telnet users will use "hope" to login.
!
Switch#(config)# line vty 0 15
Switch#(config-line)# password hope
Switch#(config-line)# login
Switch#(config-line)# end
Switch#
```

Example 6-3 shows the resulting configuration in the switch per the **show running-config** command. The gray lines highlight the new configuration. Note that many unrelated lines of output have been deleted from the output to keep focused on the password configuration.

Example 6-3 *Resulting Running-Config File (Subset) Per Example 6-2 Configuration*

```
Switch# show running-config
!
Building configuration...
```

```
Current configuration: 1333 bytes
!
version 12.2
!
enable secret 5 $1$OwtI$A58c2XgqWyDNeDnv51mNR.
!
interface FastEthernet0/1
!
interface FastEthernet0/2
!
! Several lines have been omitted here - in particular, lines for
! FastEthernet interfaces 0/3 through 0/23.
!
interface FastEthernet0/24
!
interface GigabitEthernet0/1
!
interface GigabitEthernet0/2
!
line con 0
 password faith
 login
!
line vty 0 4
 password hope
 login
!
line vty 5 15
 password hope
 login
```

NOTE For historical reasons, the output of the **show running-config** command, in the last six lines of Example 6-3, separates the first five vty lines (0 through 4) from the rest (5 through 15).

Securing User Mode Access with Local Usernames and Passwords

Cisco switches support two other login security methods that both use per-user username/password pairs instead of a shared password with no username. One method, referred to as local usernames and passwords, configures the username/password pairs locally—that is, in the switch's configuration. Switches support this local username/password option for the console, for Telnet, and even for SSH, but do not replace the enable password used to reach enable mode.

The configuration to migrate from using the simple shared passwords to instead using local usernames/passwords requires only some small configuration changes, as shown in Figure 6-3.

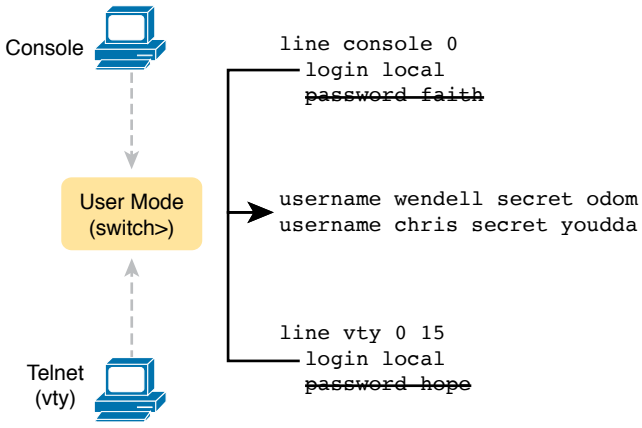


Figure 6-3 *Configuring Switches to Use Local Username Login Authentication*

Working through the configuration in the figure, first, the switch of course needs to know the list of username/password pairs. To create these, repeatedly use the **username** *name* **secret** *password* global configuration command. Then, to enable this different type of console or Telnet security, simply enable this login security method with the **login local** line. Basically, this command means "use the local list of usernames for login." You can also use the **no password** command (without even typing in the password) to clean up any remaining password subcommands from console or vty mode because these commands are not needed when using local usernames and passwords.

The following checklist details the commands to configure local username login, mainly as a method for easier study and review:

Config Checklist

Step 1. Use the **username** *name* **secret** *password* global configuration command to add one or more username/password pairs on the local switch.

Step 2. Configure the console to use locally configured username/password pairs:

- **A.** Use the **line con 0** command to enter console configuration mode.
- **B.** Use the **login local** subcommand to enable the console to prompt for both username and password, checked versus the list of local usernames/passwords.
- **C.** (Optional) Use the **no password** subcommand to remove any existing simple shared passwords, just for good housekeeping of the configuration file.

Step 3. Configure Telnet (vty) to use locally configured username/password pairs.

- **A.** Use the **line vty 0 15** command to enter vty configuration mode for all 16 vty lines (numbered 0 through 15).
- **B.** Use the **login local** subcommand to enable the switch to prompt for both username and password for all inbound Telnet users, checked versus the list of local usernames/passwords.
- **C.** (Optional) Use the **no password** subcommand to remove any existing simple shared passwords, just for good housekeeping of the configuration file.

When a Telnet user connects to the switch configured as shown in Figure 6-3, the user will be prompted first for a username and then for a password, as shown in Example 6-4. The username/password pair must be from the list of local usernames; otherwise, the login is rejected.

Example 6-4 *Telnet Login Process After Applying Configuration in Figure 6-3*

```
SW2# telnet 10.9.9.19
Trying 10.9.9.19 ... Open



User Access Verification

Username: wendell
Password:
SW1> enable
Password:
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)#^Z
SW1#
*Mar 1 02:00:56.229: %SYS-5-CONFIG_I: Configured from console by wendell on vty0
(10.9.9.19)
```


NOTE Example 6-4 does not show the password value as having been typed because Cisco switches do not display the typed password for security reasons.

Securing User Mode Access with External Authentication Servers

The end of Example 6-4 points out one of the many security improvements when requiring each user to log in with their own username. The end of the example shows the user entering configuration mode (**configure terminal**) and then immediately leaving (**end**). Note that when a user exits configuration mode, the switch generates a log message. If the user logged in with a username, the log message identifies that username; note the "wendell" in the log message.

However, using a username/password configured directly on the switch causes some administrative headaches. For instance, every switch and router needs the configuration for all users who might need to log in to the devices. Then, when any changes need to happen, like an occasional change to the passwords for good security practices, the configuration of all devices must be changed.

A better option would be to use tools like those used for many other IT login functions. Those tools allow for a central place to securely store all username/password pairs, with tools to make users change their passwords regularly, tools to revoke users when they leave their current jobs, and so on.

Cisco switches allow exactly that option using an external server called an authentication, authorization, and accounting (AAA) server. These servers hold the usernames/passwords. Typically, these servers allow users to do self-service and forced maintenance to their passwords. Many production networks use AAA servers for their switches and routers today.

The underlying login process requires some additional work on the part of the switch for each user login, but once set up, the username/password administration is much less. When using a AAA server for authentication, the switch (or router) simply sends a message to the AAA server asking whether the username and password are allowed, and the AAA server replies. Figure 6-4 shows an example, with the user first supplying a username/password, the switch asking the AAA server, and the server replying to the switch stating that the username/password is valid.

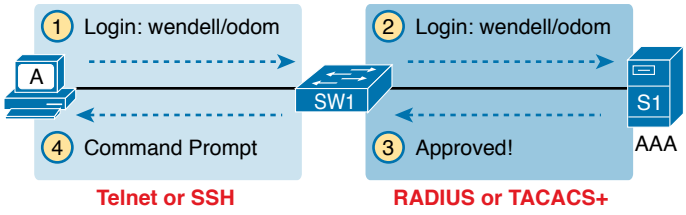


Figure 6-4 *Basic Authentication Process with an External AAA Server*

While the figure shows the general idea, note that the information flows with a couple of different protocols. On the left, the connection between the user and the switch or router uses Telnet or SSH. On the right, the switch and AAA server typically use either the RADIUS or TACACS+ protocol, both of which encrypt the passwords as they traverse the network.

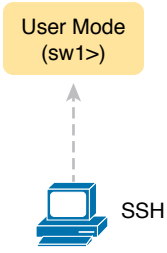
Securing Remote Access with Secure Shell

So far, this chapter has focused on the console and on Telnet, mostly ignoring SSH. Telnet has one serious disadvantage: all data in the Telnet session flows as clear text, including the password exchanges. So, anyone that can capture the messages between the user and the switch (in what is called a man-in-the-middle attack) can see the passwords. SSH encrypts all data transmitted between the SSH client and server, protecting the data and passwords.

SSH can use the same local login authentication method as Telnet, with the locally configured username and password. (SSH cannot rely on authentication methods that do not include a username, like shared passwords.) So, the configuration to support local usernames for Telnet, as shown previously in Figure 6-3, also enables local username authentication for incoming SSH connections.

Figure 6-5 shows one example configuration of what is required to support SSH. The figure repeats the local username configuration as shown earlier in Figure 6-3, as used for Telnet. Figure 6-5 shows three additional commands required to complete the configuration of SSH on the switch.

Key Topic



SSH-Specific Configuration

```
hostname sw1
ip domain-name example.com
! Next Command Uses FQDN "sw1.example.com"
crypto key generate rsa
```

Local Username Configuration (Like Telnet)

```
username wendell secret odom
username chris secret youdda
!
line vty 0 15
  login local
```

Figure 6-5 *Adding SSH Configuration to Local Username Configuration*

IOS uses the three SSH-specific configuration commands in the figure to create the SSH encryption keys. The SSH server uses the fully qualified domain name (FQDN) of the switch as input to create that key. The switch creates the FQDN from the hostname and domain name of the switch. Figure 6-5 begins by setting both values (just in case they are not already configured). Then the third command, the **crypto key generate rsa** command, generates the SSH encryption keys.

The configuration in Figure 6-5 relies on two default settings that the figure therefore conveniently ignored. IOS runs an SSH server by default. In addition, IOS allows SSH connections into the vty lines by default.

Seeing the configuration happen in configuration mode, step by step, can be particularly helpful with SSH configuration. Note in particular that in this example, the **crypto key** command prompts the user for the key modulus; you could also add the parameters **modulus** *modulus-value* to the end of the **crypto key** command to add this setting on the command. Example 6-5 shows the commands in Figure 6-5 being configured, with the encryption key as the final step.

Example 6-5 *SSH Configuration Process to Match Figure 6-5*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
!
! Step 1 next. The hostname is already set, but it is repeated just
! to be obvious about the steps.
!
SW1(config)# hostname SW1
SW1(config)# ip domain-name example.com
SW1(config)# crypto key generate rsa
The name for the keys will be: SW1.example.com
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.
```

```
How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 4 seconds)
SW1(config)#
!
! Optionally, set the SSH version to version 2 (only) - preferred
!
SW1(config)# ip ssh version 2
!
! Next, configure the vty lines for local username support, just like
! with Telnet
!
SW1(config)# line vty 0 15
SW1(config-line)# login local
SW1(config-line)# exit
!
! Define the local usernames, just like with Telnet
!
SW1(config)# username wendell password odom
SW1(config)# username chris password youdaman
SW1(config)# ^Z
SW1#
```

Earlier, I mentioned that one useful default was that the switch defaults to support both SSH and Telnet on the vty lines. However, because Telnet is a security risk, you could disable Telnet to enforce a tighter security policy. (For that matter, you can disable SSH support and allow Telnet on the vty lines as well.)

To control which protocols a switch supports on its vty lines, use the **transport input** {**all | none | telnet | ssh**} vty subcommand in vty mode, with the following options:

transport input all or **transport input telnet ssh:** Support both Telnet and SSH

transport input none: Support neither

transport input telnet: Support only Telnet

transport input ssh: Support only SSH

To complete this section about SSH, the following configuration checklist details the steps for one method to configure a Cisco switch to support SSH using local usernames. (SSH support in IOS can be configured in several ways; this checklist shows one simple way to configure it.) The process shown here ends with a comment to configure local username support on vty lines, as was discussed earlier in the section titled "Securing User Mode Access with Local Usernames and Passwords."

Config Checklist

Step 1. Configure the switch to generate a matched public and private key pair to use for encryption:

A. If not already configured, use the **hostname** *name* in global configuration mode to configure a hostname for this switch.

B. If not already configured, use the **ip domain-name** *name* in global configuration mode to configure a domain name for the switch, completing the switch's FQDN.

C. Use the **crypto key generate rsa** command in global configuration mode (or the **crypto key generate rsa modulus** *modulus-value* command to avoid being prompted for the key modulus) to generate the keys. (Use at least a 768-bit key to support SSH version 2.)

Step 2. (Optional) Use the **ip ssh version 2** command in global configuration mode to override the default of supporting both versions 1 and 2, so that only SSHv2 connections are allowed.

Step 3. (Optional) If not already configured with the setting you want, configure the vty lines to accept SSH and whether to also allow Telnet:

A. Use the **transport input ssh** command in vty line configuration mode to allow SSH only.

B. Use the **transport input all** command (default) or **transport input telnet ssh** command in vty line configuration mode to allow both SSH and Telnet.

Step 4. Use various commands in vty line configuration mode to configure local username login authentication as discussed earlier in this chapter.

NOTE Cisco routers often default to **transport input none**, so you must add the **transport input** line subcommand to enable Telnet and/or SSH into a router.

Two key commands give some information about the status of SSH on the switch. First, the **show ip ssh** command lists status information about the SSH server itself. The **show ssh** command then lists information about each SSH client currently connected into the switch. Example 6-6 shows samples of each, with user wendell currently connected to the switch.

Example 6-6 *Displaying SSH Status*

```
SW1# show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3

SW1# show ssh
Connection Version Mode  Encryption    Hmac          State               Username
0          2.0     IN    aes126-cbc    hmac-sha1     Session started     wendell
0          2.0     OUT   aes126-cbc    hmac-sha1     Session started     wendell
%No SSHv1 server connections running.
```

Enabling IPv4 for Remote Access

To allow Telnet or SSH access to the switch, and to allow other IP-based management protocols (for example, Simple Network Management Protocol, or SNMP) to function as intended,

the switch needs an IP address, as well as a few other related settings. The IP address has nothing to do with how switches forward Ethernet frames; it simply exists to support overhead management traffic.

This next topic begins by explaining the IPv4 settings needed on a switch, followed by the configuration. Note that although switches can be configured with IPv6 addresses with commands similar to those shown in this chapter, this chapter focuses solely on IPv4. All references to IP in this chapter imply IPv4.

Host and Switch IP Settings

A switch needs the same kind of IP settings as a PC with a single Ethernet interface. For perspective, a PC has a CPU, with the operating system running on the CPU. It has an Ethernet network interface card (NIC). The OS configuration includes an IP address associated with the NIC, either configured or learned dynamically with DHCP.

A switch uses the same ideas, except that the switch needs to use a virtual NIC inside the switch. Like a PC, a switch has a real CPU, running an OS (called IOS). The switch obviously has lots of Ethernet ports, but instead of assigning its management IP address to any of those ports, the switch then uses a NIC-like concept called a switched virtual interface (SVI), or more commonly, a VLAN interface, that acts like the switch's own NIC. Then the settings on the switch look something like a host, with the switch configuration assigning IP settings, like an IP address, to this VLAN interface, as shown in Figure 6-6.

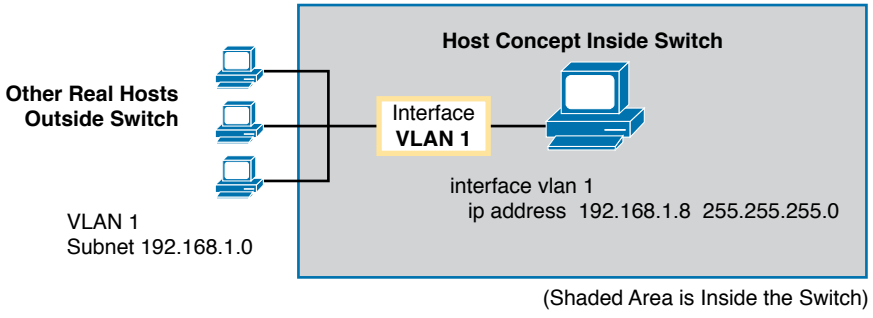


Figure 6-6 *Switch Virtual Interface (SVI) Concept Inside a Switch*

By using interface VLAN 1 for the IP configuration, the switch can then send and receive frames on any of the ports in VLAN 1. In a Cisco switch, by default, all ports are assigned to VLAN 1.

In most networks, switches configure many VLANs, so the network engineer has a choice of where to configure the IP address. That is, the management IP address does not have to be configured on the VLAN 1 interface (as configured with the **interface vlan 1** command seen in Figure 6-6).

A Layer 2 Cisco LAN switch needs only one IP address for management purposes. However, you can choose to use any VLAN to which the switch connects. The configuration then includes a VLAN interface for that VLAN number, with an appropriate IP address.

For example, Figure 6-7 shows a Layer 2 switch with some physical ports in two different VLANs (VLANs 1 and 2). The figure also shows the subnets used on those VLANs. The network engineer could choose to use either

- Interface VLAN 1, with an IP address in subnet 192.168.1.0
- Interface VLAN 2, with an IP address in subnet 192.168.2.0

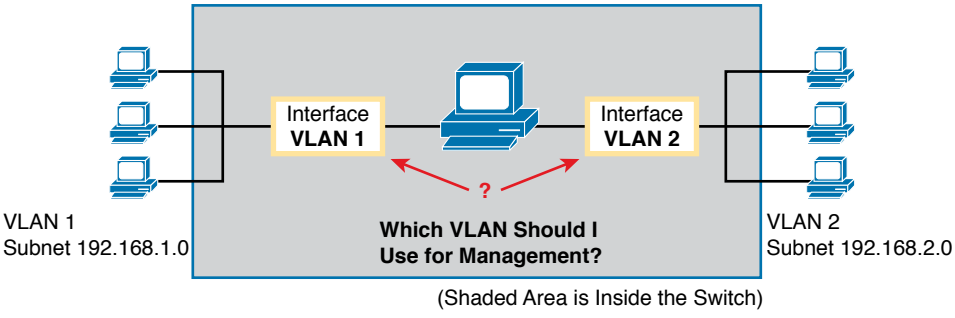


Figure 6-7 *Choosing One VLAN on Which to Configure a Switch IP Address*

Note that you should not try to use a VLAN interface for which there are no physical ports assigned to the same VLAN. If you do, the VLAN interface will not reach an up/up state, and the switch will not have the physical ability to communicate outside the switch.

NOTE Some Cisco switches can be configured to act as either a Layer 2 switch or a Layer 3 switch. When acting as a Layer 2 switch, a switch forwards Ethernet frames as discussed in depth in Chapter 5, "Analyzing Ethernet LAN Switching." Alternatively, a switch can also act as a *multilayer switch* or *Layer 3 switch*, which means the switch can do both Layer 2 switching and Layer 3 IP routing of IP packets, using the Layer 3 logic normally used by routers. This chapter assumes all switches are Layer 2 switches. Chapter 17, "IP Routing in the LAN," discusses Layer 3 switching in depth along with using multiple VLAN interfaces at the same time.

Configuring the IP address (and mask) on one VLAN interface allows the switch to send and receive IP packets with other hosts in a subnet that exists on that VLAN; however, the switch cannot communicate outside the local subnet without another configuration setting called the default gateway. The reason a switch needs a default gateway setting is the same reason that hosts need the same setting—because of how hosts think when sending IP packets. Specifically:

- To send IP packets to hosts in the same subnet, send them directly
- To send IP packets to hosts in a different subnet, send them to the local router; that is, the default gateway

Figure 6-8 shows the ideas. In this case, the switch (on the right) will use IP address 192.168.1.200 as configured on interface VLAN 1. However, to communicate with host A, on the far left of the figure, the switch must use Router R1 (the default gateway) to forward

IP packets to host A. To make that work, the switch needs to configure a default gateway setting, pointing to Router R1's IP address (192.168.1.1 in this case). Note that the switch and router both use the same mask, 255.255.255.0, which puts the addresses in the same subnet.

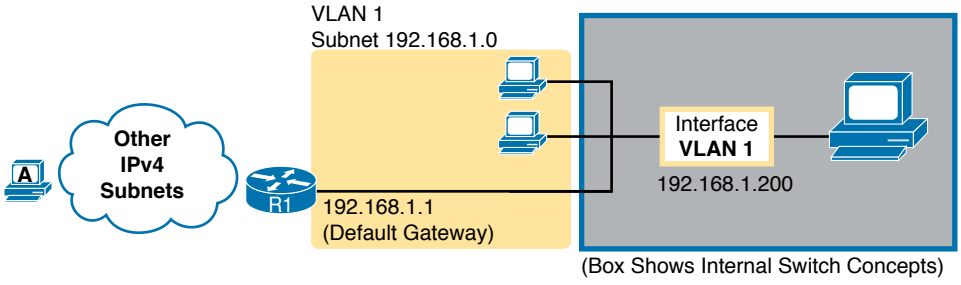


Figure 6-8 *The Need for a Default Gateway*

Configuring IPv4 on a Switch

A switch configures its IPv4 address and mask on this special NIC-like *VLAN interface*. The following steps list the commands used to configure IPv4 on a switch, assuming that the IP address is configured to be in VLAN 1, with Example 6-7 that follows showing an example configuration.

Config Checklist

Step 1. Use the **interface vlan 1** command in global configuration mode to enter interface VLAN 1 configuration mode.

Step 2. Use the **ip address** *ip-address mask* command in interface configuration mode to assign an IP address and mask.

Step 3. Use the **no shutdown** command in interface configuration mode to enable the VLAN 1 interface if it is not already enabled.

Step 4. Add the **ip default-gateway** *ip-address* command in global configuration mode to configure the default gateway.

Step 5. (Optional) Add the **ip name-server** *ip-address1 ip-address2 …* command in global configuration mode to configure the switch to use Domain Name System (DNS) to resolve names into their matching IP address.

Example 6-7 *Switch Static IP Address Configuration*

```
Emma# configure terminal
Emma(config)# interface vlan 1
Emma(config-if)# ip address 192.168.1.200 255.255.255.0
Emma(config-if)# no shutdown
00:25:07: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
00:25:08: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed
 state to up
Emma(config-if)# exit
Emma(config)# ip default-gateway 192.168.1.1
```

On a side note, this example shows a particularly important and common command: the [**no**] **shutdown** command. To administratively enable an interface on a switch, use the **no shutdown** interface subcommand; to disable an interface, use the **shutdown** interface subcommand. This command can be used on the physical Ethernet interfaces that the switch uses to switch Ethernet messages in addition to the VLAN interface shown here in this example.

Also, pause long enough to look at the messages that appear just below the **no shutdown** command in Example 6-7. Those messages are syslog messages generated by the switch stating that the switch did indeed enable the interface. Switches (and routers) generate syslog messages in response to a variety of events, and by default, those messages appear at the console. Chapter 9, "Device Management Protocols," in the *CCNA 200-301 Official Cert Guide, Volume 2*, discusses syslog messages in more detail.

Configuring a Switch to Learn Its IP Address with DHCP

The switch can also use Dynamic Host Configuration Protocol (DHCP) to dynamically learn its IPv4 settings. Basically, all you have to do is tell the switch to use DHCP on the interface and enable the interface. Assuming that DHCP works in this network, the switch will learn all its settings. The following list details the steps, again assuming the use of interface VLAN 1, with Example 6-8 that follows showing an example:

Config Checklist

Step 1. Enter VLAN 1 configuration mode using the **interface vlan 1** global configuration command, and enable the interface using the **no shutdown** command as necessary.

Step 2. Assign an IP address and mask using the **ip address dhcp** interface subcommand.

Example 6-8 *Switch Dynamic IP Address Configuration with DHCP*

```
Emma# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Emma(config)# interface vlan 1
Emma(config-if)# ip address dhcp
Emma(config-if)# no shutdown
Emma(config-if)# ^Z
Emma#
00:38:20: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
00:38:21: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to up
```

Verifying IPv4 on a Switch

The switch IPv4 configuration can be checked in several places. First, you can always look at the current configuration using the **show running-config** command. Second, you can look at the IP address and mask information using the **show interfaces vlan** *x* command, which shows detailed status information about the VLAN interface in VLAN *x*. Finally, if using DHCP, use the **show dhcp lease** command to see the (temporarily) leased IP address and other parameters. (Note that the switch does not store the DHCP-learned IP configuration in

the running-config file.) Example 6-9 shows sample output from these commands to match the configuration in Example 6-8.

Example 6-9 *Verifying DHCP-Learned Information on a Switch*

```
Emma# show dhcp lease
Temp IP addr: 192.168.1.101   for peer on Interface: Vlan1
Temp sub net mask: 255.255.255.0
   DHCP Lease server: 192.168.1.1, state: 3 Bound
   DHCP transaction id: 1966
   Lease: 86400 secs,  Renewal: 43200 secs,  Rebind: 75600 secs
Temp default-gateway addr: 192.168.1.1
   Next timer fires after: 11:59:45
   Retry count: 0   Client-ID: cisco-0019.e86a.6fc0-Vl1
   Hostname: Emma
Emma# show interfaces vlan 1
Vlan1 is up, line protocol is up
  Hardware is EtherSVI, address is 0019.e86a.6fc0 (bia 0019.e86a.6fc0)
  Internet address is 192.168.1.101/24
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
! lines omitted for brevity
Emma# show ip default-gateway
192.168.1.1
```

The output of the **show interfaces vlan 1** command lists two very important details related to switch IP addressing. First, this **show** command lists the interface status of the VLAN 1 interface—in this case, "up and up." If the VLAN 1 interface is not up, the switch cannot use its IP address to send and receive management traffic. Notably, if you forget to issue the **no shutdown** command, the VLAN 1 interface remains in its default shutdown state and is listed as "administratively down" in the **show** command output.

Second, note that the output lists the interface's IP address on the third line. If you statically configure the IP address, as in Example 6-7, the IP address will always be listed; however, if you use DHCP and DHCP fails, the **show interfaces vlan** *x* command will not list an IP address here. When DHCP works, you can see the IP address with the **show interfaces vlan 1** command, but that output does not remind you whether the address is either statically configured or DHCP leased. So it does take a little extra effort to make sure you know whether the address is statically configured or DHCP-learned on the VLAN interface.

Miscellaneous Settings Useful in the Lab

This last short section of the chapter touches on a couple of commands that can help you be a little more productive when practicing in a lab.

History Buffer Commands

When you enter commands from the CLI, the switch saves the last several commands in the history buffer. Then, as mentioned in Chapter 4, "Using the Command-Line Interface," you

can use the up-arrow key or press Ctrl+P to move back in the history buffer to retrieve a command you entered a few commands ago. This feature makes it very easy and fast to use a set of commands repeatedly. Table 6-2 lists some of the key commands related to the history buffer.

Table 6-2 Commands Related to the History Buffer

Command	Description
show history	An EXEC command that lists the commands currently held in the history buffer.
terminal history size *x*	From EXEC mode, this command allows a single user to set, just for this one login session, the size of his or her history buffer.
history size *x*	A configuration command that, from console or vty line configuration mode, sets the default number of commands saved in the history buffer for the users of the console or vty lines, respectively.

The logging synchronous, exec-timeout, and no ip domain-lookup Commands

These next three configuration commands have little in common, other than the fact that they can be useful settings to reduce your frustration when using the console of a switch or router.

The console automatically receives copies of all unsolicited syslog messages on a switch. The idea is that if the switch needs to tell the network administrator some important and possibly urgent information, the administrator might be at the console and might notice the message.

Unfortunately, IOS (by default) displays these syslog messages on the console's screen at any time—including right in the middle of a command you are entering, or in the middle of the output of a **show** command. Having a bunch of text show up unexpectedly can be a bit annoying.

You could simply disable the feature that sends these messages to the console and then re-enable the feature later using the **no logging console** and **logging console** global configuration commands. For example, when working from the console, if you want to temporarily not be bothered by log messages, you can disable the display of these messages with the **no logging console** global configuration command, and then when finished, enable them again.

However, IOS supplies a reasonable compromise, telling the switch to display syslog messages only at more convenient times, such as at the end of output from a **show** command. To do so, just configure the **logging synchronous** console line subcommand, which basically tells IOS to synchronize the syslog message display with the messages requested using **show** commands.

Another way to improve the user experience at the console is to control timeouts of the login session from the console or when using Telnet or SSH. By default, the switch automatically disconnects console and vty (Telnet and SSH) users after 5 minutes of inactivity. The **exec-timeout** *minutes seconds* line subcommand enables you to set the length of that inactivity timer. In the lab (but not in production), you might want to use the special value of 0 minutes and 0 seconds meaning "never time out."

Finally, IOS has an interesting combination of features that can make you wait for a minute or so when you mistype a command. First, IOS tries to use DNS name resolution on IP hostnames—a generally useful feature. If you mistype a command, however, IOS thinks you want to telnet to a host by that name. With all default settings in the switch, the switch tries to resolve the hostname, cannot find a DNS server, and takes about a minute to time out and give you control of the CLI again.

To avoid this problem, configure the **no ip domain-lookup** global configuration command, which disables IOS's attempt to resolve the hostname into an IP address.

Example 6-10 collects all these commands into a single example, as a template for some good settings to add in a lab switch to make you more productive.

Example 6-10 *Commands Often Used in the Lab to Increase Productivity*

```
no ip domain-lookup
!
line console 0
 exec-timeout 0 0
 logging synchronous
 history size 20
!
line vty 0 15
 exec-timeout 0 0
 logging synchronous
 history size 20
```

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element section titled "Step 2: Build Your Study Habits Around the Chapter" for more details. Table 6-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 6-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review config checklists		Book, website
Do labs		Sim Lite, blog
Review command tables		Book

Review All the Key Topics

Key Topic

Table 6-4 Key Topics for Chapter 6

Key Topic Element	Description	Page Number
Example 6-2	Example of configuring password login security (no usernames)	132
Figure 6-5	SSH configuration commands with related username login security	137

Key Terms You Should Know

Telnet, Secure Shell (SSH), local username, AAA, AAA server, enable mode, default gateway, VLAN interface, history buffer, DNS, name resolution, log message

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included with this book for free. The subset of labs mostly relate to this part. Take the time to try some of the labs. As always, also check the author's blog site pages for configuration exercises (Config Labs) at https://blog.certskills.com.

Command References

Tables 6-5, 6-6, 6-7, and 6-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 6-5 Login Security Commands

Command	Mode/Purpose/Description
line console 0	Changes the context to console configuration mode.
line vty *1st-vty last-vty*	Changes the context to vty configuration mode for the range of vty lines listed in the command.
login	Console and vty configuration mode. Tells IOS to prompt for a password.
password *pass-value*	Console and vty configuration mode. Lists the password required if the **login** command (with no other parameters) is configured.
login local	Console and vty configuration mode. Tells IOS to prompt for a username and password, to be checked against locally configured **username** global configuration commands on this switch or router.
username *name* **secret** *pass-value*	Global command. Defines one of possibly multiple usernames and associated passwords, used for user authentication. Used when the **login local** line configuration command has been used.

Command	Mode/Purpose/Description
crypto key generate rsa [**modulus** *360..2048*]	Global command. Creates and stores (in a hidden location in flash memory) the keys required by SSH.
transport input {**telnet** | **ssh** | **all** | **none**}	vty line configuration mode. Defines whether Telnet/SSH access is allowed into this switch. Both values can be configured on one command to allow both Telnet and SSH access (the default).

Table 6-6 Switch IPv4 Configuration

Command	Mode/Purpose/Description
interface vlan *number*	Changes the context to VLAN interface mode. For VLAN 1, allows the configuration of the switch's IP address.
ip address *ip-address subnet-mask*	VLAN interface mode. Statically configures the switch's IP address and mask.
ip address dhcp	VLAN interface mode. Configures the switch as a DHCP client to discover its IPv4 address, mask, and default gateway.
ip default-gateway *address*	Global command. Configures the switch's default gateway IPv4 address. Not required if the switch uses DHCP.
ip name-server *server-ip-1 server-ip-2 …*	Global command. Configures the IPv4 addresses of DNS servers, so any commands when logged in to the switch will use the DNS for name resolution.

Table 6-7 Other Switch Configuration

Command	Mode/Purpose/Description
hostname *name*	Global command. Sets this switch's hostname, which is also used as the first part of the switch's command prompt.
enable secret *pass-value*	Global command. Sets this switch's password that is required for any user to reach enable mode.
history size *length*	Line config mode. Defines the number of commands held in the history buffer, for later recall, for users of those lines.
logging synchronous	Console or vty mode. Tells IOS to send log messages to the user at natural break points between commands rather than in the middle of a line of output.
[no] logging console	Global command that disables or enables the display of log messages to the console.
exec-timeout *minutes* [*seconds*]	Console or vty mode. Sets the inactivity timeout, so that after the defined period of no action, IOS closes the current user login session.

Table 6-8 Chapter 6 EXEC Command Reference

Command	Purpose
show running-config	Lists the currently used configuration.
show running-config | begin line vty	Pipes (sends) the command output to the **begin** command, which only lists output beginning with the first line that contains the text "line vty."
show dhcp lease	Lists any information the switch acquires as a DHCP client. This includes IP address, subnet mask, and default gateway information.
show crypto key mypubkey rsa	Lists the public and shared key created for use with SSH using the **crypto key generate rsa** global configuration command.
show ip ssh	Lists status information for the SSH server, including the SSH version.
show ssh	Lists status information for current SSH connections into and out of the local switch.
show interfaces vlan *number*	Lists the interface status, the switch's IPv4 address and mask, and much more.
show ip default-gateway	Lists the switch's setting for its IPv4 default gateway.
terminal history size *x*	Changes the length of the history buffer for the current user only, only for the current login to the switch.
show history	Lists the commands in the current history buffer.

CHAPTER 7

Configuring and Verifying Switch Interfaces

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.b L2 and L3 switches

1.4 Describe switching concepts

So far in this part, you have learned the skills to navigate the command-line interface (CLI) and use commands that configure and verify switch features. You learned about the primary purpose of a switch—forwarding Ethernet frames—and learned how to see that process in action by looking at the switch MAC address table. After learning about the switch data plane in Chapter 5, "Analyzing Ethernet LAN Switching," you learned a few management plane features in Chapter 6, "Configuring Basic Switch Management," like how to configure the switch to support Telnet and Secure Shell (SSH) by configuring IP address and login security.

This chapter focuses on switch interfaces in two major sections. The first section shows how you can configure and change the operation of switch interfaces: how to change the speed, duplex, or even disable the interface. The second half then focuses on how to use show commands on a switch to verify switch interface status and how to interpret the output to find some of the more common issues with switch interfaces.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 7-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Configuring Switch Interfaces	1–3
Analyzing Switch Interface Status and Statistics	4–6

1. Which of the following describes a way to disable IEEE standard autonegotiation on a 10/100 port on a Cisco switch?
   a. Configure the **negotiate disable** interface subcommand
   b. Configure the **no negotiate** interface subcommand
   c. Configure the **speed 100** interface subcommand
   d. Configure the **duplex half** interface subcommand
   e. Configure the **duplex full** interface subcommand
   f. Configure the **speed 100** and **duplex full** interface subcommands
2. In which of the following modes of the CLI could you configure the duplex setting for interface Fast Ethernet 0/5?
   a. User mode
   b. Enable mode
   c. Global configuration mode
   d. VLAN mode
   e. Interface configuration mode
3. A Cisco Catalyst switch connects with its Gigabit0/1 port to an end user's PC. The end user, thinking the user is helping, manually sets the PC's OS to use a speed of 1000 Mbps and to use full duplex, and disables the use of autonegotiation. The switch's G0/1 port has default settings for speed and duplex. What speed and duplex settings will the switch decide to use? (Choose two answers.)
   a. Full duplex
   b. Half duplex
   c. 10 Mbps
   d. 1000 Mbps
4. The output of the **show interfaces status** command on a 2960 switch shows interface Fa0/1 in a "disabled" state. Which of the following is true about interface Fa0/1? (Choose three answers.)
   a. The interface is configured with the **shutdown** command.
   b. The **show interfaces fa0/1** command will list the interface with two status codes of administratively down and line protocol down.
   c. The **show interfaces fa0/1** command will list the interface with two status codes of up and down.
   d. The interface cannot currently be used to forward frames.
   e. The interface can currently be used to forward frames.

5. Switch SW1 uses its Gigabit 0/1 interface to connect to switch SW2's Gigabit 0/2 interface. SW2's Gi0/2 interface is configured with the **speed 1000** and **duplex full** commands. SW1 uses all defaults for interface configuration commands on its Gi0/1 interface. Which of the following are true about the link after it comes up? (Choose two answers.)
   a. The link works at 1000 Mbps (1 Gbps).
   b. SW1 attempts to run at 10 Mbps because SW2 has effectively disabled IEEE standard autonegotiation.
   c. The link runs at 1 Gbps, but SW1 uses half duplex and SW2 uses full duplex.
   d. Both switches use full duplex.
6. Switch SW1 connects via a cable to switch SW2's G0/1 port. Which of the following conditions is the most likely to cause SW1's late collision counter to continue to increment?
   a. SW2's G0/1 has been configured with a **shutdown** interface subcommand.
   b. The two switches have been configured with different values on the **speed** interface subcommand.
   c. A duplex mismatch exists with SW1 set to full duplex.
   d. A duplex mismatch exists with SW1 set to half duplex.

Foundation Topics

Configuring Switch Interfaces

IOS uses the term *interface* to refer to physical ports used to forward data to and from other devices. Each interface can be configured with several settings, each of which might differ from interface to interface. IOS uses interface subcommands to configure these settings. Each of these settings may be different from one interface to the next, so you would first identify the specific interface, and then configure the specific setting.

This section begins with a discussion of three relatively basic per-interface settings: the port speed, duplex, and a text description. Following that, the text takes a short look at a pair of the most common interface subcommands: the **shutdown** and **no shutdown** commands, which administratively disable and enable the interface, respectively. This section ends with a discussion about autonegotiation concepts, which in turn dictates what settings a switch chooses to use when using autonegotiation.

Configuring Speed, Duplex, and Description

Switch interfaces that support multiple speeds (10/100 and 10/100/1000 interfaces), by default, will autonegotiate what speed to use. However, you can configure the speed and duplex settings with the **duplex {auto | full | half}** and **speed** {**auto | 10 | 100 | 1000**} interface subcommands. Simple enough.

Most of the time, using autonegotiation makes good sense, so when you set the duplex and speed manually using these commands, you typically have a good reason to do so. For instance, maybe you want to set the speed to the fastest possible on links between switches just to avoid the chance that autonegotiation chooses a slower speed.

The **description** text interface subcommand lets you add a text description to the interface. For instance, if you have good reason to configure the speed and duplex on a port, maybe add a description that says why you did. Example 7-1 shows how to configure **duplex** and **speed**, as well as the **description** command, which is simply a text description that can be configured by the administrator.

Key Topic

Example 7-1 *Configuring* **speed**, **duplex**, *and* **description** *on Switch Emma*

```
Emma# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Emma(config)# interface FastEthernet 0/1
Emma(config-if)# duplex full
Emma(config-if)# speed 100
Emma(config-if)# description Printer on 3rd floor, Preset to 100/full
Emma(config-if)# exit
Emma(config)# interface range FastEthernet 0/11 - 20
Emma(config-if-range)# description end-users connect here
Emma(config-if-range)# ^Z
Emma#
```

First, focus on the mechanics of moving around in configuration mode again by looking closely at the command prompts. The various **interface** commands move the user from global mode into interface configuration mode for a specific interface. For instance, the example configures the **duplex**, **speed**, and **description** commands all just after the **interface FastEthernet 0/1** command, which means that all three of those configuration settings apply to interface Fa0/1, and not to the other interfaces.

The **show interfaces status** command lists much of the detail configured in Example 7-1, even with only one line of output per interface. Example 7-2 shows an example, just after the configuration in Example 7-1 was added to the switch.

Example 7-2 *Displaying Interface Status*

```
Emma# show interfaces status
Port      Name                 Status       Vlan      Duplex  Speed Type
Fa0/1     Printer on 3rd floo  notconnect   1           full    100 10/100BaseTX
Fa0/2                          notconnect   1           auto   auto 10/100BaseTX
Fa0/3                          notconnect   1           auto   auto 10/100BaseTX
Fa0/4                          connected    1         a-full  a-100 10/100BaseTX
Fa0/5                          notconnect   1           auto   auto 10/100BaseTX
Fa0/6                          connected    1         a-full  a-100 10/100BaseTX
Fa0/7                          notconnect   1           auto   auto 10/100BaseTX
Fa0/8                          notconnect   1           auto   auto 10/100BaseTX
Fa0/9                          notconnect   1           auto   auto 10/100BaseTX
Fa0/10                         notconnect   1           auto   auto 10/100BaseTX
Fa0/11    end-users connect    notconnect   1           auto   auto 10/100BaseTX
Fa0/12    end-users connect    notconnect   1           auto   auto 10/100BaseTX
Fa0/13    end-users connect    notconnect   1           auto   auto 10/100BaseTX
Fa0/14    end-users connect    notconnect   1           auto   auto 10/100BaseTX
```

```
Fa0/15     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/16     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/17     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/18     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/19     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/20     end-users connect   notconnect   1           auto   auto 10/100BaseTX
Fa0/21                         notconnect   1           auto   auto 10/100BaseTX
Fa0/22                         notconnect   1           auto   auto 10/100BaseTX
Fa0/23                         notconnect   1           auto   auto 10/100BaseTX
Fa0/24                         notconnect   1           auto   auto 10/100BaseTX
Gi0/1                          notconnect   1           auto   auto 10/100/1000BaseTX
Gi0/2                          notconnect   1           auto   auto 10/100/1000BaseTX
```

Working through the output in the example:

FastEthernet 0/1 (Fa0/1): This output lists the first few characters of the configured description. It also lists the configured speed of 100 and duplex full per the **speed** and **duplex** commands in Example 7-1. However, it also states that Fa0/1 has a status of notconnect, meaning that the interface is not currently working. (That switch port did not have a cable connected when collecting this example, on purpose.)

FastEthernet 0/2 (Fa0/2): Example 7-1 did not configure this port at all. This port had all default configuration. Note that the "auto" text under the speed and duplex heading means that this port will attempt to autonegotiate both settings when the port comes up. However, this port also does not have a cable connected (again on purpose, for comparison).

FastEthernet 0/4 (Fa0/4): Like Fa0/2, this port has all default configuration but was cabled to another working device to give yet another contrasting example. This device completed the autonegotiation process, so instead of "auto" under the speed and duplex headings, the output lists the negotiated speed and duplex (**a-full** and **a-100**). Note that the text includes the **a-** to mean that the listed speed and duplex values were autonegotiated.

Configuring Multiple Interfaces with the interface range Command

The bottom of the configuration in Example 7-1 shows a way to shorten your configuration work when making the same setting on multiple consecutive interfaces. To do so, use the **interface range** command. In the example, the **interface range FastEthernet 0/11 - 20** command tells IOS that the next subcommand(s) apply to interfaces Fa0/11 through Fa0/20. You can define a range as long as all interfaces are the same type and are numbered consecutively.

NOTE This book spells out all parameters fully to avoid confusion. However, most everyone abbreviates what they type in the CLI to the shortest unique abbreviation. For instance, the configuration commands **int f0/1** and **int ran f0/11 - 20** would also be acceptable.

IOS does not actually put the **interface range** command into the configuration. Instead, it acts as if you had typed the subcommand under every single interface in the specified

Answers to the "Do I Know This Already?" quiz:

1 F **2** E **3** A, D **4** A, B, D **5** A, D **6** D

range. Example 7-3 shows an excerpt from the **show running-config** command, listing the configuration of interfaces F0/11–12 from the configuration in Example 7-1. The example shows the same description command on both interfaces; to save space, the example does not bother to show all 10 interfaces that have the same description text.

Example 7-3 *How IOS Expands the Subcommands Typed After* **interface range**

```
Emma# show running-config
! Lines omitted for brevity
interface FastEthernet0/11
 description end-users connect here
!
interface FastEthernet0/12
 description end-users connect here
! Lines omitted for brevity
```

Administratively Controlling Interface State with shutdown

As you might imagine, network engineers need a way to bring down an interface without having to travel to the switch and remove a cable. In short, we need to be able to decide which ports should be enabled and which should be disabled.

In an odd turn of phrase, Cisco uses two interface subcommands to configure the idea of administratively enabling and disabling an interface: the **shutdown** command (to disable) and the **no shutdown** command (to enable). While the **no shutdown** command might seem like an odd command to enable an interface at first, you will use this command a lot in the lab, and it will become second nature. (Most people, in fact, use the abbreviations **shut** and **no shut**.)

Example 7-4 shows an example of disabling an interface using the **shutdown** interface subcommand. In this case, switch SW1 has a working interface F0/1. The user connects at the console and disables the interface. IOS generates a log message each time an interface fails or recovers, and log messages appear at the console, as shown in the example.

Key Topic

Example 7-4 *Administratively Disabling an Interface with* **shutdown**

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# interface fastEthernet 0/1
SW1(config-if)# shutdown
SW1(config-if)#
*Mar 2 03:02:19.701: %LINK-5-CHANGED: Interface FastEthernet0/1, changed state to
administratively down
*Mar 2 03:02:20.708: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to down
```

To bring the interface back up again, all you have to do is follow the same process but use the **no shutdown** command instead.

Before leaving the simple but oddly named **shutdown/no shutdown** commands, take a look at two important show commands that list the status of a shutdown interface. The **show**

interfaces status command lists one line of output per interface, and when shut down, lists the interface status as "disabled." That makes logical sense to most people. The **show interfaces** command (without the **status** keyword) lists many lines of output per interface, giving a much more detailed picture of interface status and statistics. With that command, the interface status comes in two parts, with one part using the phrase "administratively down," matching the highlighted log message in Example 7-4.

Example 7-5 shows an example of each of these commands. Note that both examples also use the F0/1 parameter (short for Fast Ethernet0/1), which limits the output to the messages about F0/1 only. Also note that F0/1 is still shut down at this point.

Example 7-5 *The Different Status Information About Shutdown in Two Different* **show** *Commands*

```
SW1# show interfaces f0/1 status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        disabled     1            auto   auto 10/100BaseTX

SW1# show interfaces f0/1
FastEthernet0/1 is administratively down, line protocol is down (disabled)
  Hardware is Fast Ethernet, address is 1833.9d7b.0e81 (bia 1833.9d7b.0e81)
  MTU 1500 bytes, BW 10000 Kbit/sec, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Auto-duplex, Auto-speed, media type is 10/100BaseTX
  input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output 00:00:36, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     164 packets input, 13267 bytes, 0 no buffer
     Received 164 broadcasts (163 multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 163 multicast, 0 pause input
     0 input packets with dribble condition detected
     66700 packets output, 5012302 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

Removing Configuration with the no Command

One purpose for the specific commands shown in Part II of the book is to teach you about that command. In some cases, the commands are not the end goal, and the text is attempting to teach you something about how the CLI works. This next short topic is more about the process than about the commands.

With some IOS configuration commands (but not all), you can revert to the default setting by issuing a **no** version of the command. What does that mean? Let me give you a few examples:

- If you earlier had configured **speed 100** on an interface, the **no speed** command on that same interface reverts to the default speed setting (which happens to be **speed auto**).
- Same idea with the **duplex** command: an earlier configuration of **duplex half** or **duplex full**, followed by **no duplex** on the same interface, reverts the configuration back to the default of duplex auto.
- If you had configured a **description** command with some text, to go back to the default state of having no **description** command at all for that interface, use the **no description** command.

Example 7-6 shows the process. In this case, switch SW1's F0/2 port has been configured with **speed 100**, **duplex half**, **description link to 2901-2**, and **shutdown**. You can see evidence of all four settings in the command that begins the example. (This command lists the running-config, but only the part for that one interface.) The example then shows the **no** versions of those commands and closes with a confirmation that all the commands have reverted to default.

Example 7-6 *Removing Various Configuration Settings Using the* **no** *Command*

```
SW1# show running-config interface f0/2
Building configuration...

Current configuration : 95 bytes
!
interface FastEthernet0/2
 description link to 2901-2
 shutdown
 speed 100
 duplex half
end

SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# interface fastethernet 0/2
SW1(config-if)# no speed
SW1(config-if)# no duplex
SW1(config-if)# no description
SW1(config-if)# no shutdown
```

```
SW1(config-if)# ^Z
SW1#
SW1# show running-config interface f0/2
Building configuration...
Current configuration : 33 bytes
!
interface FastEthernet0/2
end
SW1#
```

NOTE The **show running-config** and **show startup-config** commands typically do not display default configuration settings, so the absence of commands listed under interface F0/2 at the end of the example means that those commands now use default values.

Autonegotiation

For any 10/100 or 10/100/1000 interfaces—that is, interfaces that can run at different speeds—Cisco Catalyst switches default to a setting of **duplex auto** and **speed auto**. As a result, those interfaces attempt to automatically determine the speed and duplex setting to use. Alternatively, you can configure most devices, switch interfaces included, to use a specific speed and/or duplex.

In practice, using autonegotiation is easy: just leave the speed and duplex at the default setting, and let the switch port negotiate what settings to use on each port. However, problems can occur due to unfortunate combinations of configuration. Therefore, this next topic walks through more detail about the concepts behind autonegotiation, so you know better how to interpret the meaning of the switch **show** commands and when to choose to use a particular configuration setting.

Autonegotiation Under Working Conditions

Ethernet devices on the ends of a link must use the same standard; otherwise, they cannot correctly send data. For example, a NIC cannot use 100BASE-T, which uses a two-pair UTP cable with a 100-Mbps speed, while the switch port on the other end of the link uses 1000BASE-T. Even if you used a cable that works with Gigabit Ethernet, the link would not work with one end trying to send at 100 Mbps while the other tried to receive the data at 1000 Mbps.

Upgrading to new and faster Ethernet standards becomes a problem because both ends have to use the same standard. For example, if you replace an old PC with a new one, the old one might have been using 100BASE-T while the new one uses 1000BASE-T. The switch port on the other end of the link needs to now use 1000BASE-T, so you upgrade the switch. If that switch had ports that would use only 1000BASE-T, you would need to upgrade all the other PCs connected to the switch. So, having both PC network interface cards (NIC) and switch ports that support multiple standards/speeds makes it much easier to migrate to the next better standard.

The IEEE autonegotiation protocol helps makes it much easier to operate a LAN when NICs and switch ports support multiple speeds. IEEE autonegotiation (IEEE standard 802.3u) defines a protocol that lets the two UTP-based Ethernet nodes on a link negotiate so that they each choose to use the same speed and duplex settings. The protocol messages flow outside the normal Ethernet electrical frequencies as out-of-band signals over the UTP cable. Basically, each node states what it can do, and then each node picks the best options that both nodes support: the fastest speed and the best duplex setting, with full duplex being better than half duplex.

NOTE Autonegotiation relies on the fact that the IEEE uses the same wiring pinouts for 10BASE-T and 100BASE-T, and that 1000BASE-T simply adds to those pinouts, adding two pairs.

Many networks use autonegotiation every day, particularly between user devices and the access layer LAN switches, as shown in Figure 7-1. The company installed four-pair cabling of the right quality to support 1000BASE-T, to be ready to support Gigabit Ethernet. As a result, the wiring supports 10-Mbps, 100-Mbps, and 1000-Mbps Ethernet options. Both nodes on each link send autonegotiation messages to each other. The switch in this case has all 10/100/1000 ports, while the PC NICs support different options.

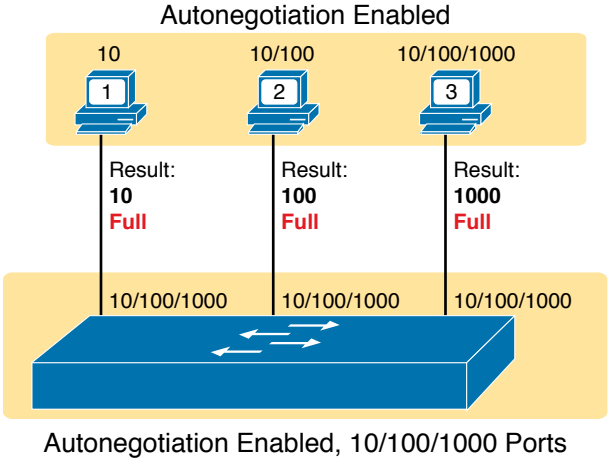


Figure 7-1 *IEEE Autonegotiation Results with Both Nodes Working Correctly*

The following list breaks down the logic, one PC at a time:

PC1: The switch port claims it can go as fast as 1000 Mbps, but PC1's NIC claims a top speed of 10 Mbps. Both the PC and the switch choose the fastest speed that each supports (10 Mbps) and the best duplex that each supports (full).

PC2: PC2 claims a best speed of 100 Mbps, which means it can use 10BASE-T or 100BASE-T. The switch port and NIC negotiate to use the best speed of 100 Mbps and full duplex.

PC3: It uses a 10/100/1000 NIC, supporting all three speeds and standards, so both the NIC and switch port choose 1000 Mbps and full duplex.

Autonegotiation Results When Only One Node Uses Autonegotiation

Figure 7-1 shows the IEEE autonegotiation results when both nodes use the process. However, most Ethernet devices can disable autonegotiation, so it is just as important to know what happens when a node tries to use autonegotiation but the node gets no response.

Disabling autonegotiation is not always a bad idea. For instance, many network engineers disable autonegotiation on links between switches and simply configure the desired speed and duplex on both switches. However, mistakes can happen when one device on an Ethernet predefines speed and duplex (and disables autonegotiation), while the device on the other end attempts autonegotiation. In that case, the link might not work at all, or it might just work poorly.

NOTE Configuring both the speed and duplex on a Cisco Catalyst switch interface disables autonegotiation.

IEEE autonegotiation defines some rules (defaults) that nodes should use as defaults when autonegotiation fails—that is, when a node tries to use autonegotiation but hears nothing from the device. The rules:

- **Speed:** Use your slowest supported speed (often 10 Mbps).
- **Duplex:** If your speed = 10 or 100, use half duplex; otherwise, use full duplex.

Cisco switches can make a better choice than that base IEEE speed default because Cisco switches can actually sense the speed used by other nodes, even without IEEE autonegotiation. As a result, Cisco switches use this slightly different logic to choose the speed when autonegotiation fails:

Key Topic

- **Speed:** Sense the speed (without using autonegotiation), but if that fails, use the IEEE default (slowest supported speed, often 10 Mbps).
- **Duplex:** Use the IEEE defaults: If speed = 10 or 100, use half duplex; otherwise, use full duplex.

NOTE Ethernet interfaces using speeds faster than 1 Gbps always use full duplex.

Figure 7-2 shows three examples in which three users change their NIC settings and disable autonegotiation, while the switch (with all 10/100/1000 ports) attempts autonegotiation. That is, the switch ports all default to **speed auto** and **duplex auto**. The top of the figure shows the configured settings on each PC NIC, with the choices made by the switch listed next to each switch port.

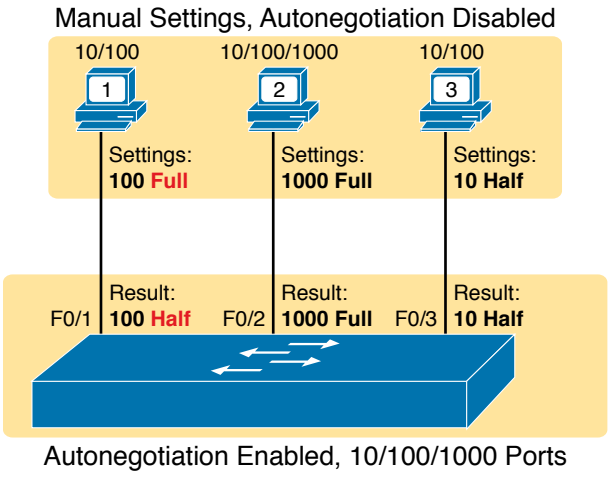


Figure 7-2 *IEEE Autonegotiation Results with Autonegotiation Disabled on One Side*

Reviewing each link, left to right:

- **PC1:** The switch receives no autonegotiation messages, so it senses the electrical signal to learn that PC1 is sending data at 100 Mbps. The switch uses the IEEE default duplex based on the 100 Mbps speed (half duplex).
- **PC2:** The switch uses the same steps and logic as with the link to PC1, except that the switch chooses to use full duplex because the speed is 1000 Mbps.
- **PC3:** The user picks poorly, choosing the slower speed (10 Mbps) and the worse duplex setting (half). However, the Cisco switch senses the speed without using IEEE autonegotiation and then uses the IEEE duplex default for 10-Mbps links (half duplex).

PC1 shows a classic and unfortunately common end result: a *duplex mismatch*. The two nodes (PC1 and SW1's port G0/1) both use 100 Mbps, so they can send data. However, PC1, using full duplex, does not attempt to use carrier sense multiple access with collision detection (CSMA/CD) logic and sends frames at any time. Switch port F0/1, with half duplex, does use CSMA/CD. As a result, switch port F0/1 will believe collisions occur on the link, even if none physically occur. The switch port will stop transmitting, back off, resend frames, and so on. As a result, the link is up, but it performs poorly. The upcoming section titled "Interface Speed and Duplex Issues" will revisit this problem with a focus on how to recognize the symptoms of a duplex mismatch.

Autonegotiation and LAN Hubs

LAN hubs also impact how autonegotiation works. Basically, hubs do not react to autonegotiation messages, and they do not forward the messages. As a result, devices connected to a hub must use the IEEE rules for choosing default settings, which often results in the devices using 10 Mbps and half duplex.

Figure 7-3 shows an example of a small Ethernet LAN that uses a 20-year-old 10BASE-T hub. In this LAN, all devices and switch ports are 10/100/1000 ports. The hub supports only 10BASE-T.

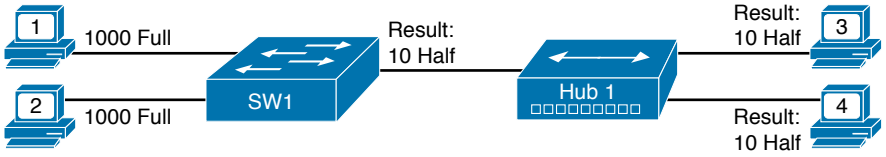


Figure 7-3 *IEEE Autonegotiation with a LAN Hub*

Note that the devices on the right need to use half duplex because the hub requires the use of the CSMA/CD algorithm to avoid collisions.

NOTE If you would like to learn more about collision domains and the impact of these older LAN hubs, look to the companion website for Appendix K, "Analyzing Ethernet LAN Designs," to the section titled "Ethernet Collision Domains."

Analyzing Switch Interface Status and Statistics

Now that you have seen some of the ways to configure switch interfaces, the rest of the chapter takes a closer look at how to verify the interfaces work correctly. This section also looks at those more unusual cases in which the interface is working but not working well, as revealed by different interface status codes and statistics.

Interface Status Codes and Reasons for Nonworking States

Cisco switches actually use two different sets of interface status codes—one set of two codes (words) that use the same conventions as do router interface status codes, and another set with a single code (word). Both sets of status codes can determine whether an interface is working.

The switch **show interfaces** and **show interfaces description** commands list the two-code status named the *line status* and *protocol status*. The line status *generally* refers to whether Layer 1 is working, with protocol status generally referring to whether Layer 2 is working.

NOTE This book refers to these two status codes in shorthand by just listing the two codes with a slash between them, such as *up/up*.

The single-code interface status corresponds to different combinations of the traditional two-code interface status codes and can be easily correlated to those codes. For example, the **show interfaces status** command lists a single-word state of *connected* state for working interfaces, with the same meaning as the two-word *up/up* state seen with the **show interfaces** and **show interfaces description** commands. Table 7-2 lists the code combinations and some root causes that could have caused a particular interface status.

Key Topic

Table 7-2 LAN Switch Interface Status Codes

Line Status	Protocol Status	Interface Status	Typical Root Cause
administratively down	down	disabled	The **shutdown** command is configured on the interface.
down	down	notconnect	No cable; bad cable; wrong cable pinouts; speed mismatch; neighboring device is (a) powered off, (b) **shutdown**, or (c) error disabled.
up	down	notconnect	Not expected on LAN switch physical interfaces.
down	down (err-disabled)	err-disabled	Port security has disabled the interface.
up	up	connected	The interface is working.

Examining the notconnect state for a moment, note that this state has many causes that have been mentioned through this book. For example, using incorrect cabling pinouts, instead of the correct pinouts explained in Chapter 2, "Fundamentals of Ethernet LANs," causes a problem. However, one topic can be particularly difficult to troubleshoot—the possibility for both speed and duplex mismatches, as explained in the next section.

As you can see in the table, having a bad cable is just one of many reasons for the down/down state (or notconnect, per the **show interfaces status** command). Some examples of the root causes of cabling problems include the following:

- The installation of any equipment that uses electricity, even non-IT equipment, can interfere with the transmission on the cabling and make the link fail.
- The cable could be damaged, for example, if it lies under carpet. If the user's chair keeps squashing the cable, eventually the electrical signal can degrade.
- Although optical cables do not suffer from electromagnetic interference (EMI), someone can try to be helpful and move a fiber-optic cable out of the way—bending it too much. A bend into too tight a shape can prevent the cable from transmitting bits (called *macro-bending*).

For the other interface states listed in Table 7-2, only the up/up (connected) state needs more discussion. An interface can be in a working state, and it might really be working—or it might be working in a degraded state. The next few topics discuss how to examine an up/up (connected) interface to find out whether it is working well or having problems.

Interface Speed and Duplex Issues

To discuss some of the speed and duplex issues, first consider the output from the **show interfaces status** and **show interfaces** commands as demonstrated in Example 7-7. The first of these commands lists a one-line summary of the interface status, while the second command gives many details—but surprisingly, the briefer **show interfaces status** command tells us more about autonegotiation.

Key Topic

Example 7-7 *Displaying Speed and Duplex Settings on Switch Interfaces*

```
SW1# show interfaces status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        notconnect   1            auto   auto 10/100BaseTX
Fa0/2                        notconnect   1            auto   auto 10/100BaseTX
Fa0/3                        notconnect   1            auto   auto 10/100BaseTX
Fa0/4                        connected    1          a-full  a-100 10/100BaseTX
Fa0/5                        connected    1          a-full  a-100 10/100BaseTX
Fa0/6                        notconnect   1            auto   auto 10/100BaseTX
Fa0/7                        notconnect   1            auto   auto 10/100BaseTX
Fa0/8                        notconnect   1            auto   auto 10/100BaseTX
Fa0/9                        notconnect   1            auto   auto 10/100BaseTX
Fa0/10                       notconnect   1            auto   auto 10/100BaseTX
Fa0/11                       connected    1          a-full     10 10/100BaseTX
Fa0/12                       connected    1            half    100 10/100BaseTX
Fa0/13                       connected    1          a-full  a-100 10/100BaseTX
Fa0/14                       disabled     1            auto   auto 10/100BaseTX
! Lines omitted for brevity

SW1# show interfaces fa0/13
FastEthernet0/13 is up, line protocol is up (connected)
   Hardware is Fast Ethernet, address is 0019.e86a.6f8d (bia 0019.e86a.6f8d)
   MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
      reliability 255/255, txload 1/255, rxload 1/255
   Encapsulation ARPA, loopback not set
   Keepalive set (10 sec)
   Full-duplex, 100Mbps, media type is 10/100BaseTX
   input flow-control is off, output flow-control is unsupported
   ARP type: ARPA, ARP Timeout 04:00:00
   Last input 00:00:05, output 00:00:00, output hang never
   Last clearing of "show interface" counters never
   Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
   Queueing strategy: fifo
   Output queue: 0/40 (size/max)
   5 minute input rate 0 bits/sec, 0 packets/sec
   5 minute output rate 0 bits/sec, 0 packets/sec
      85022 packets input, 10008976 bytes, 0 no buffer
      Received 284 broadcasts (0 multicast)
      0 runts, 0 giants, 0 throttles
      0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
      0 watchdog, 281 multicast, 0 pause input
      0 input packets with dribble condition detected
      95226 packets output, 10849674 bytes, 0 underruns
      0 output errors, 0 collisions, 1 interface resets
```

```
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 PAUSE output
     0 output buffer failures, 0 output buffers swapped out
```

Although both commands in the example can be useful, only the **show interfaces status** command implies how the switch determined the speed and duplex settings. The command output lists autonegotiated settings with a prefix of **a-** and the manually set values without the **a-** prefix.

For example, consider ports Fa0/12 and Fa0/13 in the output of the **show interfaces status** command. For Fa0/13, **a-full** means full duplex as autonegotiated, whereas **half** on Fa0/12 means half duplex but as manually configured. The example shades the command output that implies that the switch's Fa0/12 interface's speed and duplex were not found through autonegotiation, but Fa0/13 did use autonegotiation.

In comparison, note that the **show interfaces fa0/13** command (without the **status** option) simply lists the speed and duplex for interface Fast Ethernet 0/13, with nothing implying that the values were learned through autonegotiation.

When the IEEE autonegotiation process works on both devices—that is, both are sending autonegotiation messages—both devices agree to the fastest speed and best duplex supported by both devices. However, when one device uses autonegotiation and the other disables it, the first device must resort to default settings as detailed earlier in section "Autonegotiation Results When Only One Node Uses Autonegotiation." As a reminder, those defaults are

Key Topic

- **Speed:** Sense the speed (without using autonegotiation), but if that fails, use the IEEE default (slowest supported speed, often 10 Mbps).
- **Duplex:** Use the IEEE defaults: If speed = 10 or 100, use half duplex; otherwise, use full duplex.

When a switch must use its defaults, it should get the speed correct, but it may choose the wrong duplex setting, creating a duplex mismatch.

For example, in Figure 7-4, imagine that SW2's Gi0/2 interface was configured with the **speed 100** and **duplex full** commands (these settings are not recommended on a Gigabit-capable interface, by the way). On Cisco switches, configuring both the **speed** and **duplex** commands disables IEEE autonegotiation on that port. If SW1's Gi0/1 interface tries to use autonegotiation, SW1 would also use a speed of 100 Mbps, but default to use half duplex. Example 7-8 shows the results of this specific case on SW1.

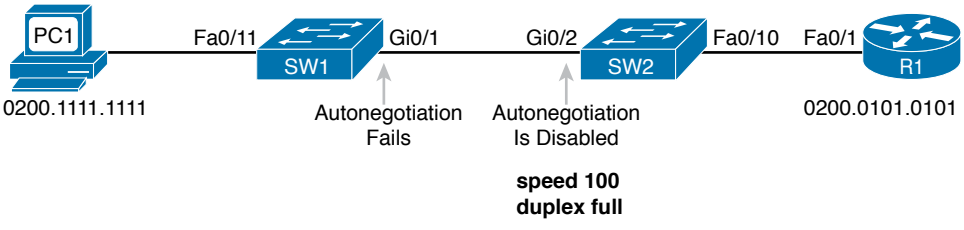


Figure 7-4 *Conditions to Create a Duplex Mismatch Between SW1 and SW2*

Example 7-8 *Confirming Duplex Mismatch on Switch SW1*

```
SW1# show interfaces gi0/1 status

Port      Name               Status       Vlan       Duplex  Speed Type
Gi0/1                        connected    trunk      a-half  a-100 10/100/1000BaseTX
```

First, note that even though SW1 had to use an autonegotiation default, the **show interfaces status** command still shows the speed and duplex with the **a-** prefix. SW2's port was manually set to 100/Full, so SW1 sensed the speed and runs at 100 Mbps; however, the autonegotiation rules then tell SW1 to use half duplex, as confirmed by the output in Example 7-8.

The output does not identify the duplex mismatch in any way; in fact, finding a duplex mismatch can be much more difficult than finding a speed mismatch. For instance, if you purposefully set the speed on the link in Figure 7-4 to be 10 Mbps on one switch and 100 Mbps on the other, both switches would list the port in a down/down or notconnect state. However, in the case shown in Example 7-8, with a duplex mismatch, *if the duplex settings do not match on the ends of an Ethernet segment, the switch interface will still be in a connected (up/up) or connected state.*

Not only does the **show** command give an appearance that the link has no issues, but the link will likely work poorly, with symptoms of intermittent problems. The reason is that the device using half duplex (SW1 in this case) uses carrier sense multiple access collision detect (CSMA/CD) logic, waiting to send when receiving a frame, believing collisions occur when they physically do not—and actually stopping sending a frame because the switch thinks a collision occurred. With enough traffic load, the interface could be in a connect state, but it's extremely inefficient for passing traffic.

To identify duplex mismatch problems, check the duplex setting on each end of the link to see if the values mismatch. You can also watch for incrementing collision and late collision counters, as explained in the next section.

Common Layer 1 Problems on Working Interfaces

When the interface reaches the connect (up/up) state, the switch considers the interface to be working. The switch, of course, tries to use the interface, and at the same time, the switch keeps various interface counters. These interface counters can help identify problems that can occur even though the interface is in a connect state, like issues related to the duplex mismatch problem that was just described. This section explains some of the related concepts and a few of the most common problems.

Whenever the physical transmission has problems, the receiving device might receive a frame whose bits have changed values. These frames do not pass the error detection logic as implemented in the FCS field in the Ethernet trailer, as covered in Chapter 2. The receiving device discards the frame and counts it as some kind of *input error*. Cisco switches list this error as a CRC error, as highlighted in Example 7-9. (Cyclic redundancy check [CRC] is a term related to how the frame check sequence [FCS] math detects an error.)

Example 7-9 *Interface Counters for Layer 1 Problems*

```
SW1# show interfaces fa0/13
! lines omitted for brevity
     Received 284 broadcasts (0 multicast)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 281 multicast, 0 pause input
     0 input packets with dribble condition detected
     95226 packets output, 10849674 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 PAUSE output
     0 output buffer failures, 0 output buffers swapped out
```

The number of input errors and the number of CRC errors are just a few of the counters in the output of the **show interfaces** command. The challenge is to decide which counters you need to think about, which ones show that a problem is happening, and which ones are normal and of no concern.

The example highlights several of the counters as examples so that you can start to understand which ones point to problems and which ones are just counting normal events that are not problems. The following list shows a short description of each highlighted counter, in the order shown in the example:

Runts: Frames that did not meet the minimum frame size requirement (64 bytes, including the 18-byte destination MAC, source MAC, type, and FCS). Can be caused by collisions.

Giants: Frames that exceed the maximum frame size requirement (1518 bytes, including the 18-byte destination MAC, source MAC, type, and FCS).

Input Errors: A total of many counters, including runts, giants, no buffer, CRC, frame, overrun, and ignored counts.

CRC: Received frames that did not pass the FCS math; can be caused by collisions.

Frame: Received frames that have an illegal format, for example, ending with a partial byte; can be caused by collisions.

Packets Output: Total number of packets (frames) forwarded out the interface.

Output Errors: Total number of packets (frames) that the switch port tried to transmit, but for which some problem occurred.

Collisions: Counter of all collisions that occur when the interface is transmitting a frame.

Late Collisions: The subset of all collisions that happen after the 64th byte of the frame has been transmitted. (In a properly working Ethernet LAN, collisions should occur within the first 64 bytes; late collisions today often point to a duplex mismatch.)

Note that many of these counters occur as part of the CSMA/CD process used when half duplex is enabled. Collisions occur as a normal part of the half-duplex logic imposed by CSMA/CD, so a switch interface with an increasing collisions counter might not even have a

problem. However, one problem, called late collisions, points to the classic duplex mismatch problem.

If a LAN design follows cabling guidelines, all collisions should occur by the end of the 64th byte of any frame. When a switch has already sent 64 bytes of a frame, and the switch receives a frame on that same interface, the switch senses a collision. In this case, the collision is a late collision, and the switch increments the late collision counter in addition to the usual CSMA/CD actions to send a jam signal, wait a random time, and try again.

With a duplex mismatch, like the mismatch between SW1 and SW2 in Figure 7-4, the half-duplex interface will likely see the late collisions counter increment. Why? The half-duplex interface sends a frame (SW1), but the full-duplex neighbor (SW2) sends at any time, even after the 64th byte of the frame sent by the half-duplex switch. So, just keep repeating the **show interfaces** command, and if you see the late collisions counter incrementing on a half-duplex interface, you might have a duplex mismatch problem.

A working interface (in an up/up state) can still suffer from issues related to the physical cabling as well. The cabling problems might not be bad enough to cause a complete failure, but the transmission failures result in some frames failing to pass successfully over the cable. For example, excessive interference on the cable can cause the various input error counters to keep growing larger, especially the CRC counter. In particular, if the CRC errors grow, but the collisions counters do not, the problem might simply be interference on the cable. (The switch counts each collided frame as one form of input error as well.)

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element section titled "Step 2: Build Your Study Habits Around the Chapter" for more details. Table 7-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 7-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Review memory tables		Website
Do labs		Sim Lite, blog

Review All the Key Topics

Key Topic

Table 7-4 Key Topics for Chapter 7

Key Topic Element	Description	Page Number
Example 7-1	Example of configuring **speed**, **duplex**, and **description**	153
Example 7-4	Example of disabling an interface using the **shutdown** command	155
List	Key decision rules for autonegotiation on Cisco switches when the other device does not participate	160
Table 7-2	Two types of interface state terms and their meanings	163
Example 7-7	Example that shows how to find the speed and duplex settings, as well as whether they were learned through autonegotiation	164
List	Defaults for IEEE autonegotiation	165
List	Explanations of different error statistics on switch interfaces	167

Key Terms You Should Know

port security, autonegotiation, full duplex, half duplex, 10/100, 10/100/1000

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included free with this book. The subnet of labs mostly relate to this part. Take the time to try some of the labs. As always, also check the author's blog site pages for configuration exercises (Config Labs) at https://blog.certskills.com.

Command References

Tables 7-5 and 7-6 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 7-5 Switch Interface Configuration

Command	Mode/Purpose/Description
interface *type port-number*	Changes context to interface mode. The type is typically Fast Ethernet or Gigabit Ethernet. The possible port numbers vary depending on the model of switch—for example, Fa0/1, Fa0/2, and so on.
interface range *type port-number - end-port-number*	Changes the context to interface mode for a range of consecutively numbered interfaces. The subcommands that follow then apply to all interfaces in the range.

Command	Mode/Purpose/Description
shutdown | no shutdown	Interface mode. Disables or enables the interface, respectively.
speed {**10 | 100 | 1000 | auto**}	Interface mode. Manually sets the speed to the listed speed or, with the auto setting, automatically negotiates the speed.
duplex {**auto | full | half**}	Interface mode. Manually sets the duplex to half or full, or to autonegotiate the duplex setting.
description *text*	Interface mode. Lists any information text that the engineer wants to track for the interface, such as the expected device on the other end of the cable.
no duplex **no speed** **no description**	Reverts to the default setting for each interface subcommand of **speed auto**, **duplex auto**, and the absence of a **description** command.

Table 7-6 Chapter 7 EXEC Command Reference

Command	Purpose
show running-config	Lists the currently used configuration
show running-config | interface *type number*	Displays the running-configuration excerpt of the listed interface and its subcommands only
show mac address-table dynamic [**interface** *type number*] [**vlan** *vlan-id*]	Lists the dynamically learned entries in the switch's address (forwarding) table, with subsets by interface and/or VLAN
show mac address-table static [**interface** *type number*]	Lists static MAC addresses and MAC addresses learned or defined with port security
show interfaces [*type number*] **status**	Lists one output line per interface (or for only the listed interface if included), noting the description, operating state, and settings for duplex and speed on each interface
show interfaces [*type number*]	Lists detailed status and statistical information about all interfaces (or the listed interface only)
show interfaces description	Displays one line of information per interface, with a two-item status (similar to the **show interfaces** command status), and includes any description that is configured on the interfaces

This page intentionally left blank

Part II Review

Keep track of your part review progress with the checklist shown in Table P2-1. Details on each task follow the table.

Table P2-1 Part II Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Review Appendix P on the Companion Website		
Videos		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book, using the PCPT software.

Answer Part Review Questions

For this task, answer the Part Review questions for this part of the book, using the PTP software.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson ICND1 or CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog: Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book, as found at http://blog.certskills.com. Then navigate to the Hands-on Config labs.

Other: If using other lab tools, as a few suggestions: Make sure to experiment heavily with VLAN configuration and VLAN trunking configuration. Also, spend some time changing interface settings like **speed** and **duplex** on a link between two switches, to make sure that you understand which cases would result in a duplex mismatch.

Review Appendix P on the Companion Website

The previous edition of the CCNA exam blueprint included the word "troubleshoot" as applied to Ethernet and VLANs, while the current CCNA exam blueprint does not. Appendix P on the companion website contains a chapter from the previous edition of the book that focused on troubleshooting. That appendix, named "LAN Troubleshooting," can be useful as a tool to review the topics in this part of the book. (Note that if you use this extra appendix, you can ignore the mentions of Port Security until you have reached that topic in the *CCNA 200-301 Official Cert Guide, Volume 2*.)

Watch Videos

Chapters 4 and 5 each recommend a video that can be helpful to anyone who is just learning about the Cisco CLI and basic switching concepts. If you have not watched those videos yet, take a moment to navigate to the companion website and watch the videos (listed under Chapters 4 and 5).

Part II of this book introduces the basics of Ethernet LANs, both in concept and in how to implement the features. However, the two primary features discussed in Part III of this book—Virtual LANs (VLANs) and Spanning Tree Protocol (STP)—impact almost everything you have learned about Ethernet so far. VLANs allow a network engineer to create separate Ethernet LANs through simple configuration choices. The ability to separate some switch ports into one VLAN and other switch ports into another VLAN gives network designers a powerful tool for creating networks. Once created, VLANs also have a huge impact on how a switch works, which then impacts how you verify and troubleshoot the operation of a campus LAN.

STP—and the related and similar Rapid STP (RSTP)—acts to prevent frames from looping around a LAN. Without STP or RSTP, in LANs with redundant links, broadcasts and some other frames would be forwarded around and around the LAN, eventually clogging the LAN so much as to make it unusable.

The current CCNA 200-301 exam blueprint includes exam topics for the configuration and verification of VLANs and related topics. However, the CCNA exam topics only mention RSTP concepts rather than configuration/verification. To that end, Part III opens with Chapter 8, which goes to the configuration/verification depth with VLAN topics, followed by Chapter 9, which introduces the concepts of STP and RSTP.

Part III closes with Chapter 10, which includes some RSTP configuration, along with Layer 2 EtherChannel configuration.

Other Resources

As one additional suggestion for those who intend to move on to CCNP Enterprise, consider skimming or reading Appendix P, "LAN Troubleshooting," found on the online companion website. This appendix, a copy of a chapter from the previous edition of the book, takes a troubleshooting approach to many of the topics found in Parts II and III of this book. Although Cisco completely removed the word *troubleshoot* from the CCNA exam blueprint in its current CCNA 200-301 version, the topics still remain relevant and can be a help for reviewing and refining what you learned in Parts II and III of this book.

Part III

Implementing VLANs and STP

Chapter 8: Implementing Ethernet Virtual LANs

Chapter 9: Spanning Tree Protocol Concepts

Chapter 10: RSTP and EtherChannel Configuration

Part III Review

CHAPTER 8

Implementing Ethernet Virtual LANs

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.13 Describe switching concepts

1.13.a MAC learning and aging

1.13.b Frame switching

1.13.c Frame flooding

1.13.d MAC address table

2.0 Network Access

2.1 Configure and verify VLANs (normal range) spanning multiple switches

2.1.a Access ports (data and voice)

2.1.b Default VLAN

2.1.c Connectivity

2.2 Configure and verify interswitch connectivity

2.2.a Trunk ports

2.2.b 802.1Q

2.2.c Native VLAN

So far in this book, you have learned that Ethernet switches receive Ethernet frames, make decisions, and then forward (switch) those Ethernet frames. That core logic revolves around MAC addresses, the interface in which the frame arrives, and the interfaces out which the switch forwards the frame.

While true, that logic omits any consideration of virtual LANs (VLANs). VLANs impact the switching logic for each frame because each VLAN acts as a subset of the switch ports in an Ethernet LAN. Switches believe each Ethernet frame to be received in an identifiable VLAN, forwarded based on MAC table entries for that VLAN, and forwarded out ports in that VLAN. This chapter explores those concepts and others related to VLANs.

As for the organization of the chapter, the first major section of the chapter explains the core concepts. These concepts include how VLANs work on a single switch, how to use VLAN trunking to create VLANs that span across multiple switches, and how to forward traffic between VLANs using a router. The second major section shows how to configure VLANs and VLAN trunks: how to statically assign interfaces to a VLAN. The final major section discusses some issues that can arise when using VLANs and trunks and how to avoid those issues.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 8-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Virtual LAN Concepts	1–3
VLAN and VLAN Trunking Configuration and Verification	4–6
Troubleshooting VLANs and VLAN Trunks	7–8

1. In a LAN, which of the following terms best equates to the term VLAN?
   - **a.** Collision domain
   - **b.** Broadcast domain
   - **c.** Subnet
   - **d.** Single switch
   - **e.** Trunk
2. Imagine a switch with three configured VLANs. How many IP subnets are required, assuming that all hosts in all VLANs want to use TCP/IP?
   - **a.** 0
   - **b.** 1
   - **c.** 2
   - **d.** 3
   - **e.** You cannot tell from the information provided.
3. Switch SW1 sends a frame to switch SW2 using 802.1Q trunking. Which of the answers describes how SW1 changes or adds to the Ethernet frame before forwarding the frame to SW2?
   - **a.** Inserts a 4-byte header and does change the MAC addresses
   - **b.** Inserts a 4-byte header and does not change the MAC addresses
   - **c.** Encapsulates the original frame behind an entirely new Ethernet header
   - **d.** None of the other answers are correct

4. Imagine that you are told that switch 1 is configured with the **dynamic auto** parameter for trunking on its Fa0/5 interface, which is connected to switch 2. You have to configure switch 2. Which of the following settings for trunking could allow trunking to work? (Choose two answers.)
   a. **trunk**
   b. **dynamic auto**
   c. **dynamic desirable**
   d. **access**
   e. None of the other answers are correct.

5. A switch has just arrived from Cisco. The switch has never been configured with any VLANs, but VTP has been disabled. An engineer configures the **vlan 22** and **name Hannahs-VLAN** commands and then exits configuration mode. Which of the following are true? (Choose two answers.)
   a. VLAN 22 is listed in the output of the **show vlan brief** command.
   b. VLAN 22 is listed in the output of the **show running-config** command.
   c. VLAN 22 is not created by this process.
   d. VLAN 22 does not exist in that switch until at least one interface is assigned to that VLAN.

6. Which of the following commands identify switch interfaces as being trunking interfaces: interfaces that currently operate as VLAN trunks? (Choose two answers.)
   a. **show interfaces**
   b. **show interfaces switchport**
   c. **show interfaces trunk**
   d. **show trunks**

7. In a switch that disables VTP, an engineer configures the commands **vlan 30** and **shutdown vlan 30**. Which answers should be true about this switch? (Choose two answers.)
   a. The **show vlan brief** command should list VLAN 30.
   b. The **show running-config** command should list VLAN 30.
   c. The switch should forward frames that arrive in access ports in VLAN 30.
   d. The switch should forward frames that arrive in trunk ports tagged with VLAN 30.

8. The **show interfaces g0/1 trunk** command provides three lists of VLAN IDs. Which items would limit the VLANs that appear in the first of the three lists of VLANs?
   a. A **shutdown vlan 30** global command
   b. A **switchport trunk allowed vlan** interface subcommand
   c. An STP choice to block on G0/1
   d. A **no vlan 30** global command

Foundation Topics

Virtual LAN Concepts

Before understanding VLANs, you must first have a specific understanding of the definition of a LAN. For example, from one perspective, a LAN includes all the user devices, servers, switches, routers, cables, and wireless access points in one location. However, an alternative narrower definition of a LAN can help in understanding the concept of a virtual LAN:

A LAN includes all devices in the same broadcast domain.

A broadcast domain includes the set of all LAN-connected devices, so that when any of the devices sends a broadcast frame, all the other devices get a copy of the frame. So, from one perspective, you can think of a LAN and a broadcast domain as being basically the same thing.

Using only default settings, a switch considers all its interfaces to be in the same broadcast domain. That is, for one switch, when a broadcast frame entered one switch port, the switch forwards that broadcast frame out all other ports. With that logic, to create two different LAN broadcast domains, you had to buy two different Ethernet LAN switches, as shown in Figure 8-1.

Figure 8-1 *Creating Two Broadcast Domains with Two Physical Switches and No VLANs*

By using two VLANs, a single switch can accomplish the same goals of the design in Figure 8-1—to create two broadcast domains—with a single switch. With VLANs, a switch can configure some interfaces into one broadcast domain and some into another, creating multiple broadcast domains. These individual broadcast domains created by the switch are called *virtual LANs* (VLAN).

For example, in Figure 8-2, the single switch creates two VLANs, treating the ports in each VLAN as being completely separate. The switch would never forward a frame sent by Dino (in VLAN 1) over to either Wilma or Betty (in VLAN 2).

Key Topic

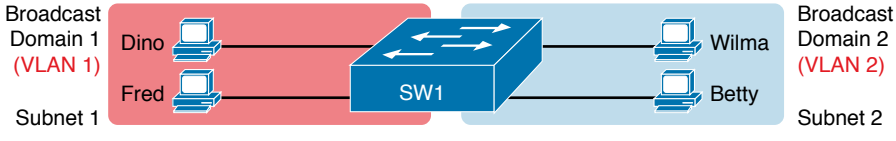


Figure 8-2 *Creating Two Broadcast Domains Using One Switch and VLANs*

Designing campus LANs to use more VLANs, each with a smaller number of devices, often helps improve the LAN in many ways. For example, a broadcast sent by one host in a VLAN will be received and processed by all the other hosts in the VLAN—but not by hosts in a different VLAN. Limiting the number of hosts that receive a single broadcast frame reduces the number of hosts that waste effort processing unneeded broadcasts. It also reduces

security risks because fewer hosts see frames sent by any one host. These are just a few reasons for separating hosts into different VLANs. The following list summarizes the most common reasons for choosing to create smaller broadcast domains (VLANs):

Key Topic

- To reduce CPU overhead on each device, improving host performance, by reducing the number of devices that receive each broadcast frame
- To reduce security risks by reducing the number of hosts that receive copies of frames that the switches flood (broadcasts, multicasts, and unknown unicasts)
- To improve security for hosts through the application of different security policies per VLAN
- To create more flexible designs that group users by department, or by groups that work together, instead of by physical location
- To solve problems more quickly, because the failure domain for many problems is the same set of devices as those in the same broadcast domain
- To reduce the workload for the Spanning Tree Protocol (STP) by limiting a VLAN to a single access switch

The rest of this chapter looks closely at the mechanics of how VLANs work across multiple Cisco switches, including the required configuration. To that end, the next section examines VLAN trunking, a feature required when installing a VLAN that exists on more than one LAN switch.

Creating Multiswitch VLANs Using Trunking

Configuring VLANs on a single switch requires only a little effort: you simply configure each port to tell it the VLAN number to which the port belongs. With multiple switches, you have to consider additional concepts about how to forward traffic between the switches.

When you are using VLANs in networks that have multiple interconnected switches, the switches need to use *VLAN trunking* on the links between the switches. VLAN trunking causes the switches to use a process called *VLAN tagging*, by which the sending switch adds another header to the frame before sending it over the trunk. This extra trunking header includes a *VLAN identifier* (VLAN ID) field so that the sending switch can associate the frame with a particular VLAN ID, and the receiving switch can then know in what VLAN each frame belongs.

Figure 8-3 shows an example that demonstrates VLANs that exist on multiple switches, but it does not use trunking. First, the design uses two VLANs: VLAN 10 and VLAN 20. Each switch has two ports assigned to each VLAN, so each VLAN exists in both switches. To forward traffic in VLAN 10 between the two switches, the design includes a link between switches, with that link fully inside VLAN 10. Likewise, to support VLAN 20 traffic between switches, the design uses a second link between switches, with that link inside VLAN 20.

The design in Figure 8-3 functions perfectly. For example, PC11 (in VLAN 10) can send a frame to PC14. The frame flows into SW1, over the top link (the one that is in VLAN 10) and over to SW2.

Answers to the "Do I Know This Already?" quiz:

1 B **2** D **3** B **4** A, C **5** A, B **6** B, C **7** A, B **8** B

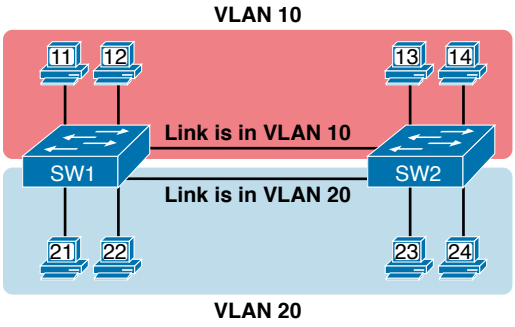


Figure 8-3 *Multiswitch VLAN Without VLAN Trunking*

The design shown in Figure 8-3 works, but it simply does not scale very well. It requires one physical link between switches to support every VLAN. If a design needed 10 or 20 VLANs, you would need 10 or 20 links between switches, and you would use 10 or 20 switch ports (on each switch) for those links.

VLAN Tagging Concepts

VLAN trunking creates one link between switches that supports as many VLANs as you need. As a VLAN trunk, the switches treat the link as if it were a part of all the VLANs. At the same time, the trunk keeps the VLAN traffic separate, so frames in VLAN 10 would not go to devices in VLAN 20, and vice versa, because each frame is identified by VLAN number as it crosses the trunk. Figure 8-4 shows the idea, with a single physical link between the two switches.

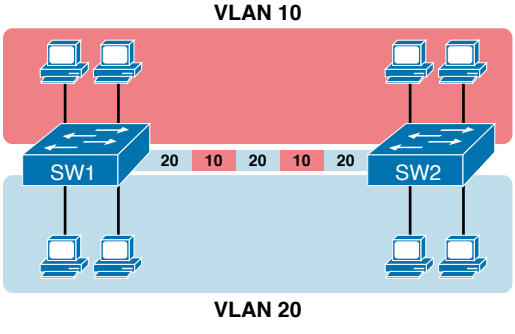


Figure 8-4 *Multiswitch VLAN with Trunking*

The use of trunking allows switches to forward frames from multiple VLANs over a single physical connection by adding a small header to the Ethernet frame. For example, Figure 8-5 shows PC11 sending a broadcast frame on interface Fa0/1 at Step 1. To flood the frame, switch SW1 needs to forward the broadcast frame to switch SW2. However, SW1 needs to let SW2 know that the frame is part of VLAN 10, so that after the frame is received, SW2 will flood the frame only into VLAN 10, and not into VLAN 20. So, as shown at Step 2, before sending the frame, SW1 adds a VLAN header to the original Ethernet frame, with the VLAN header listing a VLAN ID of 10 in this case.

Key Topic

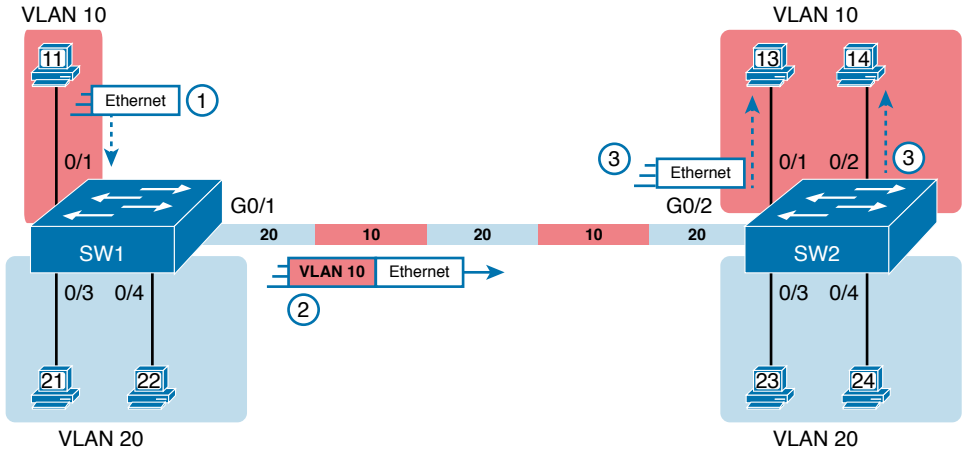


Figure 8-5 *VLAN Trunking Between Two Switches*

When SW2 receives the frame, it understands that the frame is in VLAN 10. SW2 then removes the VLAN header, forwarding the original frame out its interfaces in VLAN 10 (Step 3).

For another example, consider the case when PC21 (in VLAN 20) sends a broadcast. SW1 sends the broadcast out port Fa0/4 (because that port is in VLAN 20) and out Gi0/1 (because it is a trunk, meaning that it supports multiple different VLANs). SW1 adds a trunking header to the frame, listing a VLAN ID of 20. SW2 strips off the trunking header after determining that the frame is part of VLAN 20, so SW2 knows to forward the frame out only ports Fa0/3 and Fa0/4, because they are in VLAN 20, and not out ports Fa0/1 and Fa0/2, because they are in VLAN 10.

The 802.1Q and ISL VLAN Trunking Protocols

Cisco has supported two different trunking protocols over the years: Inter-Switch Link (ISL) and IEEE 802.1Q. Cisco created the ISL years before 802.1Q, in part because the IEEE had not yet defined a VLAN trunking standard. Today, 802.1Q has become the more popular trunking protocol, with Cisco not even bothering to support ISL in many of its switch models today.

While both ISL and 802.1Q tag each frame with the VLAN ID, the details differ. 802.1Q inserts an extra 4-byte 802.1Q VLAN header into the original frame's Ethernet header, as shown at the top of Figure 8-6. As for the fields in the 802.1Q header, only the 12-bit VLAN ID field inside the 802.1Q header matters for topics discussed in this book. This 12-bit field supports a theoretical maximum of 2^{12} (4096) VLANs, but in practice it supports a maximum of 4094. (Both 802.1Q and ISL use 12 bits to tag the VLAN ID, with two reserved values [0 and 4095].)

Cisco switches break the range of VLAN IDs (1–4094) into two ranges: the normal range and the extended range. All switches can use normal-range VLANs with values from 1 to 1005. Only some switches can use extended-range VLANs with VLAN IDs from 1006 to 4094. The rules for which switches can use extended-range VLANs depend on the configuration of the VLAN Trunking Protocol (VTP), which is discussed briefly in the section "VLAN Trunking Configuration," later in this chapter.

Key Topic

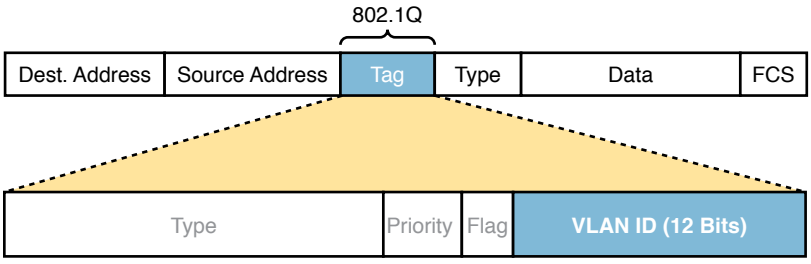


Figure 8-6 *802.1Q Trunking*

802.1Q also defines one special VLAN ID on each trunk as the *native VLAN* (defaulting to use VLAN 1). By definition, 802.1Q simply does not add an 802.1Q header to frames in the native VLAN. When the switch on the other side of the trunk receives a frame that does not have an 802.1Q header, the receiving switch knows that the frame is part of the native VLAN. Note that because of this behavior, both switches must agree on which VLAN is the native VLAN.

The 802.1Q native VLAN provides some interesting functions, mainly to support connections to devices that do not understand trunking. For example, a Cisco switch could be cabled to a switch that does not understand 802.1Q trunking. The Cisco switch could send frames in the native VLAN—meaning that the frame has no trunking header—so that the other switch would understand the frame. The native VLAN concept gives switches the capability of at least passing traffic in one VLAN (the native VLAN), which can allow some basic functions, like reachability to telnet into a switch.

Forwarding Data Between VLANs

If you create a campus LAN that contains many VLANs, you typically still need all devices to be able to send data to all other devices. This next topic discusses some concepts about how to route data between those VLANs.

8

The Need for Routing Between VLANs

LAN switches that forward data based on Layer 2 logic, as discussed so far in this book, often go by the name *Layer 2 switch*. For example, Chapter 5, "Analyzing Ethernet LAN Switching," discussed how LAN switches receive Ethernet frames (a Layer 2 concept), look at the destination Ethernet MAC address (a Layer 2 address), and forward the Ethernet frame out some other interface. All those concepts are defined by Layer 2 protocols, hence the name Layer 2 switch.

Layer 2 switches perform their logic per VLAN. For example, in Figure 8-7, the two PCs on the left sit in VLAN 10, in subnet 10. The two PCs on the right sit in a different VLAN (20), with a different subnet (20). Note that the figure repeats earlier Figure 8-2, but with the switch broken into halves, to emphasize the point that Layer 2 switches will not forward data between two VLANs.

As shown in the figure, when configured with some ports in VLAN 10 and others in VLAN 20, the switch acts like two separate switches in which it will forward traffic. In fact, one goal of VLANs is to separate traffic in one VLAN from another, preventing frames in one VLAN from leaking over to other VLANs. For example, when Dino (in VLAN 10) sends any Ethernet frame, if SW1 is a Layer 2 switch, that switch will not forward the frame to the PCs on the right in VLAN 20.

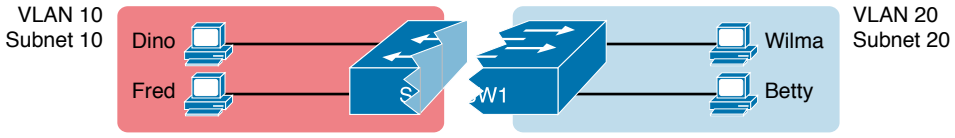


Figure 8-7 *Layer 2 Switch Does Not Route Between the VLANs*

Routing Packets Between VLANs with a Router

When including VLANs in a campus LAN design, the devices in a VLAN need to be in the same subnet. Following the same design logic, devices in different VLANs need to be in different subnets.

To forward packets between VLANs, the network must use a device that acts as a router. You can use an actual router, as well as some other switches that can perform some functions like a router. These switches that also perform Layer 3 routing functions go by the name *multilayer switch* or *Layer 3 switch*. This section first discusses how to forward data between VLANs when using Layer 2 switches and ends with a brief discussion of how to use Layer 3 switches.

For example, Figure 8-8 shows a router that can route packets between subnets 10 and 20. The figure shows the same Layer 2 switch as shown in Figure 8-7, with the same perspective of the switch being split into parts with two different VLANs, and with the same PCs in the same VLANs and subnets. Now Router R1 has one LAN physical interface connected to the switch and assigned to VLAN 10, and a second physical interface connected to the switch and assigned to VLAN 20. With an interface connected to each subnet, the Layer 2 switch can keep doing its job—forwarding frames inside a VLAN, while the router can do its job—routing IP packets between the subnets.

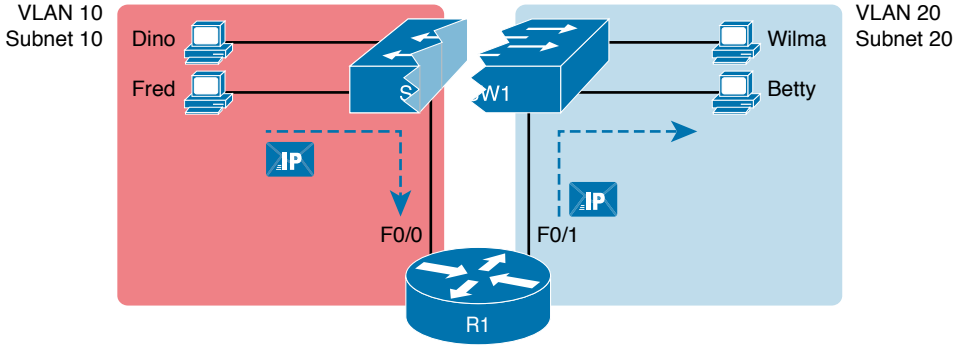


Figure 8-8 *Routing Between Two VLANs on Two Physical Interfaces*

The figure shows an IP packet being routed from Fred, which sits in one VLAN/subnet, to Betty, which sits in the other. The Layer 2 switch forwards two different Layer 2 Ethernet frames: one in VLAN 10, from Fred to R1's F0/0 interface, and the other in VLAN 20, from R1's F0/1 interface to Betty. From a Layer 3 perspective, Fred sends the IP packet to its default router (R1), and R1 routes the packet out another interface (F0/1) into another subnet where Betty resides.

The design in Figure 8-8 works, but there are several different solutions for routing packets between VLANs. This chapter shows the option of using a separate physical router, with a

separate link per VLAN, because it can be the easiest of the options to understand and visualize. Chapter 17, "IP Routing in the LAN," works through those other features for routing packets between VLANs.

VLAN and VLAN Trunking Configuration and Verification

Cisco switches do not require any configuration to work. You can purchase Cisco switches, install devices with the correct cabling, turn on the switches, and they work. You would never need to configure the switch, and it would work fine, even if you interconnected switches, until you needed more than one VLAN. But if you want to use VLANs—and most enterprise networks do—you need to add some configuration.

This chapter separates the VLAN configuration details into two major sections. The first section looks at how to configure static access interfaces: switch interfaces configured to be in one VLAN only, therefore not using VLAN trunking. The second part shows how to configure interfaces that do use VLAN trunking.

Creating VLANs and Assigning Access VLANs to an Interface

This section shows how to create a VLAN, give the VLAN a name, and assign interfaces to a VLAN. To focus on these basic details, this section shows examples using a single switch, so VLAN trunking is not needed.

For a Cisco switch to forward frames in a particular VLAN, the switch must be configured to believe that the VLAN exists. In addition, the switch must have nontrunking interfaces (called *access interfaces*, or *static access interfaces*) assigned to the VLAN, and/or trunks that support the VLAN. The configuration steps for access interfaces are as follows:

Config Checklist

Step 1. To configure a new VLAN, follow these steps:

A. From configuration mode, use the **vlan** *vlan-id* command in global configuration mode to create the VLAN and to move the user into VLAN configuration mode.

B. (Optional) Use the **name** *name* command in VLAN configuration mode to list a name for the VLAN. If not configured, the VLAN name is VLAN*ZZZZ*, where *ZZZZ* is the four-digit decimal VLAN ID.

Step 2. For each access interface, follow these steps:

A. Use the **interface** *type number* command in global configuration mode to move into interface configuration mode for each desired interface.

B. Use the **switchport access vlan** *id-number* command in interface configuration mode to specify the VLAN number associated with that interface.

C. (Optional) Use the **switchport mode access** command in interface configuration mode to make this port always operate in access mode (that is, to not trunk).

While the list might look a little daunting, the process on a single switch is actually pretty simple. For example, if you want to put the switch's ports in three VLANs—11, 12, and

13—you first add three **vlan** commands: **vlan 11**, **vlan 12**, and **vlan 13**. Then, for each interface, add a **switchport access vlan 11** (or **12** or **13**) command to assign that interface to the proper VLAN.

NOTE The term *default VLAN* (as shown in the exam topics) refers to the default setting on the **switchport access vlan** *vlan-id* command, and that default is VLAN ID 1. In other words, by default, each port is assigned to access VLAN 1.

VLAN Configuration Example 1: Full VLAN Configuration

Examples 8-1, 8-2, and 8-3 work through one scenario with VLAN configuration and verification. To begin, Example 8-1 begins by showing the VLANs in switch SW1 in Figure 8-9, with all default settings related to VLANs.

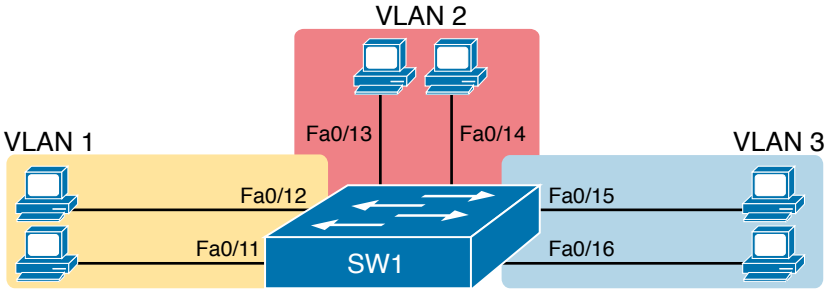


Figure 8-9 *Network with One Switch and Three VLANs*

Example 8-1 *Configuring VLANs and Assigning VLANs to Interfaces*

```
SW1# show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                Gi0/1, Gi0/2
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

The example begins with the **show vlan brief** command, confirming the default settings of five nondeletable VLANs, with all interfaces assigned to VLAN 1. VLAN 1 cannot be deleted but can be used. VLANs 1002–1005 cannot be deleted and cannot be used as access VLANs today. In particular, note that this 2960 switch has 24 Fast Ethernet ports (Fa0/1–Fa0/24) and two Gigabit Ethernet ports (Gi0/1 and Gi0/2), all of which are listed as being in

VLAN 1 per that first command's output, confirming that by default, Cisco switches assign all ports to VLAN 1.

Next, Example 8-2 shows steps that mirror the VLAN configuration checklist, namely the configuration of VLAN 2, plus the assignment of VLAN 2 as the access VLAN on two ports: Fa0/13 and Fa0/14.

Example 8-2 *Configuring VLANs and Assigning VLANs to Interfaces*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# vlan 2
SW1(config-vlan)# name Freds-vlan
SW1(config-vlan)# exit
SW1(config)# interface range fastethernet 0/13 - 14
SW1(config-if)# switchport access vlan 2
SW1(config-if)# switchport mode access
SW1(config-if)# end

SW1# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                Fa0/15, Fa0/16, Fa0/17, Fa0/18
                                                Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                                Fa0/23, Fa0/24, Gi0/1, Gi0/2
2    Freds-vlan                       active    Fa0/13, Fa0/14
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Take a moment to compare the output of the **show vlan brief** commands in Example 8-2 (after adding the configuration) versus Example 8-1. Example 8-2 shows new information about VLAN 2, with ports Fa0/13 and Fa0/14 no longer being listed with VLAN 1, but now listed as assigned to VLAN 2.

To complete this scenario, Example 8-3 shows a little more detail about the VLAN itself. First, the **show running-config** command lists both the **vlan 2** and **switchport access vlan 2** commands as configured in Example 8-2. Also, note that earlier Example 8-2 uses the **interface range** command, with one instance of the **switchport access vlan 2** interface subcommand. However, Example 8-3 shows how the switch actually applied that command to both Fa0/13 and Fa0/14. Example 8-3 ends with the **show vlan id 2** command, which confirms the operational status that ports Fa0/13 and Fa0/14 are assigned to VLAN 2.

Example 8-3 *Configuring VLANs and Assigning VLANs to Interfaces*

```
SW1# show running-config
! Many lines omitted for brevity
! Early in the output:
vlan 2
 name Freds-vlan
!
! more lines omitted for brevity
interface FastEthernet0/13
 switchport access vlan 2
 switchport mode access
!
interface FastEthernet0/14
 switchport access vlan 2
 switchport mode access
!

SW1# show vlan id 2
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
2    Freds-vlan                       active    Fa0/13, Fa0/14

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
2    enet  100010     1500  -      -      -        -    -        0      0

Remote SPAN VLAN
----------------
Disabled

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------
```

The example surrounding Figure 8-9 uses six switch ports, all of which need to operate as access ports. That is, each port should not use trunking but instead should be assigned to a single VLAN, as assigned by the **switchport access vlan** *vlan-id* command. For ports that should always act as access ports, add the optional interface subcommand **switchport mode access**. This command tells the switch to always be an access interface and disables the protocol that negotiates trunking (Dynamic Trunking Protocol [DTP]) with the device on the other end of the link. (The upcoming section "VLAN Trunking Configuration" discusses more details about the commands that allow a port to negotiate whether it should use trunking.)

NOTE The book includes a video that works through a different VLAN configuration example as well. You can find the video on the companion website.

VLAN Configuration Example 2: Shorter VLAN Configuration

Example 8-2 shows how to configure a VLAN and add two ports to the VLAN as access ports. Example 8-4 does the same, this time with VLAN 3, and this time with a much briefer alternative configuration. The configuration completes the configuration of the design shown in Figure 8-9, by adding two ports to VLAN 3.

Example 8-4 *Shorter VLAN Configuration Example (VLAN 3)*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# interface range Fastethernet 0/15 - 16
SW1(config-if-range)# switchport access vlan 3
% Access VLAN does not exist. Creating vlan 3
SW1(config-if-range)# ^Z

SW1# show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
1 default                              active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                 Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                 Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                 Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                 Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                 Gi0/1, Gi0/2
2 Freds-vlan                           active    Fa0/13, Fa0/14
3 VLAN0003                             active    Fa0/15, Fa0/16
1002 fddi-default                      act/unsup
1003 token-ring-default                act/unsup
1004 fddinet-default                   act/unsup
1005 trnet-default                     act/unsup
```

Example 8-4 shows how a switch can dynamically create a VLAN—the equivalent of the **vlan** *vlan-id* global config command—when the **switchport access vlan** interface subcommand refers to a currently unconfigured VLAN. This example begins with SW1 not knowing about VLAN 3. With the addition of the **switchport access vlan 3** interface subcommand, the switch realized that VLAN 3 did not exist, and as noted in the shaded message in the example, the switch created VLAN 3, using a default name (VLAN0003). The engineer did not need to type the **vlan 3** global command to create VLAN 3; the switch did that automatically. No other steps are required to create the VLAN. At the end of the process, VLAN 3 exists in the switch, and interfaces Fa0/15 and Fa0/16 are in VLAN 3, as noted in the shaded part of the **show vlan brief** command output.

VLAN Trunking Protocol

Before showing more configuration examples, you also need to know something about a Cisco protocol and tool called the VLAN Trunking Protocol (VTP). VTP is a Cisco proprietary

tool on Cisco switches that advertises each VLAN configured in one switch (with the **vlan** *number* command) so that all the other switches in the campus learn about that VLAN.

This book does not discuss VTP as an end to itself for a few different reasons. First, the current CCNA 200-301 exam blueprint ignores VTP, as do the CCNP Enterprise Core and CCNP Enterprise Advanced Routing blueprints. Additionally, many enterprises choose to disable VTP.

Also, you can easily disable VTP so that it has no impact on your switches in the lab, which is exactly what I did when building all the examples in this book.

However, VTP has some small impact on how every Cisco Catalyst switch works, even if you do not try to use VTP. This brief section introduces enough details of VTP so that you can see these small differences in VTP that cannot be avoided.

First, all examples in this book (and in Volume 2) use switches that disable VTP in some way. Interestingly, for much of VTP's decades of existence, most switches did not allow VTP to be disabled completely; on those switches, to effectively disable VTP, the engineer would set the switch to use VTP transparent mode (with the **vtp mode transparent** global command). Some switches now have an option to disable VTP completely with the **vtp mode off** global command. For the purposes of this book, configuring a switch with either transparent mode or off mode disables VTP.

Note that both transparent and off modes prevent VTP from learning and advertising about VLAN configuration. Those modes allow a switch to configure all VLANs, including standard- and extended-range VLANs. Additionally, switches using transparent or off modes list the **vlan** configuration commands in the running-config file.

Finally, on a practical note, if you happen to do lab exercises with real switches or with simulators, and you see unusual results with VLANs, check the VTP status with the **show vtp status** command. If your switch uses VTP server or client mode, you will find

- The server switches can configure VLANs in the standard range only (1–1005).
- The client switches cannot configure VLANs.
- Both servers and clients may be learning new VLANs from other switches and seeing their VLANs deleted by other switches because of VTP.
- The **show running-config** command does not list any **vlan** commands; you must use other **show** commands to find out about the configured VLANs.

If possible in the lab, switch to disable VTP and ignore VTP for your switch configuration practice until you decide to learn more about VTP for other purposes.

NOTE Do not change VTP settings on any switch that also connects to the production network until you know how VTP works and you talk with experienced colleagues. Doing so can cause real harm to your LAN. For example, if the switch you configure connects to other switches, which in turn connect to switches used in the production LAN, you could accidentally change the VLAN configuration in other switches with serious impact to the operation of the network. You could delete VLANs and cause outages. Be careful and never experiment with VTP settings on a switch unless it and the other switches connected to it have absolutely no physical links connected to the production LAN.

VLAN Trunking Configuration

Trunking configuration between two Cisco switches can be very simple if you just statically configure trunking. For example, most Cisco Catalyst switches today support only 802.1Q and not ISL. You could literally add one interface subcommand for the switch interface on each side of the link (**switchport mode trunk**), and you would create a VLAN trunk that supported all the VLANs known to each switch.

However, trunking configuration on Cisco switches includes many more options, including several options for dynamically negotiating various trunking settings. The configuration can either predefine different settings or tell the switch to negotiate the settings, as follows:

- **The type of trunking:** IEEE 802.1Q, ISL, or negotiate which one to use, on switches that support both types of trunking.
- **The administrative mode:** Whether to always trunk, always not trunk, or negotiate whether to trunk or not.

First, consider the type of trunking. Cisco switches that support ISL and 802.1Q can negotiate which type to use, using the Dynamic Trunking Protocol (DTP). If both switches support both protocols, they use ISL; otherwise, they use the protocol that both support. Today, many Cisco switches do not support the older ISL trunking protocol. Switches that support both types of trunking use the **switchport trunk encapsulation** {**dot1q | isl | negotiate**} interface subcommand to either configure the type or allow DTP to negotiate the type.

DTP can also negotiate whether the two devices on the link agree to trunk at all, as guided by the local switch port's administrative mode. The administrative mode refers to the configuration setting for whether trunking should be used. Each interface also has an *operational* mode, which refers to what is currently happening on the interface and might have been chosen by DTP's negotiation with the other device. Cisco switches use the **switchport mode** interface subcommand to define the administrative trunking mode, as listed in Table 8-2.

Key Topic

Table 8-2 Trunking Administrative Mode Options with the **switchport mode** Command

Command Option	Description
access	Always act as an access (nontrunk) port
trunk	Always act as a trunk port
dynamic desirable	Initiates negotiation messages and responds to negotiation messages to dynamically choose whether to start using trunking
dynamic auto	Passively waits to receive trunk negotiation messages, at which point the switch will respond and negotiate whether to use trunking

For example, consider the two switches shown in Figure 8-10. This figure expands the design shown earlier in Figure 8-9, with a trunk to a new switch (SW2) and with parts of VLANs 1 and 3 on ports attached to SW2. The two switches use a Gigabit Ethernet link for the trunk. In this case, the trunk does not dynamically form by default because both (2960) switches default to an administrative mode of *dynamic auto*, meaning that neither switch initiates the trunk negotiation process. When one switch is changed to use *dynamic desirable* mode, which does initiate the negotiation, the switches negotiate to use trunking, specifically 802.1Q because the 2960s support only 802.1Q.

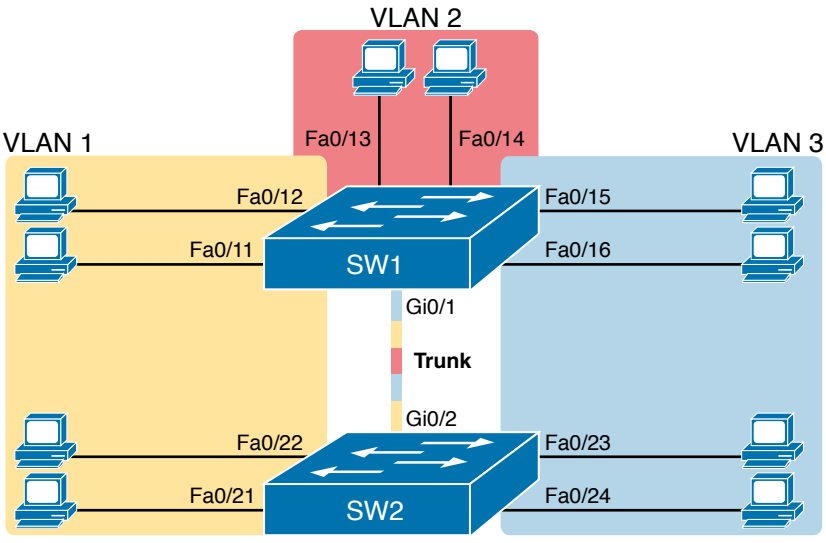


Figure 8-10 *Network with Two Switches and Three VLANs*

Example 8-5 begins with SW1 configured as shown in Examples 8-2 and 8-4—that is, SW1 has two ports each assigned to VLANs 1, 2, and 3. However, both SW1 and SW2 currently have all default settings on the interfaces that connect the two switches. With the default setting of **switchport mode dynamic auto**, the two switches do not trunk.

Example 8-5 *Initial (Default) State: Not Trunking Between SW1 and SW2*

```
SW1# show interfaces gigabit 0/1 switchport
Name: Gi0/1
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
```

```
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Protected: false
Unknown unicast blocked: disabled
Unknown multicast blocked: disabled
Appliance trust: none

! Note that the next command results in a single empty line of output.
SW1# show interfaces trunk
SW1#
```

First, focus on the highlighted items from the output of the **show interfaces switchport** command at the beginning of Example 8-3. The output lists the default administrative mode setting of dynamic auto. Because SW2 also defaults to dynamic auto, the command lists SW1's operational status as "access," meaning that it is not trunking. ("Dynamic auto" tells both switches to sit there and wait on the other switch to start the negotiations.) The third shaded line points out the only supported type of trunking (802.1Q). (On a switch that supports both ISL and 802.1Q, this value would by default list "negotiate," to mean that the type of encapsulation is negotiated.) Finally, the operational trunking type is listed as "native," which is a reference to the 802.1Q native VLAN.

The end of the example shows the output of the **show interfaces trunk** command, but with no output. This command lists information about all interfaces that currently operationally trunk; that is, it lists interfaces that currently use VLAN trunking. With no interfaces listed, this command also confirms that the link between switches is not trunking.

Next, consider Example 8-6, which shows the new configuration that enables trunking. In this case, SW1 is configured with the **switchport mode dynamic desirable** command, which asks the switch to both negotiate as well as to begin the negotiation process, rather than waiting on the other device. The example shows that as soon as the command is issued, log messages appear showing that the interface goes down and then back up again, which happens when the interface transitions from access mode to trunk mode.

Example 8-6 *SW1 Changes from Dynamic Auto to Dynamic Desirable*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# interface gigabit 0/1
SW1(config-if)# switchport mode dynamic desirable
SW1(config-if)# ^Z
SW1#
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to
  down
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to
  up
SW1# show interfaces gigabit 0/1 switchport
Name: Gi0/1
Switchport: Enabled
Administrative Mode: dynamic desirable
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
! lines omitted for brevity
```

Example 8-6 repeats the **show interfaces gi0/1 switchport** command seen in Example 8-5, but after configuring VLAN trunking, so this time the output shows that SW1's G0/1 interface now operates as a trunk. Note that the command still lists the administrative settings, which denote the configured values along with the operational settings, which list what the switch is currently doing. SW1 now claims to be in an operational mode of *trunk*, with an operational trunking encapsulation of dot1Q.

Example 8-7 now repeats the same **show interfaces trunk** command that showed no output at all back in Example 8-5. Now that SW1 trunks on its G0/1 port, the output in Example 8-7 lists G0/1, confirming that G0/1 is now operationally trunking. The next section discusses the meaning of the output of this command. Also, note that the end of the example repeats the **show vlan id 2** command; of note, it includes the trunk port G0/1 in the output because the trunk port can forward traffic in VLAN 2.

Example 8-7 *A Closer Look at SW1's G0/1 Trunk Port*

```
SW1# show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Gi0/1       desirable        802.1q         trunking      1

Port        Vlans allowed on trunk
Gi0/1       1-4094

Port        Vlans allowed and active in management domain
Gi0/1       1-3

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/1       1-3
```

```
SW1# show vlan id 2
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
2    Freds-vlan                       active    Fa0/13, Fa0/14, G0/1

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
2    enet  100010     1500  -      -      -        -    -        0      0

Remote SPAN VLAN
----------------
Disabled

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------
```

For the exams, you should be ready to interpret the output of the **show interfaces switchport** command, realize the administrative mode implied by the output, and know whether the link should operationally trunk based on those settings. Table 8-3 lists the combinations of the trunking administrative modes and the expected operational mode (trunk or access) resulting from the configured settings. The table lists the administrative mode used on one end of the link on the left, and the administrative mode on the switch on the other end of the link across the top of the table.

Key Topic

Table 8-3 Expected Trunking Operational Mode Based on the Configured Administrative Modes

Administrative Mode	Access	Dynamic Auto	Trunk	Dynamic Desirable
access	Access	Access	Do Not Use[1]	Access
dynamic auto	Access	Access	Trunk	Trunk
trunk	Do Not Use[1]	Trunk	Trunk	Trunk
dynamic desirable	Access	Trunk	Trunk	Trunk

[1] When two switches configure a mode of "access" on one end and "trunk" on the other, problems occur. Avoid this combination.

Finally, before leaving the discussion of configuring trunks, Cisco recommends disabling trunk negotiation on most ports for better security. The majority of switch ports on most switches will be used to connect to users and configured with the command **switchport mode access**—which also disables DTP. For ports without the **switchport mode access** command—for instance, ports statically configured to trunk with the **switchport mode trunk** command—DTP still operates, but you can disable DTP negotiations altogether using the **switchport nonegotiate** interface subcommand.

Implementing Interfaces Connected to Phones

This next topic is strange, at least in the context of access links and trunk links. In the world of IP telephony, telephones use Ethernet ports to connect to an Ethernet network so they can use IP to send and receive voice traffic sent via IP packets. To make that work, the switch's Ethernet port acts like an access port, but at the same time, the port acts like a trunk in some ways. This last topic of the chapter works through those main concepts.

Data and Voice VLAN Concepts

Before IP telephony, a PC could sit on the same desk as a phone. The phone happened to use UTP cabling, with that phone connected to some voice device (often called a *voice switch* or a *private branch exchange [PBX]*). The PC, of course, connected using an unshielded twisted-pair (UTP) cable to the usual LAN switch that sat in the wiring closet—sometimes in the same wiring closet as the voice switch. Figure 8-11 shows the idea.

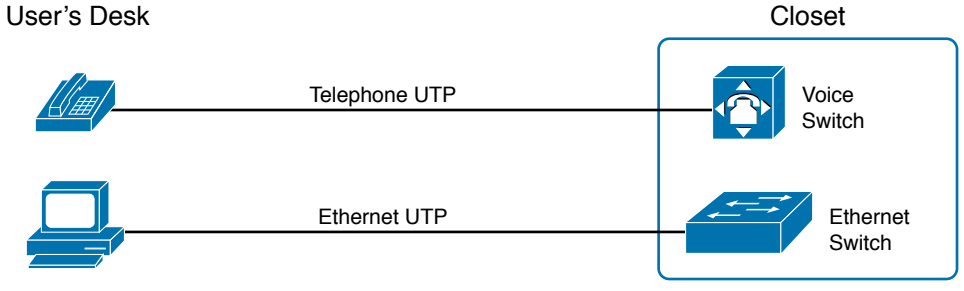


Figure 8-11 *Before IP Telephony: PC and Phone, One Cable Each, Connect to Two Different Devices*

The term *IP telephony* refers to the branch of networking in which the telephones use IP packets to send and receive voice as represented by the bits in the data portion of the IP packet. The phones connect to the network like most other end-user devices, using either Ethernet or Wi-Fi. These new IP phones did not connect via cable directly to a voice switch, instead connecting to the IP network using an Ethernet cable and an Ethernet port built in to the phone. The phones then communicated over the IP network with software that replaced the call setup and other functions of the PBX. (The current products from Cisco that perform this IP telephony control function are called *Cisco Unified Communication Manager*.)

The migration from using the already-installed telephone cabling to these new IP phones that needed UTP cables that supported Ethernet caused some problems in some offices. In particular:

- The older non-IP phones used a category of UTP cabling that often did not support 100-Mbps or 1000-Mbps Ethernet.
- Most offices had a single UTP cable running from the wiring closet to each desk, but now two devices (the PC and the new IP phone) both needed a cable from the desktop to the wiring closet.
- Installing a new cable to every desk would be expensive, plus you would need more switch ports.

To solve this problem, Cisco embedded small three-port switches into each phone.

IP telephones have included a small LAN switch, on the underside of the phone, since the earliest IP telephone products. Figure 8-12 shows the basic cabling, with the wiring closet cable connecting to one physical port on the embedded switch, the PC connecting with a short patch cable to the other physical port, and the phone's internal CPU connecting to an internal switch port.

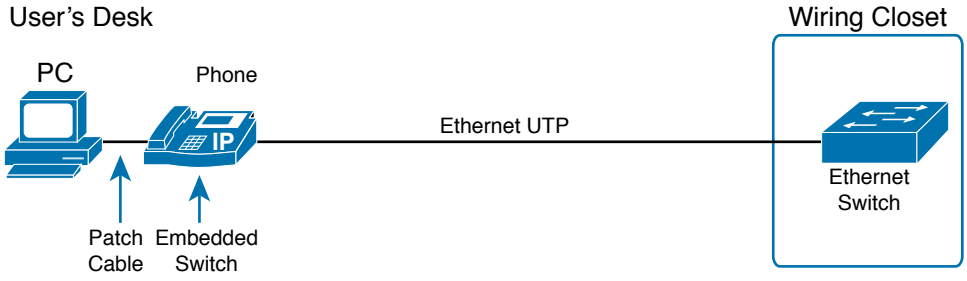


Figure 8-12 *Cabling with an IP Phone, a Single Cable, and an Integrated Switch*

Sites that use IP telephony, which includes almost every company today, now have two devices off each access port. In addition, Cisco best practices for IP telephony design tell us to put the phones in one VLAN and the PCs in a different VLAN. To make that happen, the switch port acts a little like an access link (for the PC's traffic), and a little like a trunk (for the phone's traffic). The configuration defines two VLANs on that port, as follows:

Key Topic

Data VLAN: Same idea and configuration as the access VLAN on an access port but defined as the VLAN on that link for forwarding the traffic for the device connected to the phone on the desk (typically the user's PC).

Voice VLAN: The VLAN defined on the link for forwarding the phone's traffic. Traffic in this VLAN is typically tagged with an 802.1Q header.

Figure 8-13 illustrates this design with two VLANs on access ports that support IP telephones.

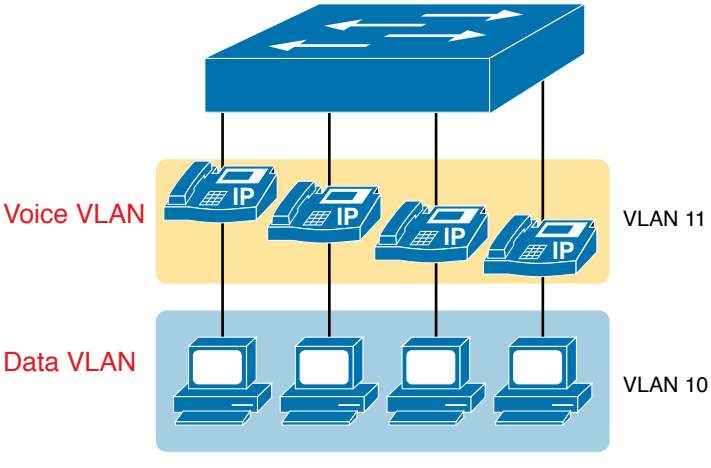


Figure 8-13 *A LAN Design, with Data in VLAN 10 and Phones in VLAN 11*

Data and Voice VLAN Configuration and Verification

Configuring a switch port to support IP phones, once you know the planned voice and data VLAN IDs, requires just a few easy commands. Making sense of the **show** commands once it is configured, however, can be a challenge. The port acts like an access port in many ways. However, with most configuration options, the voice frames flow with an 802.1Q header, so that the link supports frames in both VLANs on the link. But that makes for some different **show** command output.

Example 8-8 shows an example configuration. In this case, all four switch ports F0/1–F0/4 begin with default configuration. The configuration adds the new data and voice VLANs. The example then configures all four ports as access ports and defines the access VLAN, which is also called the *data VLAN* when discussing IP telephony. Finally, the configuration includes the **switchport voice vlan 11** command, which defines the voice VLAN used on the port. The example matches Figure 8-13, using ports F0/1–F0/4.

Example 8-8 *Configuring the Voice and Data VLAN on Ports Connected to Phones*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# vlan 10
SW1(config-vlan)# vlan 11
SW1(config-vlan)# interface range FastEthernet0/1 - 4
SW1(config-if)# switchport mode access
SW1(config-if)# switchport access vlan 10
SW1(config-if)# switchport voice vlan 11
SW1(config-if)#^Z
SW1#
```

NOTE CDP, which is discussed in the *CCNA 200-301 Official Cert Guide, Volume 2*, Chapter 9, "Device Management Protocols," must be enabled on an interface for a voice access port to work with Cisco IP phones. Cisco switches and routers enable CDP by default, so its configuration is not shown here.

The following list details the configuration steps for easier review and study:

Config Checklist

Step 1. Use the **vlan** *vlan-id* command in global configuration mode to create the data and voice VLANs if they do not already exist on the switch.

Step 2. Configure the data VLAN like an access VLAN, as usual:

- **A.** Use the **interface** *type number* command global configuration mode to move into interface configuration mode.
- **B.** Use the **switchport access vlan** *id-number* command in interface configuration mode to define the data VLAN.
- **C.** Use the **switchport mode access** command in interface configuration mode to make this port always operate in access mode (that is, to not trunk).

Step 3. Use the **switchport voice vlan** *id-number* command in interface configuration mode to set the voice VLAN ID.

Verifying the status of a switch port configured like Example 8-8 shows some different output compared to the pure access port and pure trunk port configurations seen earlier in this chapter. For example, the **show interfaces switchport** command shows details about the operation of an interface, including many details about access ports. Example 8-9 shows those details for port F0/4 after the configuration in Example 8-8 was added.

Example 8-9 *Verifying the Data VLAN (Access VLAN) and Voice VLAN*

```
SW1# show interfaces FastEthernet 0/4 switchport
Name: Fa0/4
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 10 (VLAN0010)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: 11 (VLAN0011)
! The rest of the output is omitted for brevity
```

Working through the first three highlighted lines in the output, all those details should look familiar for any access port. The **switchport mode access** configuration command statically configures the administrative mode to be an access port, so the port of course operates as an access port. Also, as shown in the third highlighted line, the **switchport access vlan 10** configuration command defined the access mode VLAN as highlighted here.

The fourth highlighted line shows the one small new piece of information: the voice VLAN ID, as set with the **switchport voice vlan 11** command in this case. This small line of output is the only piece of information in the output that differs from the earlier access port examples in this chapter.

These ports act more like access ports than trunk ports. In fact, the **show interfaces** *type number* **switchport** command boldly proclaims, "Operational Mode: static access." However, one other **show** command reveals just a little more about the underlying operation with 802.1Q tagging for the voice frames.

As mentioned earlier, the **show interfaces trunk** command—that is, the command that does not include a specific interface in the middle of the command—lists the operational trunks on a switch. With IP telephony ports, the ports do not show up in the list of trunks either—providing evidence that these links are *not* treated as trunks. Example 8-10 shows just such an example.

However, the **show interfaces trunk** command with the interface listed in the middle of the command, as is also shown in Example 8-10, does list some additional information. Note that in this case, the **show interfaces F0/4 trunk** command lists the status as not-trunking, but with VLANs 10 and 11 allowed on the trunk. (Normally, on an access port, only the access VLAN is listed in the "VLANs allowed on the trunk" list in the output of this command.)

Example 8-10 *Allowed VLAN List and the List of Active VLANs*

```
SW1# show interfaces trunk
SW1# show interfaces F0/4 trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/4       off              802.1q         not-trunking  1

Port        Vlans allowed on trunk
Fa0/4       10-11

Port        Vlans allowed and active in management domain
Fa0/4       10-11

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/4       10-11
```

Summary: IP Telephony Ports on Switches

It might seem as though this short topic about IP telephony and switch configuration includes a lot of small twists and turns and trivia, and it does. The most important items to remember are as follows:

- Configure these ports like a normal access port to begin: Configure it as a static access port and assign it an access VLAN.
- Add one more command to define the voice VLAN (**switchport voice vlan** *vlan-id*).
- Look for the mention of the voice VLAN ID, but no other new facts, in the output of the **show interfaces** *type number* **switchport** command.
- Look for both the voice and data (access) VLAN IDs in the output of the **show interfaces** *type number* **trunk** command.
- Do not expect to see the port listed in the list of operational trunks as listed by the **show interfaces trunk** command.

Troubleshooting VLANs and VLAN Trunks

A switch's data plane forwarding processes depend in part on VLANs and VLAN trunking. This final section of the chapter focuses on issues related to VLANs and VLAN trunks that could prevent LAN switching from working properly, focusing on a few items not yet discussed in the chapter. In particular, this section examines these steps an engineer can take to avoid issues:

Step 1. Confirm that all VLANs are both defined and active.

Step 2. Check the allowed VLAN lists on both ends of each trunk to ensure that all VLANs intended to be used are included.

Step 3. Check for incorrect trunk configuration settings that result in one switch operating as a trunk, with the neighboring switch not operating as a trunk.

Step 4. Check the native VLAN settings on both ends of the trunk to ensure the settings match.

Access VLANs Undefined or Disabled

Switches do not forward frames for VLANs that are (a) not known because the VLAN is not configured or has not been learned with VTP or (b) the VLAN is known, but it is disabled (shut down). This next topic summarizes the best ways to confirm that a switch knows that a particular VLAN exists, and if it exists, determines the shutdown state of the VLAN.

First, on the issue of whether a VLAN exists on a switch, a VLAN can be defined to a switch in two ways: using the **vlan** *number* global configuration command, or it can be learned from another switch using VTP. As mentioned earlier in this chapter, the examples in this book assume that you are not using VTP. If you discover that a VLAN does not exist on a switch, simply configure the VLAN as discussed earlier in the section, "Creating VLANs and Assigning Access VLANs to an Interface."

In addition to checking the configuration, you can check for the status of the VLAN (as well as whether it is known to the switch) using the **show vlan** command. No matter the VTP mode, this command will list all VLANs known to the switch, plus one of two VLAN state values, depending on the current state: either *active* or *act/lshut*. The second of these states means that the VLAN is shut down. Shutting down a VLAN disables the VLAN on that switch only, so *the switch will not forward frames in that VLAN*.

Switch IOS gives you two similar configuration methods with which to disable (**shutdown**) and enable (**no shutdown**) a VLAN. Example 8-11 shows how, first by using the global command **[no] shutdown vlan** *number* and then using the VLAN mode subcommand **[no] shutdown**. The example shows the global commands enabling and disabling VLANs 10 and 20, respectively, and using VLAN subcommands to enable and disable VLANs 30 and 40, respectively.

Example 8-11 *Enabling and Disabling VLANs on a Switch*

```
SW2# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                Fa0/14, Fa0/15, Fa0/16, Fa0/17
                                                Fa0/18, Fa0/19, Fa0/20, Fa0/21
                                                Fa0/22, Fa0/23, Fa0/24, Gi0/1
10   VLAN0010                         act/lshut Fa0/13
20   VLAN0020                         active
30   VLAN0030                         act/lshut
```

```
40   VLAN0040                         active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

SW2# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW2(config)# no shutdown vlan 10
SW2(config)# shutdown vlan 20
SW2(config)# vlan 30
SW2(config-vlan)# no shutdown
SW2(config-vlan)# vlan 40
SW2(config-vlan)# shutdown
SW2(config-vlan)#
```

NOTE The output of the **show vlan brief** command also lists a state of "act/unsup" for the reserved VLAN IDs 1002–1005, with "unsup" meaning "unsupported."

Mismatched Trunking Operational States

Trunking can be configured correctly so that both switches use trunking. However, trunks can also be misconfigured, with a couple of different results: either both switches do not trunk, or one switch trunks and the other does not. Both results cause problems.

The most common incorrect configuration—which results in both switches not trunking—is a configuration that uses the **switchport mode dynamic auto** command on both switches on the link. The word *auto* just makes us all want to think that the link would trunk automatically, but this command is both automatic and passive. As a result, both switches passively wait on the other device on the link to begin negotiations. Example 8-12 highlights those parts of the output from the **show interfaces switchport** command that confirm both the configured and operational states. Note that the output lists the operational mode as "static access" rather than "trunking."

Example 8-12 *Operational Trunking State*

```
SW2# show interfaces gigabit0/2 switchport
Name: Gi0/2
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
! lines omitted for brevity
```

A different incorrect trunking configuration has an even worse result: one switch trunks, sending tagged frames, while the neighboring switch does not trunk, so the neighboring switch discards any frames it receives that have a VLAN tag in the header. When this combination of events happens, the interface works in that the status on each end will be up/up or connected. Traffic in the native VLAN will actually cross the link successfully because those frames have no VLAN tags (headers). However, traffic in all the rest of the VLANs will not cross the link.

Figure 8-14 shows the incorrect configuration along with which side trunks and which does not. The side that trunks (SW1 in this case) enables trunking using the command **switchport mode trunk** but also disables Dynamic Trunking Protocol (DTP) negotiations using the **switchport nonegotiate** command. SW2's configuration also helps create the problem, by using one of the two trunking options that relies on DTP. Because SW1 has disabled DTP, SW2's DTP negotiations fail, and SW2 chooses to not trunk.

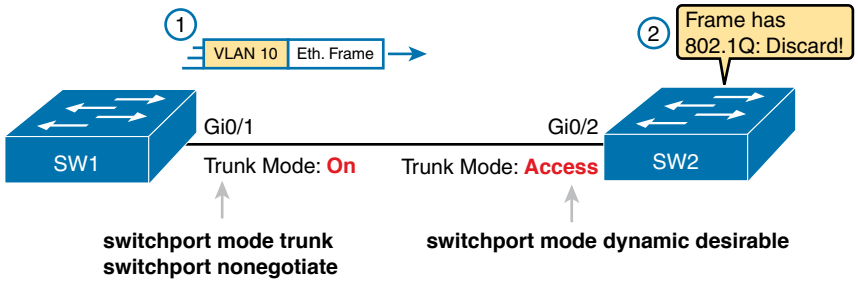


Figure 8-14 *Mismatched Trunking Operational States*

The figure shows what happens when using this incorrect configuration. At Step 1, SW1 could (for example) forward a frame in VLAN 10. However, SW2 would view any frame that arrives with an 802.1Q header as illegal because the frame has an 802.1Q header, and SW2 treats its G0/2 port as an access port. So, SW2 discards any 802.1Q frames received on that port.

The trunking issues shown here can be easily avoided by checking the configuration and by checking the trunk's operational state (mode) on both sides of the trunk. The best commands to check trunking-related facts are **show interfaces trunk** and **show interfaces switchport**. Just be aware that the switches do not prevent you from making these configuration mistakes.

The Supported VLAN List on Trunks

A Cisco switch can forward traffic for all defined and active VLANs. However, a particular VLAN trunk may not forward traffic for a defined and active VLAN for a variety of other reasons. You should learn how to identify which VLANs a particular trunk port currently supports and the reasons why the switch might not be forwarding frames for a VLAN on that trunk port.

The first category in this step can be easily done using the **show interfaces** *interface-id* **trunk** command, which only lists information about currently operational trunks. The best place to begin with this command is the last section of output, which lists the VLANs whose

traffic will be forwarded over the trunk. Any VLANs that make it to this final list of VLANs in the command output meet the following criteria:

- The VLAN has not been removed from the *allowed VLAN list* on the trunk (as configured with the **switchport trunk allowed vlan** interface subcommand).
- The VLAN exists and is active on the local switch (as seen in the **show vlan** command).
- The VLAN has not been VTP-pruned from the trunk. (Because this book attempts to ignore VTP as much as possible, this section assumes that VTP is not used and this feature has no impact on any trunks.) The trunk is in an STP forwarding state in that VLAN (as also seen in the **show spanning-tree vlan** *vlan-id* command).

The **switchport trunk allowed vlan** interface subcommand gives the network engineer a method to administratively limit the VLANs whose traffic uses a trunk. If the engineer wants all defined VLANs to be supported on a trunk, the engineer simply does not configure this command. If the engineer would like to limit the trunk to support a subset of the VLANs known to the switch, however, the engineer can add one or more **switchport trunk allowed vlan** interface subcommands.

For instance, in a switch that has configured VLANs 1 through 100, but no others, by default the switch would allow traffic in all 100 VLANs. However, the trunk interface command **switchport trunk allowed vlan 1-60** would limit the trunk to forward traffic for VLANs 1 through 60, but not the rest of the VLANs. Example 8-13 shows a sample of the command output from the **show interfaces trunk** command, which confirms the first list of VLAN IDs now lists VLANs 1–60. Without the **switchport trunk allowed vlan** command, the first list would have included VLANs 1–4094.

Example 8-13 *Allowed VLAN List and List of Active VLANs*

```
SW1# show interfaces trunk

Port        Mode         Encapsulation   Status       Native vlan
Gi0/1       desirable    802.1q          trunking     1

Port        Vlans allowed on trunk
Gi0/1       1-60

Port        Vlans allowed and active in management domain
Gi0/1       1-59

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/1       1-58
```

The output of the **show interfaces trunk** command creates three separate lists of VLANs, each under a separate heading. These three lists show a progression of reasons why a VLAN is not forwarded over a trunk. Table 8-4 summarizes the headings that precede each list and the reasons why a switch chooses to include or not include a VLAN in each list. For instance, in Example 8-13, VLAN 60 has been shut down, and VLAN 59 happens to be in

an STP blocking state. (Chapter 9, "Spanning Tree Protocol Concepts," has more information about STP.)

Key Topic

Table 8-4 VLAN Lists in the **show interfaces trunk** Command

List Position	Heading	Reasons
First	VLANs allowed	VLANs 1–4094, minus those removed by the **switchport trunk allowed** command
Second	VLANs allowed and active…	The first list, minus VLANs not defined to the local switch (that is, there is not a **vlan** global configuration command or the switch has not learned of the VLAN with VTP), and also minus those VLANs in shutdown mode
Third	VLANs in spanning tree…	The second list, minus VLANs in an STP blocking state for that interface, and minus VLANs VTP pruned from that trunk

NOTE The companion website includes a video from the CCNA Exam Prep LiveLessons product, named "Troubleshooting VLANs Allowed on a Trunk #1," which works through the three lists of VLANs in the output of the **show interfaces** *interface-id* **trunk** command in more detail.

Mismatched Native VLAN on a Trunk

Unfortunately, it *is* possible to set the native VLAN ID to different VLANs on either end of the trunk, using the **switchport trunk native vlan** *vlan-id* command. If the native VLANs differ according to the two neighboring switches, the switches will cause frames sent in the native VLAN to jump from one VLAN to the other.

For example, if switch SW1 sends a frame using native VLAN 1 on an 802.1Q trunk, SW1 does not add a VLAN header, as is normal for the native VLAN. When switch SW2 receives the frame, noticing that no 802.1Q header exists, SW2 assumes that the frame is part of SW2's configured native VLAN. If SW2 has been configured to think VLAN 2 is the native VLAN on that trunk, SW2 will try to forward the received frame into VLAN 2. (This effect of a frame being sent in one VLAN but then being believed to be in a different VLAN is called *VLAN hopping*.)

Chapter Review

Review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 8-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 8-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review config checklists		Book, website
Review command tables		Book
Review memory tables		Website
Do labs		Sim Lite, blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 8-6 Key Topics for Chapter 8

Key Topic Element	Description	Page Number
Figure 8-2	Basic VLAN concept	179
List	Reasons for using VLANs	180
Figure 8-5	Diagram of VLAN trunking	182
Figure 8-6	802.1Q header	183
Table 8-2	Options of the **switchport mode** command	191
Table 8-3	Expected trunking results based on the configuration of the **switchport mode** command	195
List	Definitions of data VLAN and voice VLAN	197
List	Summary of data and voice VLAN concepts, configuration, and verification	200
Table 8-4	Analysis of the three VLAN lists in the output from the **show interfaces** *interface-id* **trunk** command	205

Key Terms You Should Know

802.1Q, trunk, trunking administrative mode, trunking operational mode, VLAN, VTP, VTP transparent mode, Layer 3 switch, access interface, trunk interface, data VLAN, voice VLAN, native VLAN, default VLAN, static access interface

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included free with this book. The Sim Lite with this book includes a couple of labs about VLANs. Also, check the author's blog site pages for configuration exercises (Config Labs) at https://blog.certskills.com.

Command References

Tables 8-7 and 8-8 list configuration and verification commands used in this chapter, respectively. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 8-7 Chapter 8 Configuration Command Reference

Command	Description
vlan *vlan-id*	Global config command that both creates the VLAN and puts the CLI into VLAN configuration mode
name *vlan-name*	VLAN subcommand that names the VLAN
[**no**] **shutdown**	VLAN mode subcommand that enables (**no shutdown**) or disables (**shutdown**) the VLAN
[**no**] **shutdown vlan** *vlan-id*	Global config command that has the same effect as the [**no**] **shutdown** VLAN mode subcommands
vtp mode {**server** | **client** | **transparent** | **off**}	Global config command that defines the VTP mode
switchport mode {**access** | **dynamic** {**auto** | **desirable**} | **trunk**}	Interface subcommand that configures the trunking administrative mode on the interface
switchport access vlan *vlan-id*	Interface subcommand that statically configures the interface into that one VLAN
switchport trunk encapsulation {**dot1q** | **isl** | **negotiate**}	Interface subcommand that defines which type of trunking to use, assuming that trunking is configured or negotiated
switchport trunk native vlan *vlan-id*	Interface subcommand that defines the native VLAN for a trunk port
switchport nonegotiate	Interface subcommand that disables the negotiation of VLAN trunking
switchport voice vlan *vlan-id*	Interface subcommand that defines the voice VLAN on a port, meaning that the switch uses 802.1Q tagging for frames in this VLAN
switchport trunk allowed vlan {**add** | **all** | **except** | **remove**} *vlan-list*	Interface subcommand that defines the list of allowed VLANs

Table 8-8 Chapter 8 EXEC Command Reference

Command	Description
show interfaces *interface-id* **switchport**	Lists information about any interface regarding administrative settings and operational state
show interfaces *interface-id* **trunk**	Lists information about all operational trunks (but no other interfaces), including the list of VLANs that can be forwarded over the trunk
show vlan [**brief** | **id** *vlan-id* | **name** *vlan-name* | **summary**]	Lists information about the VLAN
show vlan [*vlan*]	Displays VLAN information
show vtp status	Lists VTP configuration and status information

This page intentionally left blank

CHAPTER 9

Spanning Tree Protocol Concepts

This chapter covers the following exam topics:

2.0 Network Access

2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)

2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations

2.5.a Root port, root bridge (primary/secondary), and other port names

2.5.b Port states (forwarding/blocking)

2.5.c PortFast benefits

Spanning Tree Protocol (STP) allows Ethernet LANs to have the added benefits of installing redundant links in a LAN, while overcoming the known problems that occur when adding those extra links. Using redundant links in a LAN design allows the LAN to keep working even when some links fail or even when some entire switches fail. Proper LAN design should add enough redundancy so that no single point of failure crashes the LAN; STP allows the design to use redundancy without causing some other problems.

Historically, the IEEE first standardized STP as part of the IEEE 802.1D standard back in 1990, with pre-standard versions working even before that time. Over time, the industry and IEEE improved STP, with the eventual replacement of STP with an improved protocol: Rapid Spanning Tree Protocol (RSTP). The IEEE first released RSTP as amendment 802.1w and, in 2004, integrated RSTP into the 802.1D standard.

An argument could be made to ignore STP today and instead focus solely on RSTP. Most modern networks use RSTP instead of STP. The most recent models and IOS versions of Cisco switches default to use RSTP instead of STP. Plus, the CCNA 200-301 exam topics mention RSTP by name, but not STP. However, STP and RSTP share many of the same mechanisms, and RSTP's improvements can be best understood in comparison to STP. For that reason, this chapter presents some details that apply only to STP, as a learning tool to help you understand RSTP.

This chapter organizes the material into three sections. The first section presents some core concepts about how both STP and RSTP discover a tree made of nodes (switches) and links so that no loops exist in a network. The second section then takes a brief look at the area for which STP differs the most from RSTP: in how STP reacts to changes in the network. This chapter ends with a third major section that details RSTP, including how RSTP works much better that STP when reacting to changes.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom

of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 9-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
STP and RSTP Basics	1–2
Details Specific to STP (and Not RSTP)	3–4
Rapid STP Concepts	5–7

1. Which of the following port states are stable states used when STP has completed convergence? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding
2. Which of the following bridge IDs wins election as root, assuming that the switches with these bridge IDs are in the same network?
   a. 32769:0200.1111.1111
   b. 32769:0200.2222.2222
   c. 4097:0200.1111.1111
   d. 4097:0200.2222.2222
   e. 40961:0200.1111.1111
3. Which of the following are transitory port states used only during the process of STP convergence? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding
4. Which of the following facts determines how often a nonroot bridge or switch sends an STP Hello BPDU message?
   a. The Hello timer as configured on that switch.
   b. The Hello timer as configured on the root switch.
   c. It is always every 2 seconds.
   d. The switch reacts to BPDUs received from the root switch by sending another BPDU 2 seconds after receiving the root BPDU.

5. Which of the following RSTP port states have the same name and purpose as a port state in traditional STP? (Choose two answers.)
   a. Blocking
   b. Forwarding
   c. Listening
   d. Learning
   e. Discarding
6. RSTP adds features beyond STP that enable ports to be used for a role if another port on the same switch fails. Which of the following statements correctly describe a port role that is waiting to take over for another port role? (Choose two answers.)
   a. An alternate port waits to become a root port.
   b. A backup port waits to become a root port.
   c. An alternate port waits to become a designated port.
   d. A backup port waits to become a designated port.
7. What STP feature causes an interface to be placed in the forwarding state as soon as the interface is physically active?
   a. STP
   b. EtherChannel
   c. Root Guard
   d. PortFast

Foundation Topics

STP and RSTP Basics

Without some mechanism like Spanning Tree Protocol (STP) or Rapid STP (RSTP), a LAN with redundant links would cause Ethernet frames to loop for an indefinite period of time. With STP or RSTP enabled, some switches block ports so that these ports do not forward frames. STP and RSTP intelligently choose which ports block, with two goals in mind:

- All devices in a VLAN can send frames to all other devices. In other words, STP or RSTP does not block too many ports, cutting off some parts of the LAN from other parts.
- Frames have a short life and do not loop around the network indefinitely.

STP and RSTP strike a balance, allowing frames to be delivered to each device, without causing the problems that occur when frames loop through the network over and over again.

NOTE This first major section of the chapter explains details of both STP and RSTP, so this section uses the term *STP/RSTP* to refer to these protocols together. Note that this term is just a convenient shorthand. Later in the chapter, the text will point out differences between STP and RSTP and begin using the terms *STP* and *RSTP* separately, referring to only the specific protocol.

STP/RSTP prevents looping frames by adding an additional check on each interface before a switch uses it to send or receive user traffic. That check: If the port is in STP/RSTP forwarding state in that VLAN, use it as normal; if it is in STP/RSTP blocking state, however, block all user traffic and do not send or receive user traffic on that interface in that VLAN.

Note that these STP/RSTP states do not change the other information you already know about switch interfaces. The interface's state of connected/notconnect does not change. The interface's operational state as either an access or trunk port does not change. STP/RSTP adds this additional state, with the blocking state basically disabling the interface.

In many ways, those last two paragraphs sum up what STP/RSTP does. However, the details of how STP/RSTP does its work can take a fair amount of study and practice. This first major section of the chapter begins by explaining the need for STP/RSTP and the basic ideas of what STP/RSTP does to solve the problem of looping frames. The majority of this section then looks at how STP/RSTP goes about choosing which switch ports to block to accomplish its goals.

The Need for Spanning Tree

STP/RSTP prevents three common problems in Ethernet LANs. All three problems occur as a side effect of one fact: without STP/RSTP, some Ethernet frames would loop around the network for a long time (hours, days, literally forever if the LAN devices and links never failed).

Just one looping frame causes what is called a *broadcast storm*. Broadcast storms happen when any kind of Ethernet frames—broadcast frames, multicast frames, or unknown-destination unicast frames—loop around a LAN indefinitely. Broadcast storms can saturate all the links with copies of that one single frame, crowding out good frames, as well as significantly impacting end-user device performance by making the PCs process too many broadcast frames.

To help you understand how this occurs, Figure 9-1 shows a sample network in which Bob sends a broadcast frame. The dashed lines show how the switches forward the frame when STP/RSTP does not exist.

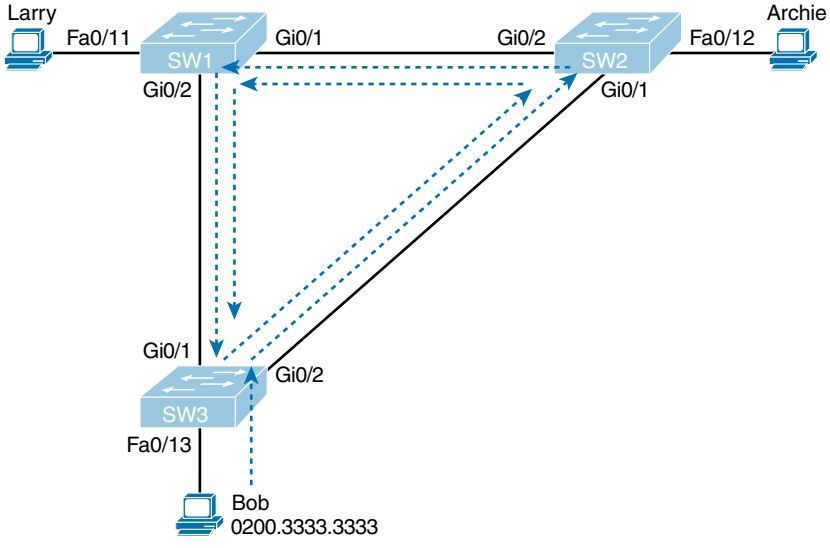


Figure 9-1 *Broadcast Storm*

NOTE Bob's original broadcast would also be forwarded around the other direction, with SW3 sending a copy of the original frame out its Gi0/1 port. To reduce clutter, Figure 9-1 does not show that frame.

Remember that LAN switch? That logic tells switches to flood broadcasts out all interfaces in the same VLAN except the interface in which the frame arrived. In Figure 9-1, that means SW3 forwards Bob's frame to SW2, SW2 forwards the frame to SW1, SW1 forwards the frame back to SW3, and SW3 forwards it back to SW2 again.

When broadcast storms happen, frames like the one in Figure 9-1 keep looping until something changes—someone shuts down an interface, reloads a switch, or does something else to break the loop. Also note that the same event happens in the opposite direction. When Bob sends the original frame, SW3 also forwards a copy to SW1, SW1 forwards it to SW2, and so on.

The storm also causes a much more subtle problem called *MAC table instability*. MAC table instability means that the switches' MAC address tables keep changing because frames with the same source MAC arrive on different ports. To see why, follow this example, in which SW3 begins Figure 9-1 with a MAC table entry for Bob, at the bottom of the figure, associated with port Fa0/13:

0200.3333.3333 Fa0/13 VLAN 1

However, now think about the switch-learning process that occurs when the looping frame goes to SW2, then SW1, and then back into SW3's Gi0/1 interface. SW3 thinks, "Hmm…the source MAC address is 0200.3333.3333, and it came in my Gi0/1 interface. Update my MAC table!" This results in the following entry on SW3, with interface Gi0/1 instead of Fa0/13:

0200.3333.3333 Gi0/1 VLAN 1

At this point, SW3 itself cannot correctly deliver frames to Bob's MAC address. At that instant, if a frame arrives at SW3 destined for Bob—a different frame than the looping frame that causes the problems—SW3 incorrectly forwards the frame out Gi0/1 to SW1, creating even more congestion.

The looping frames in a broadcast storm also cause a third problem: multiple copies of the frame arrive at the destination. Consider a case in which Bob sends a frame to Larry but none of the switches know Larry's MAC address. Switches flood frames sent to unknown destination unicast MAC addresses. When Bob sends the frame destined for Larry's MAC address, SW3 sends a copy to both SW1 and SW2. SW1 and SW2 also flood the frame, causing copies of the frame to loop. SW1 also sends a copy of each frame out Fa0/11 to Larry. As a result, Larry gets multiple copies of the frame, which may result in an application failure, if not more pervasive networking problems.

Table 9-2 summarizes the main three classes of problems that occur when STP/RSTP is not used in a LAN that has redundancy.

Answers to the "Do I Know This Already?" quiz:

1 A, B **2** C **3** C, D **4** B **5** B, D **6** A, D **7** D

Key Topic

Table 9-2 Three Classes of Problems Caused by Not Using STP in Redundant LANs

Problem	Description
Broadcast storms	The forwarding of a frame repeatedly on the same links, consuming significant parts of the links' capacities
MAC table instability	The continual updating of a switch's MAC address table with incorrect entries, in reaction to looping frames, resulting in frames being sent to the wrong locations
Multiple frame transmission	A side effect of looping frames in which multiple copies of one frame are delivered to the intended host, confusing the host

What Spanning Tree Does

STP/RSTP prevents loops by placing each switch port in either a forwarding state or a blocking state. Interfaces in the forwarding state act as normal, forwarding and receiving frames. However, interfaces in a blocking state do not process any frames except STP/RSTP messages (and some other overhead messages). Interfaces that block do not forward user frames, do not learn MAC addresses of received frames, and do not process received user frames.

Figure 9-2 shows a simple STP/RSTP tree that solves the problem shown in Figure 9-1 by placing one port on SW3 in the blocking state.

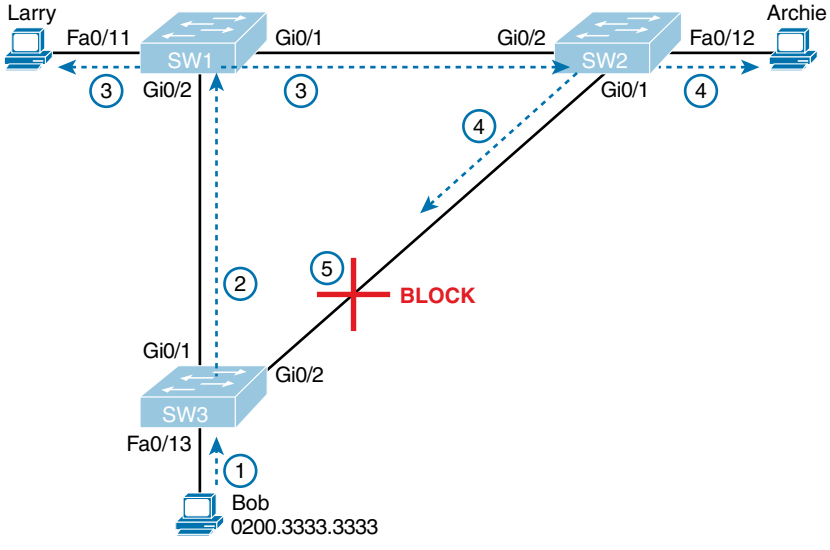


Figure 9-2 *What STP/RSTP Does: Blocks a Port to Break the Loop*

Now when Bob sends a broadcast frame, the frame does not loop. As shown in the steps in the figure:

Step 1. Bob sends the frame to SW3.

Step 2. SW3 forwards the frame only to SW1, but not out Gi0/2 to SW2, because SW3's Gi0/2 interface is in a blocking state.

Step 3. SW1 floods the frame out both Fa0/11 and Gi0/1.

Step 4. SW2 floods the frame out Fa0/12 and Gi0/1.

Step 5. SW3 physically receives the frame, but it ignores the frame received from SW2 because SW3's Gi0/2 interface is in a blocking state.

With the STP/RSTP topology in Figure 9-2, the switches simply do not use the link between SW2 and SW3 for traffic in this VLAN, which is the minor negative side effect of STP. However, if either of the other two links fails, STP/RSTP converges so that SW3 forwards instead of blocks on its Gi0/2 interface.

NOTE The term *STP convergence* refers to the process by which the switches collectively realize that something has changed in the LAN topology and determine whether they need to change which ports block and which ports forward.

That completes the description of what STP/RSTP does, placing each port into either a forwarding or blocking state. The more interesting question, and the one that takes a lot more work to understand, is how and why STP/RSTP makes its choices. How does STP/RSTP manage to make switches block or forward on each interface? And how does it converge to change state from blocking to forwarding to take advantage of redundant links in response to network outages? The following pages answer these questions.

How Spanning Tree Works

The STP/RSTP algorithm creates a spanning tree of interfaces that forward frames. The tree structure of forwarding interfaces creates a single path to and from each Ethernet link, just like you can trace a single path in a living, growing tree from the base of the tree to each leaf.

NOTE STP was created before LAN switches even existed, using LAN *bridges* to connect LANs. Today, switches play the same role as bridges, implementing STP/RSTP. However, many STP/RSTP terms still refer to bridge. For the purposes of STP/RSTP and this chapter, consider the terms *bridge* and *switch* synonymous.

The process used by STP, sometimes called the *spanning-tree algorithm* (STA), chooses the interfaces that should be placed into a forwarding state. For any interfaces not chosen to be in a forwarding state, STP/RSTP places the interfaces in blocking state. In other words, STP/RSTP simply picks which interfaces should forward, and any interfaces left over go to a blocking state.

STP/RSTP uses three criteria to choose whether to put an interface in forwarding state:

- STP/RSTP elects a root switch. STP puts all working interfaces on the root switch in forwarding state.
- Each nonroot switch considers one of its ports to have the least administrative cost between itself and the root switch. The cost is called that switch's *root cost*. STP/RSTP

places its port that is part of the least root cost path, called that switch's *root port* (RP), in forwarding state.

- Many switches can attach to the same Ethernet segment, but due to the fact that links connect two devices, a link would have at most two switches. With two switches on a link, the switch with the lowest root cost, as compared with the other switches attached to the same link, is placed in forwarding state. That switch is the designated switch, and that switch's interface, attached to that segment, is called the *designated port* (DP).

NOTE The real reason the root switches place all working interfaces in a forwarding state (at step 1 in the list) is that all its interfaces on the root switch will become DPs. However, it is easier to just remember that all the root switches' working interfaces will forward frames.

All other interfaces are placed in blocking state. Table 9-3 summarizes the reasons STP/RSTP places a port in forwarding or blocking state.

Key Topic

Table 9-3 STP/RSTP: Reasons for Forwarding or Blocking

Characterization of Port	STP State	Description
All the root switch's ports	Forwarding	The root switch is always the designated switch on all connected segments.
Each nonroot switch's root port	Forwarding	The port through which the switch has the least cost to reach the root switch (lowest root cost).
Each LAN's designated port	Forwarding	The switch forwarding the Hello on to the segment, with the lowest root cost, is the designated switch for that segment.
All other working ports	Blocking	The port is not used for forwarding user frames, nor are any frames received on these interfaces considered for forwarding.

NOTE STP/RSTP only considers working interfaces (those in a connected state). Failed interfaces (for example, interfaces with no cable installed) or administratively shutdown interfaces are instead placed into an STP/RSTP *disabled* state. So, this section uses the term *working ports* to refer to interfaces that could forward frames if STP/RSTP placed the interface into a forwarding state.

NOTE STP and RSTP do differ slightly in the use of the names of some states like blocking and disabled, with RSTP using the status term *discarding*. However, those minor differences do not change the meaning of the discussions in this first section of the chapter. The upcoming section titled "Comparing STP and RSTP" discusses these differences, both important and minor.

The STP Bridge ID and Hello BPDU

The STA begins with an election of one switch to be the root switch. To better understand this election process, you need to understand the STP/RSTP messages sent between switches as well as the concept and format of the identifier used to uniquely identify each switch.

The STP/RSTP *bridge ID* (BID) is an 8-byte value unique to each switch. The bridge ID consists of a 2-byte priority field and a 6-byte system ID, with the system ID being based on a universal (burned-in) MAC address in each switch. Using a burned-in MAC address ensures that each switch's bridge ID will be unique.

STP/RSTP defines messages called *bridge protocol data units* (BPDU), also called configuration BPDUs, which switches use to exchange information with each other. The most common BPDU, called a Hello BPDU, lists many details, including the sending switch's BID. By listing its own unique BID, switches can tell which switch sent which Hello BPDU. Table 9-4 lists some of the key information in the Hello BPDU.

Key Topic

Table 9-4 Fields in the STP Hello BPDU

Field	Description
Root bridge ID	The bridge ID of the switch the sender of this Hello currently believes to be the root switch
Sender's bridge ID	The bridge ID of the switch sending this Hello BPDU
Sender's root cost	The STP/RSTP cost between this switch and the current root
Timer values on the root switch	Includes the Hello timer, MaxAge timer, and forward delay timer

For the time being, just keep the first three items from Table 9-4 in mind as the following sections work through the three steps in how STP/RSTP chooses the interfaces to place into a forwarding state. Next, the text examines the three main steps in the STP/RSTP process.

Electing the Root Switch

Switches elect a root switch based on the BIDs in the BPDUs. The root switch is the switch with the lowest numeric value for the BID. Because the two-part BID starts with the priority value, essentially the switch with the lowest priority becomes the root. For example, if one switch has priority 4096, and another switch has priority 8192, the switch with priority 4096 wins, regardless of what MAC address was used to create the BID for each switch.

If a tie occurs based on the priority portion of the BID, the switch with the lowest MAC address portion of the BID is the root. No other tiebreaker should be needed because switches use one of their own universal (burned-in) MAC addresses as the second part of their BIDs. So if the priorities tie, and one switch uses a MAC address of 0200.0000.0000 as part of the BID and the other uses 0811.1111.1111, the first switch (MAC 0200.0000.0000) becomes the root switch.

STP/RSTP elects a root switch in a manner not unlike a political election. The process begins with all switches claiming to be the root by sending Hello BPDUs listing their own BID as the root BID. If a switch hears a Hello that lists a better (lower) BID, that switch stops

advertising itself as root and starts forwarding the superior Hello. The Hello sent by the better switch lists the better switch's BID as the root. It works like a political race in which a less-popular candidate gives up and leaves the race, throwing his support behind the more popular candidate. Eventually, everyone agrees which switch has the best (lowest) BID, and everyone supports the elected switch—which is where the political race analogy falls apart.

NOTE A better Hello, meaning that the listed root's BID is better (numerically lower), is called a *superior Hello*; a worse Hello, meaning that the listed root's BID is not as good (numerically higher), is called an *inferior Hello*.

Figure 9-3 shows the beginning of the root election process. In this case, SW1 has advertised itself as root, as have SW2 and SW3. However, SW2 now believes that SW1 is a better root, so SW2 is now forwarding the Hello originating at SW1. So, at this point, the figure shows SW1 is saying Hello, claiming to be root; SW2 agrees and is forwarding SW1's Hello that lists SW1 as root; but SW3 is still claiming to be best, sending its own Hello BPDUs, listing SW3's BID as the root.

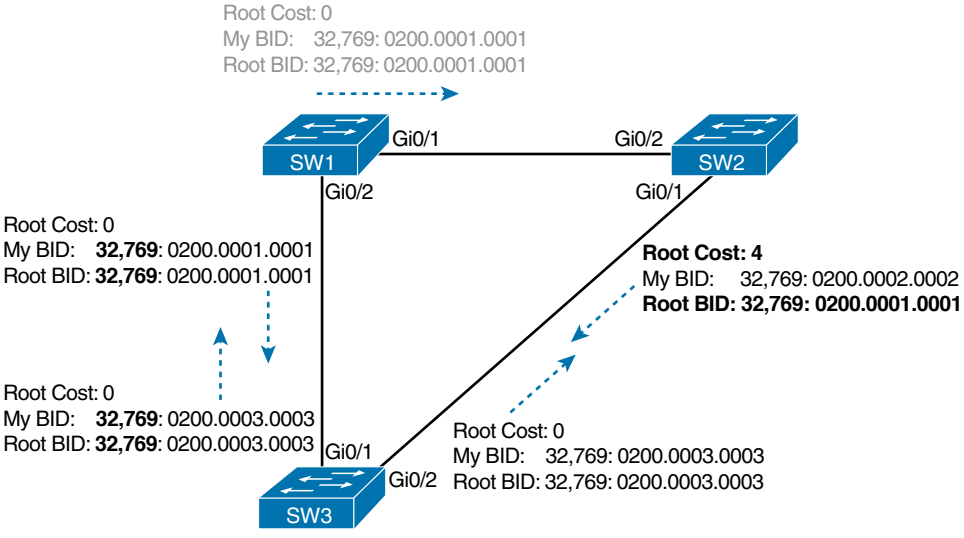


Figure 9-3 *Beginnings of the Root Election Process*

Two candidates still exist in Figure 9-3: SW1 and SW3. So, who wins? Well, from the BID, the lower-priority switch wins; if a tie occurs, the lower MAC address wins. As shown in the figure, SW1 has a lower BID (32769:0200.0001.0001) than SW3 (32769:0200.0003.0003), so SW1 wins, and SW3 now also believes that SW1 is the better switch. Figure 9-4 shows the resulting Hello messages sent by the switches.

Summarizing, the root election happens through each switch claiming to be root, with the best switch being elected based on the numerically lowest BID. Breaking down the BID into its components, the comparisons can be made as

Key Topic

- The lowest priority
- If that ties, the lowest switch MAC address

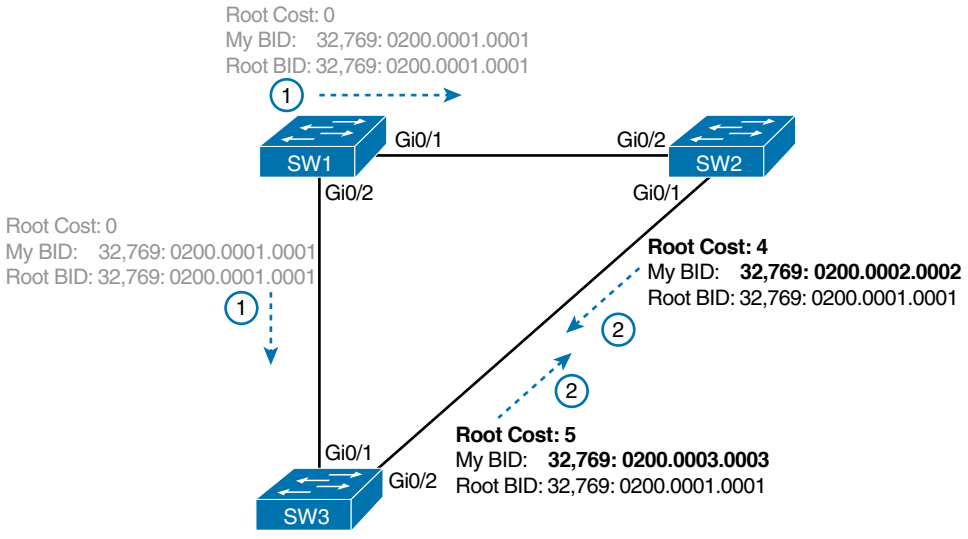


Figure 9-4 *SW1 Wins the Election*

Choosing Each Switch's Root Port

The second part of the STP/RSTP process occurs when each nonroot switch chooses its one and only *root port*. A switch's RP is its interface through which it has the least STP/RSTP cost to reach the root switch (least root cost).

The idea of a switch's cost to reach the root switch can be easily seen for humans. Just look at a network diagram that shows the root switch, lists the STP/RSTP cost associated with each switch port, and identifies the nonroot switch in question. Switches use a different process than looking at a network diagram, of course, but using a diagram can make it easier to learn the idea.

Figure 9-5 shows just such a figure, with the same three switches shown in the last several figures. SW1 has already won the election as root, and the figure considers the cost from SW3's perspective. (Note that the figure uses some nondefault cost settings.)

SW3 has two possible physical paths to send frames to the root switch: the direct path to the left and the indirect path to the right through switch SW2. The cost is the sum of the costs of all the *switch ports the frame would exit* if it flowed over that path. (The calculation ignores the inbound ports.) As you can see, the cost over the direct path out SW3's G0/1 port has a total cost of 5, and the other path has a total cost of 8. SW3 picks its G0/1 port as root port because it is the port that is part of the least-cost path to send frames to the root switch.

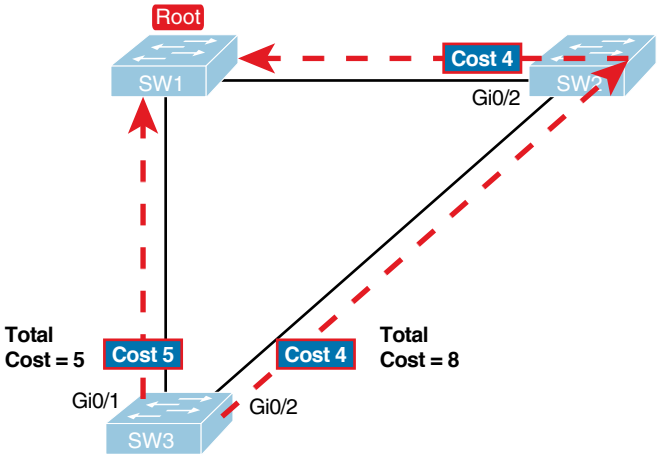


Figure 9-5 *How a Human Might Calculate STP/RSTP Cost from SW3 to the Root (SW1)*

Switches come to the same conclusion but using a different process. Instead, they add their local interface STP/RSTP cost to the root cost listed in each received Hello BPDU. The STP/RSTP port cost is simply an integer value assigned to each interface, per VLAN, for the purpose of providing an objective measurement that allows STP/RSTP to choose which interfaces to add to the STP/RSTP topology. The switches also look at their neighbor's root cost, as announced in Hello BPDUs received from each neighbor.

Figure 9-6 shows an example of how switches calculate their best root cost and then choose their root port, using the same topology and STP/RSTP costs as shown in Figure 9-5. STP/RSTP on SW3 calculates its cost to reach the root over the two possible paths by adding the advertised cost (in Hello messages) to the interface costs listed in the figure.

Key Topic

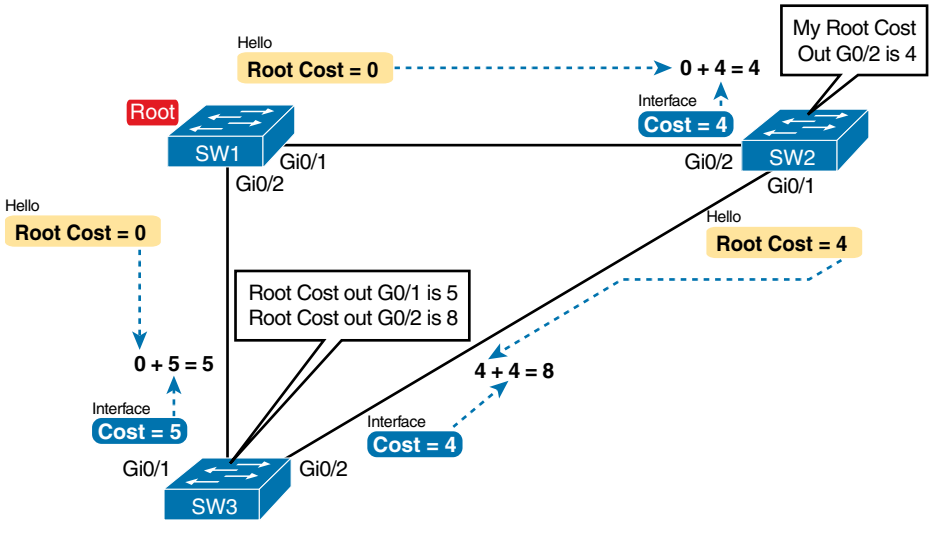


Figure 9-6 *How STP/RSTP Actually Calculates the Cost from SW3 to the Root*

Focus on the process for a moment. The root switch sends Hellos, with a listed root cost of 0. The idea is that the root's cost to reach itself is 0.

Next, look on the left of the figure. SW3 takes the received cost (0) from the Hello sent by SW1 and adds the interface cost (5) of the interface on which that Hello was received. SW3 calculates that the cost to reach the root switch, out that port (G0/1), is 5.

On the right side, SW2 has realized its best cost to reach the root is cost 4. So, when SW2 forwards the Hello toward SW3, SW2 lists a root cost 4. SW3's STP/RSTP port cost on port G0/2 is 4, so SW3 determines a total cost to reach root out its G0/2 port of 8.

As a result of the process depicted in Figure 9-6, SW3 chooses Gi0/1 as its RP because the cost to reach the root switch through that port (5) is lower than the other alternative (Gi0/2, cost 8). Similarly, SW2 chooses Gi0/2 as its RP, with a cost of 4 (SW1's advertised cost of 0 plus SW2's Gi0/2 interface cost of 4). Each switch places its root port into a forwarding state.

Switches need a tiebreaker to use in case the best root cost ties for two or more paths. If a tie occurs, the switch applies these three tiebreakers to the paths that tie, in order, as follows:

1. Choose based on the lowest neighbor bridge ID.
2. Choose based on the lowest neighbor port priority.
3. Choose based on the lowest neighbor internal port number.

Choosing the Designated Port on Each LAN Segment

STP/RSTP's final step to choose the STP/RSTP topology is to choose the designated port on each LAN segment. The designated port (DP) on each LAN segment is the switch port that advertises the lowest-cost Hello onto a LAN segment. When a nonroot switch forwards a Hello, the nonroot switch sets the root cost field in the Hello to that switch's cost to reach the root. In effect, the switch with the lower cost to reach the root, among all switches connected to a segment, becomes the DP on that segment.

For example, earlier Figure 9-4 shows in bold text the parts of the Hello messages from both SW2 and SW3 that determine the choice of DP on that segment. Note that both SW2 and SW3 list their respective cost to reach the root switch (cost 4 on SW2 and cost 5 on SW3). SW2 lists the lower cost, so SW2's Gi0/1 port is the designated port on that LAN segment.

All DPs are placed into a forwarding state; so in this case, SW2's Gi0/1 interface will be in a forwarding state.

If the advertised costs tie, the switches break the tie by choosing the switch with the lower BID. In this case, SW2 would also have won, with a BID of 32769:0200.0002.0002 versus SW3's 32769:0200.0003.0003.

NOTE Two additional tiebreakers are needed in some cases, although these would be unlikely today. A single switch can connect two or more interfaces to the same collision domain by connecting to a hub. In that case, the one switch hears its own BPDUs. So, if a switch ties with itself, two additional tiebreakers are used: the lowest interface STP/RSTP priority and, if that ties, the lowest internal interface number.

The only interface that does not have a reason to be in a forwarding state on the three switches in the examples shown in Figures 9-3 through 9-6 is SW3's Gi0/2 port. So, the STP/RSTP process is now complete. Table 9-5 outlines the state of each port and shows why it is in that state.

Table 9-5 State of Each Interface

Switch Interface	State	Reason Why the Interface Is in Forwarding State
SW1, Gi0/1	Forwarding	The interface is on the root switch, so it becomes the DP on that link.
SW1, Gi0/2	Forwarding	The interface is on the root switch, so it becomes the DP on that link.
SW2, Gi0/2	Forwarding	The root port of SW2.
SW2, Gi0/1	Forwarding	The designated port on the LAN segment to SW3.
SW3, Gi0/1	Forwarding	The root port of SW3.
SW3, Gi0/2	Blocking	Not the root port and not the designated port.

Note that the examples in this section focus on the links between the switches, but switch ports connected to endpoint devices should become DPs and settle into a forwarding state. Working through the logic, each switch will forward BPDUs on each port as part of the process to determine the DP on that LAN. Endpoints should ignore those messages because they do not run STP/RSTP, so the switch will win and become DP on every access port.

Configuring to Influence the STP Topology

STP/RSTP works by default on Cisco switches, so all the settings needed by a switch have a useful default. Switches have a default BID, based on a default priority value and adding a universal MAC address that comes with the switch hardware. Additionally, switch interfaces have default STP/RSTP costs based on the current operating speed of the switch interfaces.

Network engineers often want to change the STP/RSTP settings to then change the choices STP/RSTP makes in a given LAN. Two main tools available to the engineer are to configure the bridge ID and to change STP/RSTP port costs.

First, to change the BID, the engineer can set the priority used by the switch, while continuing to use the universal MAC address as the final 48 bits of the BID. For instance, giving a switch the lowest priority value among all switches will cause that switch to win the root election.

Port costs also have default values, per port, per VLAN. You can configure these port costs, which will in turn impact many switch's calculations of the root cost. For instance, to favor one link, give the ports on that link a lower cost, or to avoid a link, give the ports a higher cost.

Of course, it helps to know the default cost values so you can then choose alternative values as needed. Table 9-6 lists the default port costs suggested by IEEE. IOS on Cisco switches has long used the default settings as defined as far back as the 1998 version of the IEEE 802.1D standard. The latest IEEE standard to suggest RSTP default costs (as of the

publication of this book), the 2018 publication of the 802.1Q standard, suggests values that are more useful when using links faster than 10 Gbps.

Key Topic

Table 9-6 Default Port Costs According to IEEE

Ethernet Speed	IEEE Cost: 1998 (and Before)	IEEE Cost: 2004 (and After)
10 Mbps	100	2,000,000
100 Mbps	19	200,000
1 Gbps	4	20,000
10 Gbps	2	2000
100 Gbps	N/A	200
1 Tbps	N/A	20

Of note in regards to these defaults, the cost defaults based on the operating speed of the link, not the maximum speed. That is, if a 10/100/1000 port runs at 10 Mbps for some reason, its default STP cost on a Cisco switch is 100, the default cost for an interface running at 10 Mbps. Also, if you prefer the defaults in the right-side column of Table 9-6, note that Cisco Catalyst switches can be configured to use those values as defaults with a single global configuration command on each switch (**spanning-tree pathcost method long**).

Details Specific to STP (and Not RSTP)

As promised in the introduction to this chapter, the first section showed features that apply to both STP and RSTP. This next heading acts as the turning point, with the next several pages being about STP only. The upcoming section titled "Rapid STP Concepts" then shows details specific to RSTP, in contrast to STP.

Once the engineer has finished all STP configuration, the STP topology should settle into a stable state and not change, at least until the network topology changes. This section examines the ongoing operation of STP while the network is stable, and then it covers how STP converges to a new topology when something changes.

Note that almost all the differences between STP and RSTP revolve around the activities of waiting for and reacting to changes in the topology. STP performed well for the era and circumstances in which it was created. The "rapid" in RSTP refers to the improvements to how fast RSTP could react when changes occur—so understanding how STP reacts will be useful to understand why RSTP reacts faster. These next few pages show the specifics of STP (and not RSTP) and how STP reacts to and manages convergence when changes happen in an Ethernet LAN.

STP Activity When the Network Remains Stable

An STP root switch sends a new Hello BPDU every 2 seconds by default. Each nonroot switch forwards the Hello on all DPs, but only after changing items listed in the Hello. (As a result, the Hello flows once over every working link in the LAN.)

When forwarding the Hello BPDU, each switch sets the root cost to that local switch's calculated root cost. The switch also sets the "sender's bridge ID" field to its own bridge ID. (The root's bridge ID field is not changed.)

Assuming a default Hello timer of 2 seconds on the root switch, each switch will forward the received (and changed) Hellos out all DPs so that all switches continue to receive Hellos every 2 seconds. The following steps summarize the steady-state operation when nothing is currently changing in the STP topology:

Key Topic

Step 1. The root creates and sends a Hello BPDU, with a root cost of 0, out all its working interfaces (those in a forwarding state).

Step 2. The nonroot switches receive the Hello on their root ports. After changing the Hello to list their own BID as the sender's BID and listing that switch's root cost, the switch forwards the Hello out all designated ports.

Step 3. Steps 1 and 2 repeat until something changes.

When a switch fails to receive a Hello, it knows a problem might be occurring in the network. Each switch relies on these periodically received Hellos from the root as a way to know that its path to the root is still working. When a switch ceases to receive the Hellos, or receives a Hello that lists different details, something has failed, so the switch reacts and starts the process of changing the spanning-tree topology.

STP Timers That Manage STP Convergence

For various reasons, the STP convergence process requires the use of three timers, listed in Table 9-7. Note that all switches use the timers as dictated by the root switch, which the root lists in its periodic Hello BPDU messages.

Key Topic

Table 9-7 STP Timers

Timer	Default Value	Description
Hello	2 seconds	The time period between Hellos created by the root.
MaxAge	10 times Hello	How long any switch should wait, after ceasing to hear Hellos, before trying to change the STP topology.
Forward delay	15 seconds	Delay that affects the process that occurs when an interface changes from blocking state to forwarding state. A port stays in an interim listening state, and then an interim learning state, for the number of seconds defined by the forward delay timer.

If a switch does not get an expected Hello BPDU within the Hello time, the switch continues as normal. However, if the Hellos do not show up again within MaxAge time, the switch reacts by taking steps to change the STP topology. With default settings, MaxAge is 20 seconds (10 times the default Hello timer of 2 seconds). So, a switch would go 20 seconds without hearing a Hello before reacting.

After MaxAge expires, the switch essentially makes all its STP choices again, based on any Hellos it receives from other switches. It reevaluates which switch should be the root switch. If the local switch is not the root, it chooses its RP. And it determines whether it is DP on each of its other links.

The best way to describe STP convergence is to show an example using the same familiar topology. Figure 9-7 shows the same familiar figure, with SW3's Gi0/2 in a blocking state, but SW1's Gi0/2 interface has just failed.

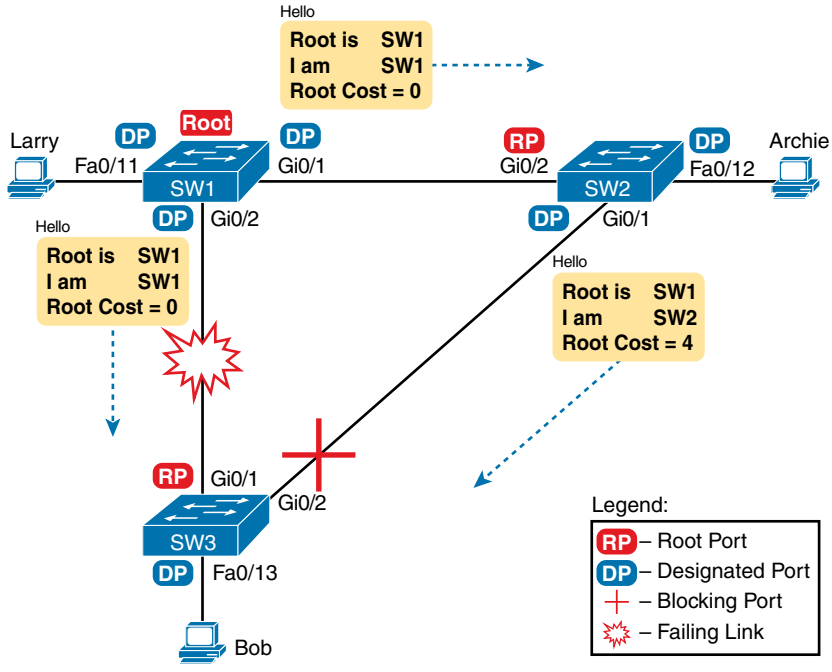


Figure 9-7 *Initial STP State Before SW1-SW3 Link Fails*

In the scenario shown in the figure, SW3 reacts to the change because SW3 fails to receive its expected Hellos on its Gi0/1 interface. However, SW2 does not need to react because SW2 continues to receive its periodic Hellos in its Gi0/2 interface. In this case, SW3 reacts either when MaxAge time passes without hearing the Hellos, or as soon as SW3 notices that interface Gi0/1 has failed. (If the interface fails, the switch can assume that the Hellos will not be arriving in that interface anymore.)

Now that SW3 can act, it begins by reevaluating the choice of root switch. SW3 still receives the Hellos from SW2, as forwarded from the root (SW1). SW1 still has a lower BID than SW3; otherwise, SW1 would not have already been the root. So, SW3 decides that SW1 wins the root election and that SW3 is not the root.

Next, SW3 reevaluates its choice of RP. At this point, SW3 is receiving Hellos on only one interface: Gi0/2. Whatever the calculated root cost, Gi0/2 becomes SW3's new RP. (The cost would be 8, assuming the STP costs had no changes since Figures 9-5 and 9-6.)

SW3 then reevaluates its role as DP on any other interfaces. In this example, no real work needs to be done. SW3 was already DP on interface Fa0/13, and it continues to be the DP because no other switches connect to that port.

Changing Interface States with STP

STP uses the idea of roles and states. *Roles*, like root port and designated port, relate to how STP analyzes the LAN topology. *States*, like forwarding and blocking, tell a switch whether to send or receive frames. When STP converges, a switch chooses new port roles, and the port roles determine the state (forwarding or blocking).

Switches using STP can simply move immediately from forwarding to blocking state, but they must take extra time to transition from blocking state to forwarding state. For instance, when switch SW3 in Figure 9-7 formerly used port G0/1 as its RP (a role), that port was in a forwarding state. After convergence, G0/1 might be neither an RP nor DP; the switch can immediately move that port to a blocking state.

However, when a port that formerly blocked needs to transition to forwarding, the switch first puts the port through two intermediate interface states. These temporary STP states help prevent temporary loops:

- **Listening:** Like the blocking state, the interface does not forward frames. The switch removes old stale (unused) MAC table entries for which no frames are received from each MAC address during this period. These stale MAC table entries could be the cause of the temporary loops.
- **Learning:** Interfaces in this state still do not forward frames, but the switch begins to learn the MAC addresses of frames received on the interface.

STP moves an interface from blocking to listening, then to learning, and then to forwarding state. STP leaves the interface in each interim state for a time equal to the forward delay timer, which defaults to 15 seconds. As a result, a convergence event that causes an interface to change from blocking to forwarding requires 30 seconds to transition from blocking to forwarding. In addition, a switch might have to wait MaxAge seconds (default 20 seconds) before even choosing to move an interface from blocking to forwarding state.

For example, follow what happens with an initial STP topology as shown in Figures 9-3 through 9-6, with the SW1-to-SW3 link failing as shown in Figure 9-7. If SW1 simply quit sending Hello messages to SW3, but the link between the two did not fail, SW3 would wait MaxAge seconds before reacting (20 seconds is the default). SW3 would actually quickly choose its ports' STP roles, but then wait 15 seconds each in listening and learning states on interface Gi0/2, resulting in a 50-second convergence delay.

Table 9-8 summarizes spanning tree's various interface states for easier review.

Key Topic

Table 9-8 IEEE STP (Not RSTP) States

State	Forwards Data Frames?	Learns MACs Based on Received Frames?	Transitory or Stable State?
Blocking	No	No	Stable
Listening	No	No	Transitory
Learning	No	Yes	Transitory
Forwarding	Yes	Yes	Stable
Disabled	No	No	Stable

Rapid STP Concepts

The original STP worked well given the assumptions about networks and networking devices in that era. However, as with any computing or networking standard, as time passes, hardware and software capabilities improve, so new protocols emerge to take advantage of those new capabilities. For STP, one of the most significant improvements over time has been the introduction of Rapid Spanning Tree Protocol (RSTP), introduced as standard IEEE 802.1w.

NOTE Just to make sure you are clear about the terminology: Throughout the rest of the chapter, *STP* refers to the original STP standard only, and use of the term *RSTP* does not include STP.

Before getting into the details of RSTP, it helps to make sense of the standards numbers a bit. 802.1w was actually an amendment to the 802.1D standard. The IEEE first published 802.1D in 1990, and anew in 1998. After the 1998 version of 802.1D, the IEEE published the 802.1w amendment to 802.1D in 2001, which first standardized RSTP.

Over the years, other meaningful changes happened in the standards as well, although those changes probably do not impact most networkers' thinking when it comes to working with STP or RSTP. But to be complete, the IEEE replaced STP with RSTP in the revised 802.1D standard in 2004. In another move, in 2011 the IEEE moved all the RSTP details into a revised 802.1Q standard. As it stands today, RSTP actually sits in the 802.1Q standards document.

As a result, when reading about RSTP, you will see documents, books, videos, and the like that refer to RSTP and include various references to 802.1w, 802.1D, and 802.1Q—and they might all be correct based on timing and context. At the same time, many people refer to RSTP as 802.1w because that was the first IEEE document to define it. However, for the purposes of this book, focus instead on the RSTP acronym rather than the IEEE standards numbers used with RSTP over its history.

NOTE The IEEE sells its standards, but through the "Get IEEE 802" program, you can get free PDFs of the current 802 standards. To read about RSTP today, you will need to download the 802.1Q standard, and then look for the sections about RSTP.

Now on to the details about RSTP in this chapter. As discussed throughout this chapter, RSTP and STP have many similarities, so this section next compares and contrasts the two. Following that, the rest of this section discusses the concepts unique to RSTP that are not found in STP—alternate root ports, different port states, backup ports, and the port roles used by RSTP.

Comparing STP and RSTP

RSTP works just like STP in several ways, as discussed in the first major section of the chapter. To review:

Key Topic

- RSTP and STP elect the root switch using the same rules and tiebreakers.
- RSTP and STP switches select their root ports with the same rules.
- RSTP and STP elect designated ports on each LAN segment with the same rules and tiebreakers.
- RSTP and STP place each port in either forwarding or blocking state, although RSTP calls the blocking state the *discarding* state.

In fact, RSTP works so much like STP that they can both be used in the same network. RSTP and STP switches can be deployed in the same network, with RSTP features working in switches that support it and traditional STP features working in the switches that support only STP.

With all these similarities, you might be wondering why the IEEE bothered to create RSTP in the first place. The overriding reason is convergence. STP takes a relatively long time to converge (50 seconds with the default settings when all the wait times must be followed). RSTP improves network convergence when topology changes occur, usually converging within a few seconds (or in slow conditions, in about 10 seconds).

RSTP changes and adds to STP in ways that avoid waiting on STP timers, resulting in quick transitions from forwarding to discarding (blocking) state and vice versa. Specifically, RSTP, compared to STP, defines more cases in which the switch can avoid waiting for a timer to expire, such as the following:

Key Topic

- RSTP adds a mechanism by which a switch can replace its root port, without any waiting to reach a forwarding state (in some conditions).
- RSTP adds a new mechanism to replace a designated port, without any waiting to reach a forwarding state (in some conditions).
- RSTP lowers waiting times for cases in which RSTP must wait for a timer.

For instance, imagine a failure case in which a link remains up, but for some reason, a non-root switch stops hearing the Hello BPDUs it had been hearing in the past. STP requires a switch to wait for MaxAge seconds, which STP defines based on 10 times the Hello timer, or 20 seconds, by default. RSTP shortens this timer, defining MaxAge as three times the Hello timer. Additionally, RSTP can send messages to the neighboring switch to inquire whether a problem has occurred rather than wait for timers.

The best way to get a sense for these mechanisms is to see how the RSTP alternate port and the backup port both work. RSTP uses the term *alternate port* to refer to a switch's other

ports that could be used as the root port if the root port ever fails. The *backup port* concept provides a backup port on the local switch for a designated port. (Note that backup ports apply only to designs that use hubs, so they are unlikely to be useful today.) However, both are instructive about how RSTP works. Table 9-9 lists these RSTP port roles.

Key Topic

Table 9-9 Port Roles in RSTP

Function	Port Role
Port that begins a nonroot switch's best path to the root	Root port
Port that replaces the root port when the root port fails	Alternate port
Switch port designated to forward onto a collision domain	Designated port
Port that replaces a designated port when a designated port fails	Backup port
Port that is administratively disabled	Disabled port

RSTP differs from STP in a few other ways as well. For instance, with STP, the root switch creates a Hello with all other switches, updating and forwarding the Hello. With RSTP, each switch independently generates its own Hellos. Additionally, RSTP allows for queries between neighbors, rather than waiting on timers to expire, as a means to avoid waiting to learn information. These types of protocol changes help RSTP-based switches isolate what has changed in a network and react quickly to choose a net RSTP topology.

The next few pages work through some of those overt RSTP features that differ from STP.

RSTP and the Alternate (Root) Port Role

With STP, each nonroot switch places one port in the STP root port (RP) role. RSTP follows that same convention, with the same exact rules for choosing the RP. RSTP then takes another step beyond STP, naming other possible RPs, identifying them as *alternate ports*.

To be an alternate port, both the RP and the alternate port must receive Hellos that identify the same root switch. For instance, in Figure 9-8, SW1 is the root. SW3 will receive Hello BPDUs on two ports: G0/1 and G0/2. Both Hellos list SW1's bridge ID (BID) as the root switch, so whichever port is not the root port meets the criteria to be an alternate port. SW3 picks G0/1 as its root port in this case and then makes G0/2 an alternate port.

An alternate port basically works like the second-best option for the root port. The alternate port can take over for the former root port, often very rapidly, without requiring a wait in other interim RSTP states. For instance, when the root port fails, or when Hellos stop arriving on the original root port, the switch changes the former root port's role and state: (a) the role from root port to a disabled port, and (b) the state from forwarding to discarding (the equivalent of STP's blocking state). Then, without waiting on any timers, the switch changes roles and state for the alternate port: its role changes to be the root port, with a forwarding state.

Notably, the new root port also does not need to spend time in other states, such as learning state, instead moving immediately to forwarding state.

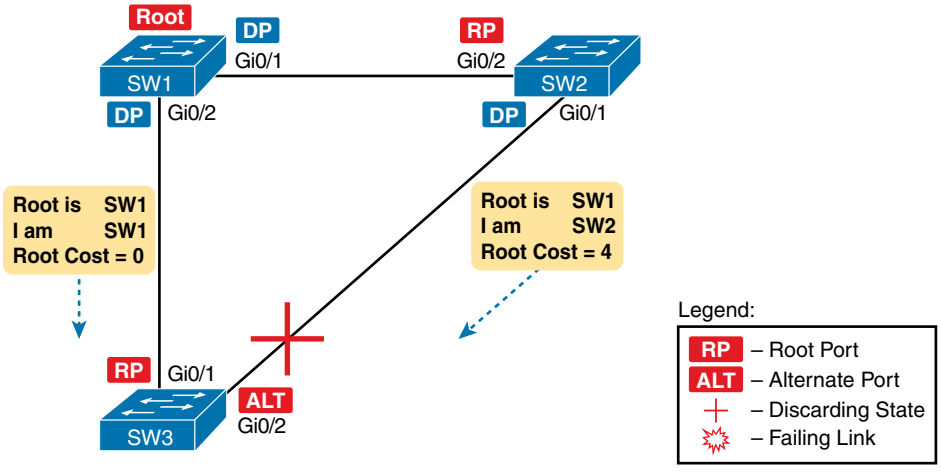


Figure 9-8 *Example of SW3 Making G0/2 Become an Alternate Port*

Figure 9-9 shows an example of RSTP convergence. SW3's root port before the failure shown in this figure is SW3's G0/1, the link connected directly to SW1 (the root switch). Then SW3's link to SW1 fails as shown in Step 1 of the figure.

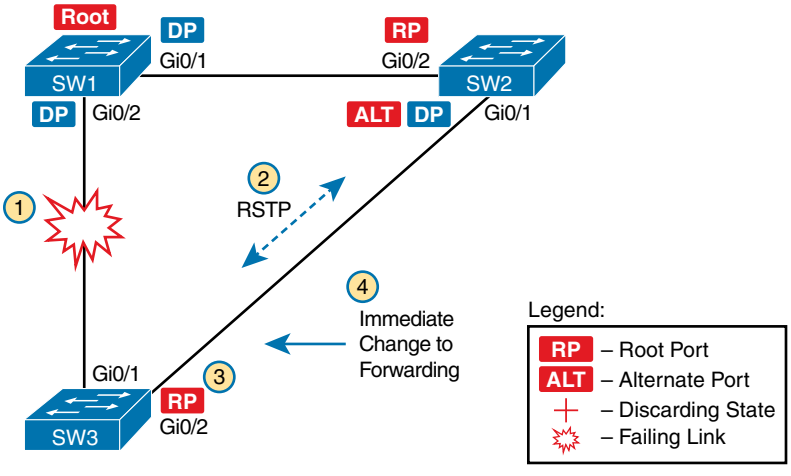


Figure 9-9 *Convergence Events with SW3 G0/1 Failure*

Following the steps in Figure 9-9:

Step 1. The link between SW1 and SW3 fails, so SW3's current root port (Gi0/1) fails.

Step 2. SW3 and SW2 exchange RSTP messages to confirm that SW3 will now transition its former alternate port (Gi0/2) to be the root port. This action causes SW2 to flush the required MAC table entries.

Step 3. SW3 transitions Gi0/1 to the disabled role and Gi0/2 to the root port role.

Step 4. SW3 transitions Gi0/2 to a forwarding state immediately, without using learning state, because this is one case in which RSTP knows the transition will not create a loop.

As soon as SW3 realizes its Gi0/1 interface has failed, the process shown in the figure takes very little time. None of the processes rely on timers, so as soon as the work can be done, the convergence completes. (This particular convergence example takes about 1 second in a lab.)

RSTP States and Processes

The depth of the previous example does not point out all details of RSTP, of course; however, the example does show enough details to discuss RSTP states and internal processes.

Both STP and RSTP use *port states*, but with some differences. First, RSTP keeps both the learning and forwarding states as compared with STP, for the same purposes. However, RSTP does not even define a listening state, finding it unnecessary. Finally, RSTP renames the blocking state to the discarding state and redefines its use slightly.

RSTP uses the discarding state for what STP defines as two states: disabled state and blocking state. Blocking should be somewhat obvious by now: the interface can work physically, but STP/RSTP chooses to not forward traffic to avoid loops. STP's disabled state simply meant that the interface was administratively disabled. RSTP just combines those into a single discarding state. Table 9-10 shows the list of STP and RSTP states for comparison purposes.

Key Topic

Table 9-10 Port States Compared: STP and RSTP

Function	STP State	RSTP State
Port is administratively disabled	Disabled	Discarding
Stable state that ignores incoming data frames and is not used to forward data frames	Blocking	Discarding
Interim state without MAC learning and without forwarding	Listening	Not used
Interim state with MAC learning and without forwarding	Learning	Learning
Stable state that allows MAC learning and forwarding of data frames	Forwarding	Forwarding

RSTP also changes some processes and message content (compared to STP) to speed convergence. For example, STP waits for a time (forward delay) in both listening and learning states. The reason for this delay in STP is that, at the same time, the switches have all been told to time out their MAC table entries. When the topology changes, the existing MAC table entries may actually cause a loop. With STP, the switches all tell each other (with BPDU messages) that the topology has changed and to time out any MAC table entries using the forward delay timer. This removes the entries, which is good, but it causes the need to wait in both listening and learning state for forward delay time (default 15 seconds each).

RSTP, to converge more quickly, avoids relying on timers. RSTP switches tell each other (using messages) that the topology has changed. Those messages also direct neighboring switches to flush the contents of their MAC tables in a way that removes all the potentially loop-causing entries, without a wait. As a result, RSTP creates more scenarios in which a formerly discarding port can immediately transition to a forwarding state, without waiting, and without using the learning state, as shown in the example in Figure 9-9.

RSTP and the Backup (Designated) Port Role

The RSTP backup port role acts as yet another new RSTP port role as compared to STP. As a reminder, the RSTP alternate port role creates a way for RSTP to quickly replace a switch's root port. Similarly, the RSTP backup port role creates a way for RSTP to quickly replace a switch's designated port on some LAN.

The need for a backup port can be a bit confusing at first because the need for the backup port role only happens in designs that are a little unlikely today. The reason is that a design must use hubs, which then allows the possibility that one switch connects more than one port to the same collision domain.

Figure 9-10 shows an example. SW3 and SW4 both connect to the same hub. SW4's port F0/1 happens to win the election as designated port (DP). The other port on SW4 that connects to the same collision domain, F0/2, acts as a backup port.

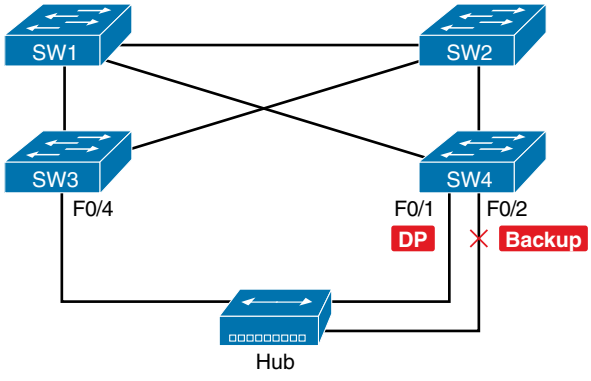


Figure 9-10 *RSTP Backup Port Example*

With a backup port, if the current designated port fails, SW4 can start using the backup port with rapid convergence. For instance, if SW4's F0/1 interface were to fail, SW4 could transition F0/2 to the designated port role, without any delay in moving from discarding state to a forwarding state.

RSTP Port Types

The final RSTP concept included here relates to some terms RSTP uses to refer to different types of ports and the links that connect to those ports.

To begin, consider the basic image in Figure 9-11. It shows several links between two switches. RSTP considers these links to be point-to-point links and the ports connected to them to be point-to-point ports because the link connects exactly two devices (points).

RSTP further classifies point-to-point ports into two categories. Point-to-point ports that connect two switches are not at the edge of the network and are simply called *point-to-point ports*. Ports that instead connect to a single endpoint device at the edge of the network, like a PC or server, are called *point-to-point edge ports*, or simply *edge ports*. In Figure 9-11, SW3's switch port connected to a PC is an edge port.

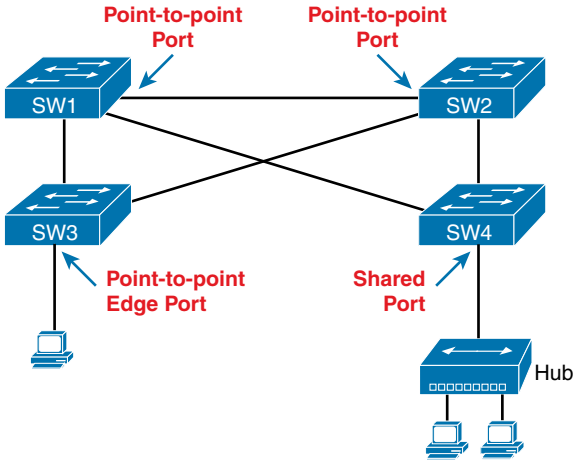


Figure 9-11 *RSTP Link Types*

Finally, RSTP defines the term *shared* to describe ports connected to a hub. The term *shared* comes from the fact that hubs create a shared Ethernet; hubs also force the attached switch port to use half-duplex logic. RSTP assumes that all half-duplex ports may be connected to hubs, treating ports that use half duplex as shared ports. RSTP converges more slowly on shared ports as compared to all point-to-point ports.

Optional STP Features

To close out the chapter, the last few topics introduce a few optional features that make STP work even better or be more secure: EtherChannel, PortFast, and BPDU Guard.

EtherChannel

One of the best ways to lower STP's convergence time is to avoid convergence altogether. EtherChannel provides a way to prevent STP convergence from being needed when only a single port or cable failure occurs.

EtherChannel combines multiple parallel segments of equal speed (up to eight) between the same pair of switches, bundled into an EtherChannel. The switches treat the EtherChannel as a single interface with regard to STP. As a result, if one of the links fails, but at least one of the links is up, STP convergence does not have to occur. For example, Figure 9-12 shows the familiar three-switch network, but now with two Gigabit Ethernet connections between each pair of switches.

With each pair of Ethernet links configured as an EtherChannel, STP treats each EtherChannel as a single link. In other words, both links to the same switch must fail for a switch to need to cause STP convergence. Without EtherChannel, if you have multiple parallel links between two switches, STP blocks all the links except one. With EtherChannel, all the parallel links can be up and working at the same time, while reducing the number of times STP must converge, which in turn makes the network more available.

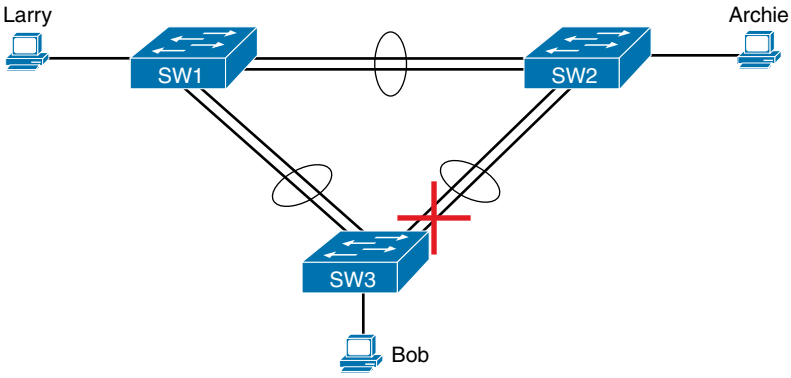


Figure 9-12 *Two-Segment EtherChannels Between Switches*

The current CCNA exam blueprint includes a topic for the configuration of both Layer 2 EtherChannels (as described here) as well as Layer 3 EtherChannels. Chapter 10, "RSTP and EtherChannel Configuration," shows how to configure Layer 2 EtherChannels, while Chapter 17, "IP Routing in the LAN," shows how to configure Layer 3 EtherChannels. Note that Layer 2 EtherChannels combine links that switches use as switch ports, with the switches using Layer 2 switching logic to forward and receive Ethernet frames over the EtherChannels. Layer 3 EtherChannels also combine links, but the switches use Layer 3 routing logic to forward packets over the EtherChannels.

PortFast

PortFast allows a switch to immediately transition from blocking to forwarding, bypassing listening and learning states. However, the only ports on which you can safely enable PortFast are ports on which you know that no bridges, switches, or other STP-speaking devices are connected. Otherwise, using PortFast risks creating loops, the very thing that the listening and learning states are intended to avoid.

PortFast is most appropriate for connections to end-user devices. If you turn on PortFast on ports connected to end-user devices, when an end-user PC boots, the switch port can move to an STP forwarding state and forward traffic as soon as the PC NIC is active. Without PortFast, each port must wait while the switch confirms that the port is a DP. With STP in particular (and not RSTP), the switch waits in the temporary listening and learning states before settling into the forwarding state.

As you might guess from the fact that PortFast speeds convergence, RSTP includes PortFast. You might recall the mention of RSTP port types, particularly point-to-point edge port types, around Figure 9-11. RSTP, by design of the protocol, converges quickly on these point-to-point edge type ports by bypassing the learning state, which is the same idea Cisco originally introduced with PortFast. In practice, Cisco switches enable RSTP point-to-point edge ports by enabling PortFast on the port.

BPDU Guard

STP and RSTP open up the LAN to several different types of possible security exposures. For example:

- An attacker could connect a switch to one of these ports, one with a low STP/RSTP priority value, and become the root switch. The new STP/RSTP topology could have worse performance than the desired topology.
- The attacker could plug into multiple ports, into multiple switches, become root, and actually forward much of the traffic in the LAN. Without the networking staff realizing it, the attacker could use a LAN analyzer to copy large numbers of data frames sent through the LAN.
- Users could innocently harm the LAN when they buy and connect an inexpensive consumer LAN switch (one that does not use STP/RSTP). Such a switch, without any STP/RSTP function, would not choose to block any ports and could cause a loop.

The *Cisco BPDU Guard* feature helps defeat these kinds of problems by disabling a port if any BPDUs are received on the port. So, this feature is particularly useful on ports that should be used only as an access port and never connected to another switch.

In addition, the BPDU Guard feature helps prevent problems with PortFast. PortFast should be enabled only on access ports that connect to user devices, not to other LAN switches. Using BPDU Guard on these same ports makes sense because if another switch connects to such a port, the local switch can disable the port before a loop is created.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 9-11 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 9-11 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Key Topic

Table 9-12 Key Topics for Chapter 9

Key Topic Element	Description	Page Number
Table 9-2	Lists the three main problems that occur when not using STP in a LAN with redundant links	215
Table 9-3	Lists the reasons why a switch chooses to place an interface into forwarding or blocking state	217
Table 9-4	Lists the most important fields in Hello BPDU messages	218
List	Logic for the root switch election	219
Figure 9-6	Shows how switches calculate their root cost	221
Table 9-6	Lists the original and current default STP port costs for various interface speeds	224
Step list	A summary description of steady-state STP operations	225
Table 9-7	STP timers	226
List	Definitions of what occurs in the listening and learning states	227
Table 9-8	Summary of 802.1D states	228
List	Key similarities between 802.1D STP and 802.1w RSTP	229
List	Methods RSTP uses to reduce convergence time	229
Table 9-9	List of 802.1w port roles	230
Table 9-10	Comparisons of port states with 802.1D and 802.1w	232

Key Terms You Should Know

blocking state, BPDU Guard, bridge ID, bridge protocol data unit (BPDU), designated port, EtherChannel, forward delay, forwarding state, Hello BPDU, learning state, listening state, MaxAge, PortFast, root port, root switch, root cost, Spanning Tree Protocol (STP), rapid STP (RSTP), alternate port, backup port, disabled port, discarding state

CHAPTER 10

RSTP and EtherChannel Configuration

This chapter covers the following exam topics:

2.0 Network Access

2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)

2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations

2.5.a Root port, root bridge (primary/secondary), and other port names

2.5.b Port states (forwarding/blocking)

2.5.c PortFast benefits

This chapter shows how to configure Rapid Spanning Tree Protocol (RSTP) and Layer 2 EtherChannels. The EtherChannel content, in the second major section of the chapter, follows a typical flow for most configuration/verification topics in a certification guide: it reviews concepts, shows configurations, and provides show commands that point out the configuration settings and operational state. The details include how to manually configure a channel, how to cause a switch to dynamically create a channel, and how EtherChannel load distribution works.

The first section of the chapter explores RSTP implementation taking a different approach. Cisco mentions RSTP concepts, but not configuration/verification, in the CCNA exam topics. However, to get a real sense of RSTP concepts, especially some concepts specific to Cisco Catalyst switches, you need to work with RSTP configuration and verification. The first section of the chapter explores RSTP implementation, but as a means to the end of more fully understanding RSTP concepts.

For those of you who, like me, probably would want to go ahead and practice configuring RSTP, do some show commands, and understand more fully, you do have some options:

- Read Appendix O, "Spanning Tree Protocol Implementation," from this book's companion website. The appendix is a chapter from the previous edition of this book, with full details of configuration/verification of STP and RSTP.
- Use the STP/RSTP config labs on my blog site (as regularly listed in the Chapter Review section of each chapter).

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 10-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Understanding RSTP Through Configuration	1–3
Implementing EtherChannel	4–6

1. Which type value on the **spanning-tree mode** *type* global command enables the use of RSTP?
   a. **rapid-pvst**
   b. **pvst**
   c. **rstp**
   d. **rpvst**

2. Examine the following output from the **show spanning-tree vlan 5** command, which describes a root switch in a LAN. Which answers accurately describe facts related to the root's bridge ID?

```
SW1# show spanning-tree vlan 5

VLAN0005
   Spanning tree enabled protocol rstp
   Root ID  Priority    32773
            Address     1833.9d7b.0e80
            Cost        15
            Port        25 (GigabitEthernet0/1)
            Hello Time  2 sec Max Age 20 sec Forward Delay 15 sec
```

   a. The system ID extension value, in decimal, is 5.
   b. The root's configured priority value is 32773.
   c. The root's configured priority value is 32768.
   d. The system ID extension value, in hexadecimal, is 1833.9d7b.0e80.

3. With the Cisco RPVST+, which of the following action(s) does a switch take to identify which VLAN is described by a BPDU? (Choose three answers.)
   a. Adds a VLAN tag when forwarding a BPDU on trunks
   b. Adds the VLAN ID in an extra TLV in the BPDU
   c. Lists the VLAN ID as the middle 12 bits of the System ID field of the BPDU
   d. Lists the VLAN ID in the System ID Extension field of the BPDU

4. An engineer configures a switch to put interfaces G0/1 and G0/2 into the same Layer 2 EtherChannel. Which of the following terms is used in the configuration commands?
   a. **EtherChannel**
   b. **PortChannel**
   c. **Ethernet-Channel**
   d. **Channel-group**
5. Which combinations of keywords on the **channel-group** interface subcommand on two neighboring switches will cause the switches to use LACP and attempt to add the link to the EtherChannel? (Choose two answers.)
   a. **desirable** and **active**
   b. **passive** and **active**
   c. **active** and **auto**
   d. **active** and **active**
6. A Cisco Catalyst switch needs to send frames over a Layer 2 EtherChannel. Which answer best describes how the switch balances the traffic over the four active links in the channel?
   a. Breaks each frame into fragments of approximately one-fourth of the original frame, sending one fragment over each link
   b. Sends the entire frame over one link, alternating links in sequence for each successive frame
   c. Sends the entire frame over one link, choosing the link by applying some math to fields in each frame's headers
   d. Sends the entire frame over one link, using the link with the lowest percent utilization as the next link to use

Foundation Topics

Understanding RSTP Through Configuration

Cisco IOS switches today typically default to using RSTP rather than STP, with default settings so that RSTP works with no configuration. You can buy some Cisco switches and connect them with Ethernet cables in a redundant topology, and RSTP will ensure that frames do not loop. And even if some switches use RSTP and some use STP, the switches can interoperate and still build a working spanning tree—and you never even have to think about changing any settings!

Although RSTP works without any configuration, most medium-size to large-size campus LANs benefit from some STP configuration. For instance, Figure 10-1 shows a typical LAN design model, with two distribution layer switches (D1 and D2). The design may have dozens of access layer switches that connect to end users; the figure shows just three access switches (A1, A2, and A3). For a variety of reasons, most network engineers make the distribution layer switches be the root.

Key Topic

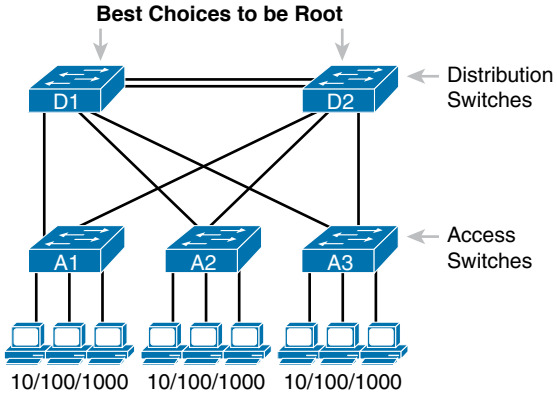


Figure 10-1 *Typical Configuration Choice: Making Distribution Switch Be Root*

NOTE Cisco uses the term *access switch* to refer to switches used to connect to endpoint devices. The term *distribution switch* refers to switches that do not connect to endpoints but rather connect to each access switch, providing a means to distribute frames throughout the LAN. If you want to read more about LAN design concepts and terms, refer to this book's companion website for Appendix K, "Analyzing Ethernet LAN Designs."

As discussed in the introduction to this chapter, this first section of the chapter examines a variety of STP/RSTP configuration topics, but with a goal of revealing a few more details about how STP and RSTP operate. Following this opening section about RSTP configuration, the next section examines how to configure Layer 2 EtherChannels, and how that impacts STP/RSTP.

The Need for Multiple Spanning Trees

The IEEE first standardized STP as the IEEE 802.1D standard, first published back in 1990. To put some perspective on that date, Cisco did not have a LAN switch product line at the time, and virtual LANs did not exist yet. Instead of multiple VLANs in a physical Ethernet LAN, the physical Ethernet LAN existed as one single broadcast domain, with one instance of STP.

By the mid 1990s, VLANs had appeared on the scene, along with LAN switches. The emergence of VLANs posed a challenge for STP—the only type of STP available at the time—because STP defined a single common spanning tree (CST) topology for the entire LAN. The IEEE needed an option to create multiple spanning trees so that traffic could be balanced across the available links, as shown in Figure 10-2. With two different STP instances, SW3 could block on a different interface in each VLAN, as shown in the figure.

Key Topic

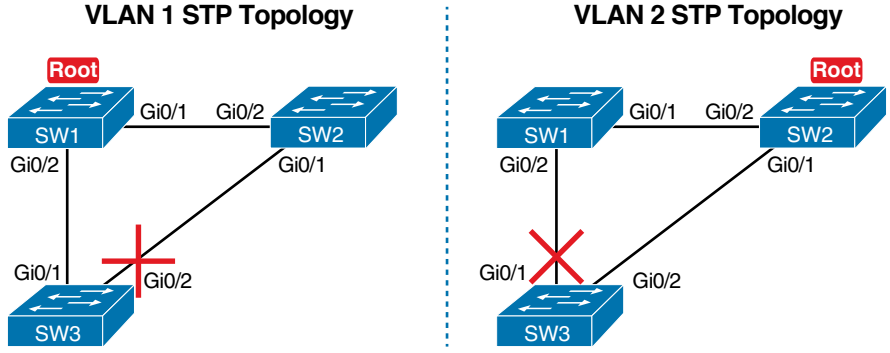


Figure 10-2 *Load Balancing with One Tree for VLAN 1 and Another for VLAN 2*

STP Modes and Standards

Because of the sequence of events over the history of the various STP family of protocols, vendors like Cisco needed to create their own proprietary features to create the per-VLAN spanning tree concept shown in Figure 10-2. That sequence resulted in the following:

- When STP was the only STP standard back in the 1990s with 802.1D, Cisco created the STP-based Per VLAN Spanning Tree Plus (PVST+) protocol, which creates one spanning tree instance per VLAN.
- When the IEEE introduced RSTP (in 802.1D amendment 802.1w, in the year 2001), Cisco also created the Rapid PVST+ (RPVST+) protocol. RPVST+ provided more features than standardized RSTP, including one tree per VLAN.
- The IEEE did not adopt Cisco's PVST+ or RPVST+ into their standards to create multiple spanning trees. Instead, the IEEE created a different method: Multiple Spanning Tree Protocol (MSTP), originally defined in 802.1Q amendment 802.1s.

Figure 10-3 shows the features as a timeline for perspective.

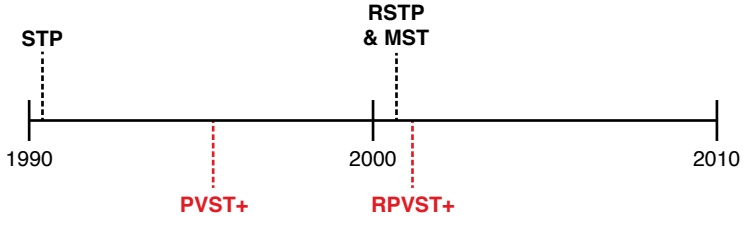


Figure 10-3 *Timeline of Per-VLAN and Multiple STP Features*

Today, Cisco Catalyst switches give us three options to configure on the **spanning-tree mode** command, which tells the switch which type of STP to use. Note that the switches do not support STP or RSTP with the single tree (CST). They can use either the Cisco-proprietary and STP-based PVST+, Cisco-proprietary and RSTP-based RPVST+, or the IEEE standard MSTP. Table 10-2 summarizes some of the facts about these standards and options,

---

Answers to the "Do I Know This Already?" quiz:

1 A **2** A, C **3** A, B, D **4** D **5** B, D **6** C

along with the keywords used on the **spanning-tree mode** global configuration command. Example 10-1, which follows, shows the command options in global configuration mode.

Key Topic

Table 10-2 STP Standards and Configuration Options

Name	Based on STP or RSTP?	# Trees	Original IEEE Standard	Config Parameter
STP	STP	1 (CST)	802.1D	N/A
PVST+	STP	1/VLAN	802.1D	**pvst**
RSTP	RSTP	1 (CST)	802.1w	N/A
Rapid PVST+	RSTP	1/VLAN	802.1w	**rapid-pvst**
MSTP	RSTP	1 or more*	802.1s	**mst**

* MSTP allows the definition of as many instances (multiple spanning tree instances, or MSTIs) as chosen by the network designer but does not require one per VLAN.

Example 10-1 *Cisco switch Spanning Tree modes*

```
SW1(config)# spanning-tree mode ?
  mst         Multiple spanning tree mode
  pvst        Per-Vlan spanning tree mode
  rapid-pvst  Per-Vlan rapid spanning tree mode
SW1(config)#
```

The Bridge ID and System ID Extension

To support the idea of multiple spanning trees, whether one per VLAN or simply multiple as created with MSTP, the protocols must consider the VLANs and VLAN trunking. (That's one reason why RSTP and MSTP now exist as part of the 802.1Q standard, which defines VLANs and VLAN trunking.) To help make that work, the IEEE redefined the format of the original BID value to help make per-VLAN instances of STP/RSTP become a reality.

Originally, a switch's BID was formed by combining the switch's 2-byte priority and its 6-byte MAC address. The revised rules divide the original priority field into two separate fields, as shown in Figure 10-4: a 4-bit priority field and a 12-bit subfield called the *system ID extension* (which represents the VLAN ID).

Key Topic

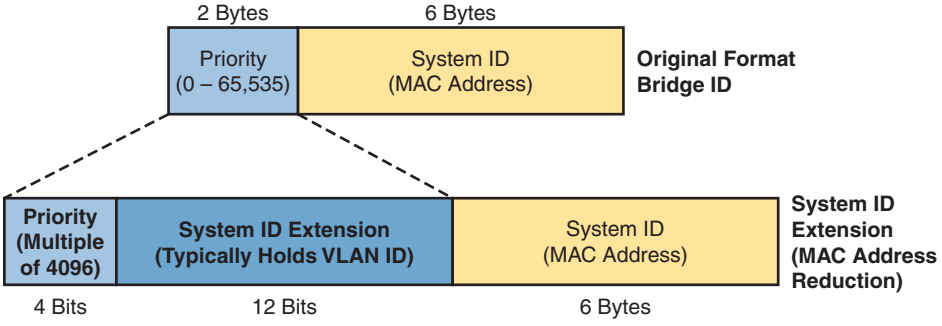


Figure 10-4 *STP System ID Extension*

10

Cisco switches let you configure the BID, but only the priority part. The switch fills in its universal (burned-in) MAC address as the system ID. It also plugs in the VLAN ID of a VLAN in the 12-bit system ID extension field; you cannot change that behavior either. The only part configurable by the network engineer is the 4-bit priority field.

However, configuring the number to put in the priority field may be one of the strangest things to configure on a Cisco router or switch. As shown at the top of Figure 10-4, the priority field was originally a 16-bit number, which represented a decimal number from 0 to 65,535. Because of that history, the configuration command (**spanning-tree vlan** *vlan-id* **priority** *x*) requires a decimal number between 0 and 65,535. But not just any number in that range will suffice; it must be a multiple of 4096, as emphasized in the help text shown in Example 10-2.

Example 10-2 *Help Shows Requirements for Using Increments of 4096 for Priority*

```
SW1(config)# spanning-tree vlan 1 priority ?
  <0-61440>  bridge priority in increments of 4096
SW1(config)#
```

Table 10-3 lists all the configurable values for the STP/RSTP priority. However, do not worry about memorizing the values. Instead, the table lists the values to emphasize two points about the binary values: the first 4 bits in each value differ, but the last 12 bits remain as 12 binary zeros.

Table 10-3 STP/RSTP Configurable Priority Values

Decimal Value	16-bit Binary Equivalent	Decimal Value	16-bit Binary Equivalent
0	**0000** 0000 0000 0000	32768	**1000** 0000 0000 0000
4096	**0001** 0000 0000 0000	36864	**1001** 0000 0000 0000
8192	**0010** 0000 0000 0000	40960	**1010** 0000 0000 0000
12288	**0011** 0000 0000 0000	45056	**1011** 0000 0000 0000
16384	**0100** 0000 0000 0000	49152	**1100** 0000 0000 0000
20480	**0101** 0000 0000 0000	53248	**1101** 0000 0000 0000
24576	**0110** 0000 0000 0000	57344	**1110** 0000 0000 0000
28672	**0111** 0000 0000 0000	61440	**1111** 0000 0000 0000

Note that while you can set the priority to any of the 16 decimal values in Table 10-3, Cisco provides a convenient means to create a primary and secondary root switch concept without configuring an actual number. In most LAN designs, only a small number of switches would be good candidates to ever be the root switch based on where the switches sit within the topology. Think of the preferred switch as the primary switch and the next-best option as the secondary switch. Then, to configure those two switches to be the two most likely switches to be the root switch, simply configure

spanning-tree vlan *x* **root primary** (on the switch that should be primary)

spanning-tree vlan *x* **root secondary** (on the switch that should be secondary)

These two commands cause the switch to make a choice of priority value but then store the chosen priority value in the **spanning-tree vlan** *x* **priority** *value* command. The command with **root primary** or **root secondary** does not appear in the configuration. When configuring **root primary**, the switch looks at the priority of the current root switch and chooses either (a) 24,576 or (b) 4096 less than the current root's priority (if the current root's priority is 24,576 or less) to the configuration instead. When configuring, **root secondary** always results in that switch using a priority of 28,672, with the assumption that the value will be less than other switches that use the default of 32,768, and higher than any switch configured as **root primary**.

How Switches Use the Priority and System ID Extension

Cisco Catalyst switches configure the priority value using a number that represents a 16-bit value; however, the system ID extension exists as the low-order 12 bits of that same number. This next topic works through connecting those ideas.

When the switch builds its BID to use for RSTP in a VLAN, it must combine the configured priority with the VLAN ID of that VLAN. Interestingly, the configured priority results in a 16-bit priority that always ends with 12 binary 0s. That fact makes the process of combining values to create the BID a little simpler for the switch and possibly a little simpler for network engineers once you understand it all.

First, consider the process shown in Figure 10-5. The top shows the configured priority value (decimal 32768), in 16-bit binary form, with a System ID Extension of 12 zeros. Moving down the figure, you see the binary version of a VLAN ID (decimal 9). At the last step, the switch replaces those last 12 bits of the System ID Extension with the value that matches the VLAN ID and uses that value as the first 16 bits of the BID.

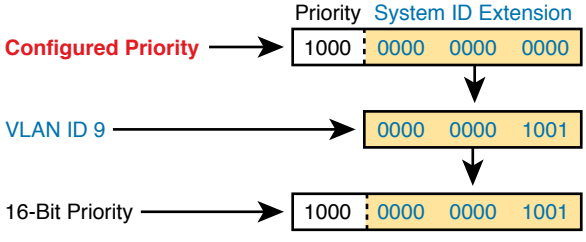


Figure 10-5 *Configured Priority (16-Bit) and System ID Extension (12-Bit) Added*

As it turns out, the process shown in Figure 10-5 is just the sum of the two numbers—both in binary and decimal. To see an example, refer to upcoming Example 10-3, which demonstrates the following details:

- The output shows details about VLAN 9.
- The root switch has been configured with the **spanning-tree vlan 9 priority 24576** command.
- The local switch (the switch on which the command was gathered) has been configured with the **spanning-tree vlan 9 priority 32768** command.

- Conveniently, the decimal equivalent of the two switches' first 16 bits—the original 16-bit priority field—can be easily calculated in decimal. In this example:
  - **Root Switch:** 24,576 (priority) + 9 (VLAN ID) = 24585
  - **Local Switch:** 32,768 (priority) + 9 (VLAN ID) = 32777

The output in Example 10-3 matches this logic. The top highlight shows the priority of the root switch (24585), which is the sum of the root switch's priority setting (configured as 24,576) plus 9 for the VLAN ID. The second highlight shows a value of 32,777, calculated as the local switch's priority setting of 32,768 plus 9 for the VLAN ID.

Example 10-3 *Examining the 16-bit Priority as Interpreted in Cisco* **show** *Commands*

```
SW1# show spanning-tree vlan 9

VLAN0009
  Spanning tree enabled protocol rstp
  Root ID     Priority    24585
              Address     1833.9d7b.0e80
              Cost        4
              Port        25 (GigabitEthernet0/1)
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID    Priority    32777 (priority 32768 sys-id-ext 9)
              Address     f47f.35cb.d780
! Output omitted for brevity
```

RSTP Methods to Support Multiple Spanning Trees

Although the history and configuration might make the BID priority idea seem a bit convoluted, having an extra 12-bit field in the BID works well in practice because it can be used to identify the VLAN ID. VLAN IDs range from 1 to 4094, requiring 12 bits.

For the purposes of discussion, focus on the standard RSTP and its Cisco-proprietary cousin RPVST+. Both use the RSTP mechanisms as discussed in Chapter 9, "Spanning Tree Protocol Concepts," but RPVST+ uses the mechanisms for every VLAN, while standard RSTP does not. So how do their methods differ?

- RSTP creates one tree—the Common Spanning Tree (CST)—while RPVST+ creates one tree for each and every VLAN.
- RSTP sends one set of RSTP messages (BPDUs) in the network, no matter the number of VLANs, while RPVST+ sends one set of messages per VLAN.
- RSTP and RPVST+ use different destination MAC addresses: RSTP with multicast address 0180.C200.0000 (an address defined in the IEEE standard), and RPVST+ with multicast address 0100.0CCC.CCCD (an address chosen by Cisco).
- When transmitting messages on VLAN trunks, RSTP sends the messages in the native VLAN with no VLAN header/tag. RPVST+ sends each VLAN's messages inside that VLAN—for instance, BPDUs about VLAN 9 have an 802.1Q header that lists VLAN 9.

- RPVST+ adds an extra type-length value (TLV) to the BPDU that identifies the VLAN ID, while RSTP does not (because it does not need to, as RSTP ignores VLANs.)
- Both view the 16-bit priority as having a 12-bit System ID Extension, with RSTP setting the value to 0000.0000.0000, meaning "no VLAN," while RPVST+ uses the VLAN ID.

In other words, standard RSTP behaves as if VLANs do not exist, while Cisco's RPVST+ integrates VLAN information into the entire process.

NOTE Some documents refer to the feature of sending BPDUs over trunks with VLAN tags matching the same VLAN as BPDU tunneling.

Other RSTP Configuration Options

This chapter does not attempt to work through all the configuration options available for RSTP. However, many of the configuration settings may be intuitive now that you know quite a bit about the protocol. This final topic in the first section of the chapter summarizes a few of the configuration concepts. As a reminder, for those interested in continuing on to CCNP Enterprise, you might be interested in reading more about RSTP configuration in the companion website's Appendix O, "Spanning Tree Protocol Implementation."

- **Switch Priority:** The global command **spanning-tree vlan** *x* **priority** *y* lets an engineer set the switch's priority in that VLAN.
- **Primary and Secondary Root Switches:** The global command **spanning-tree vlan** *x* **root primary | secondary** also lets you set the priority, but the switch decides on a value to make that switch likely to be the primary root switch (the root) or the secondary root switch (the switch that becomes root if the primary fails).
- **Port Costs:** The interface subcommand **spanning-tree** [**vlan** *x*] **cost** *y* lets an engineer set the switch's STP/RSTP cost on that port, either for all VLANs or for a specific VLAN on that port. Changing those costs then changes the root cost for some switches, which impacts the choice of root ports and designated ports.

That concludes this chapter's examination of RSTP configuration—now on to Layer 2 EtherChannel!

Configuring Layer 2 EtherChannel

As introduced in Chapter 9, two neighboring switches can treat multiple parallel links between each other as a single logical link called an *EtherChannel*. Without EtherChannel, a switch treats each physical port as an independent port, applying MAC learning, forwarding, and STP logic per physical port. With EtherChannel, the switch applies all those same processes to a group of physical ports as one entity: the EtherChannel. Without EtherChannel, with parallel links between two switches, STP/RSTP would block all links except one, but with EtherChannel, the switch can use all the links, load balancing the traffic over the links.

NOTE All references to EtherChannel in this chapter refer to Layer 2 EtherChannels, not to Layer 3 EtherChannels (as discussed in Chapter 17, "IP Routing in the LAN"). CCNA 200-301 exam topics include both Layer 2 and Layer 3 EtherChannels.

EtherChannel may be one of the most challenging switch features to make work. First, the configuration has several options, so you have to remember the details of which options work together. Second, the switches also require a variety of other interface settings to match among all the links in the channel, so you have to know those settings as well.

This section shows how to configure a Layer 2 EtherChannel, first through manual (static) configuration, and then by allowing dynamic protocols to create the channel. This section closes with some information about some common configuration issues that occur with Layer 2 EtherChannels.

Configuring a Manual Layer 2 EtherChannel

To configure a Layer 2 EtherChannel so that all the ports always attempt to be part of the channel, simply add the correct **channel-group** configuration command to each physical interface, on each switch, all with the **on** keyword, and all with the same number. The **on** keyword tells the switches to place a physical interface into an EtherChannel, and the number identifies the PortChannel interface number that the interface should be a part of.

Before getting into the configuration and verification, however, you need to start using three terms as synonyms: *EtherChannel*, *PortChannel*, and *Channel-group*. Oddly, IOS uses the **channel-group** configuration command, but then to display its status, IOS uses the **show etherchannel** command. Then the output of this **show** command refers to neither an "EtherChannel" nor a "Channel-group," instead using the term "PortChannel." So, pay close attention to these three terms in the example.

To configure an EtherChannel manually, follow these steps:

Key Topic

Config Checklist

Step 1. Add the **channel-group** *number* **mode on** command in interface configuration mode under each physical interface that should be in the channel to add it to the channel.

Step 2. Use the same number for all commands on the same switch, but the channel-group number on the neighboring switch can differ.

Example 10-4 shows a simple example, with two links between switches SW1 and SW2, as shown in Figure 10-6. The configuration shows SW1's two interfaces placed into channel-group 1, with two **show** commands to follow.

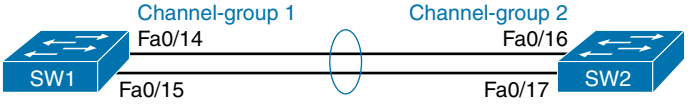


Figure 10-6 *Sample LAN Used in EtherChannel Example*

Example 10-4 *Configuring and Monitoring EtherChannel*

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# interface fa 0/14
SW1(config-if)# channel-group 1 mode on
SW1(config)# interface fa 0/15
```

```
SW1(config-if)# channel-group 1 mode on
SW1(config-if)# ^Z

SW1# show spanning-tree vlan 3

VLAN0003
 Spanning tree enabled protocol ieee
 Root ID     Priority    28675
             Address     0019.e859.5380
             Cost        12
             Port        72 (Port-channel1)
             Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID   Priority    28675 (priority 28672 sys-id-ext 3)
             Address     0019.e86a.6f80
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time  300

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Po1                 Root FWD 12        128.64   P2p Peer(STP)

SW1# show etherchannel 1 summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      N - not in use, no aggregation
        f - failed to allocate aggregator

        M - not in use, minimum links not met
        m - not in use, port not aggregated due to minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators: 1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)           -        Fa0/14(P) Fa0/15(P)
```

Take a few moments to look at the output in the two **show** commands in the example, as well. First, the **show spanning-tree** command lists Po1, short for PortChannel1, as

an interface. This interface exists because of the **channel-group** commands using the **1** parameter. STP no longer operates on physical interfaces Fa0/14 and Fa0/15, instead operating on the PortChannel1 interface, so only that interface is listed in the output.

Next, note the output of the **show etherchannel 1 summary** command. It lists as a heading "Port-channel," with Po1 below it. It also lists both Fa0/14 and Fa0/15 in the list of ports, with a (P) beside each. Per the legend, the *P* means that the ports are bundled in the port channel, which is a code that means these ports have passed all the configuration checks and are valid to be included in the channel.

Configuring Dynamic EtherChannels

In addition to manual configuration, Cisco switches also support two different configuration options that then use a dynamic protocol to negotiate whether a particular link becomes part of an EtherChannel or not. Basically, the configuration enables a protocol for a particular channel-group number. At that point, the switch can use the protocol to send messages to/from the neighboring switch and discover whether their configuration settings pass all checks. If a given physical link passes, the link is added to the EtherChannel and used; if not, it is placed in a down state, and not used, until the configuration inconsistency can be resolved.

Most Cisco Catalyst switches support the Cisco-proprietary Port Aggregation Protocol (PAgP) and the IEEE standard Link Aggregation Control Protocol (LACP), based on IEEE standard 802.3ad. Although differences exist between the two, to the depth discussed here, they both accomplish the same task: negotiate so that only links that pass the configuration checks are actually used in an EtherChannel.

One difference of note is that LACP does support more links in a channel—16—as compared to PAgP's maximum of 8. With LACP, only 8 can be active at one time, with the others waiting to be used should any of the other links fail.

To configure either protocol, a switch uses the **channel-group** configuration commands on each switch, but with a keyword that either means "use this protocol and begin negotiations" or "use this protocol and wait for the other switch to begin negotiations." As shown in Figure 10-7, the **desirable** and **auto** keywords enable PAgP, and the **active** and **passive** keywords enable LACP. With these options, at least one side has to begin the negotiations. In other words, with PAgP, at least one of the two sides must use **desirable**, and with LACP, at least one of the two sides must use **active**.

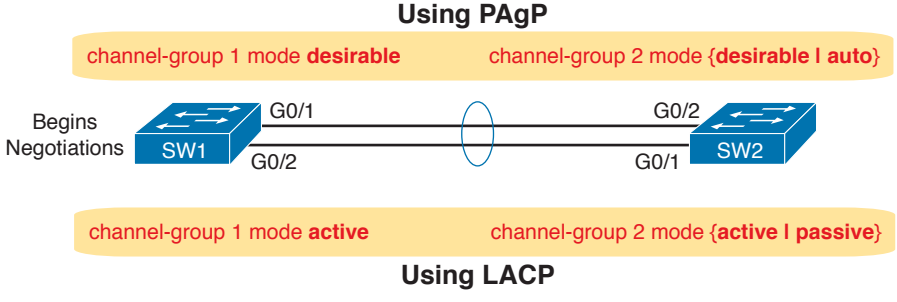


Figure 10-7 *Correct EtherChannel Configuration Combinations*

NOTE Do not use the **on** parameter on one end, and either **auto** or **desirable** (or for LACP, **active** or **passive**) on the neighboring switch. The **on** option uses neither PAgP nor LACP, so a configuration that uses **on**, with PAgP or LACP options on the other end, would prevent the EtherChannel from working.

For example, in the design shown in Figure 10-7, imagine both physical interfaces on both switches were configured with the **channel-group 1 mode desirable** interface subcommand. As a result, the two switches would negotiate and create an EtherChannel. Example 10-5 shows the verification of that configuration, with the command **show etherchannel 1 port-channel**. This command confirms the protocol in use (PAgP, because the **desirable** keyword was configured), and the list of interfaces in the channel.

Example 10-5 *EtherChannel Verification: PAgP Desirable Mode*

```
SW1# show etherchannel 1 port-channel
                Port-channels in the group:
                ---------------------------

Port-channel: Po1
------------
Age of the Port-channel   = 0d:00h:04m:04s
Logical slot/port   = 16/1          Number of ports = 2
GC                  = 0x00020001      HotStandBy port = null
Port state          = Port-channel Ag-Inuse
Protocol            = PAgP
Port security       = Disabled
Load share deferral = Disabled

Ports in the Port-channel:

Index   Load   Port     EC state        No of bits
------+------+------+------------------+-----------
  0     00     Gi0/1    Desirable-Sl       0
  0     00     Gi0/2    Desirable-Sl       0

Time since last port bundled: 0d:00h:03m:57s Gi0/2
```

Physical Interface Configuration and EtherChannels

Even when the **channel-group** commands have all been configured correctly, other configuration settings can prevent a switch from using a physical port in an EtherChannel—even physical ports manually configured to be part of the channel. The next topic examines those reasons.

First, before using a physical port in an EtherChannel, the switch compares the new physical port's configuration to the existing ports in the channel. That new physical interface's settings must be the same as the existing ports' settings; otherwise, the switch does not add the new link to the list of approved and working interfaces in the channel. That is, the

physical interface remains configured as part of the PortChannel, but it is not used as part of the channel, often being placed into some nonworking state.

The list of items the switch checks includes the following:

- Speed
- Duplex
- Operational access or trunking state (all must be access, or all must be trunks)
- If an access port, the access VLAN
- If a trunk port, the allowed VLAN list (per the **switchport trunk allowed** command)
- If a trunk port, the native VLAN
- STP interface settings

In addition, switches check the settings on the neighboring switch. To do so, the switches either use PAgP or LACP (if already in use) or use Cisco Discovery Protocol (CDP) if using manual configuration. When checking neighbors, all settings except the STP settings must match.

As an example, SW1 and SW2 again use two links in one EtherChannel from Figure 10-7. Before configuring the EtherChannel, SW1's G0/2 was given a different RSTP port cost than G0/1. Example 10-6 picks up the story just after configuring the correct **channel-group** commands, when the switch is deciding whether to use G0/1 and G0/2 in this EtherChannel.

Example 10-6 *Local Interfaces Fail in EtherChannel Because of Mismatched STP Cost*

```
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
 Po1, putting Gi0/1 in err-disable state
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
 Po1, putting Gi0/2 in err-disable state
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on Po1,
putting Po1 in err-disable state
*Mar 1 23:18:58.120: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to
  down
*Mar 1 23:18:58.137: %LINK-3-UPDOWN: Interface Port-channel1, changed state to down
*Mar 1 23:18:58.137: %LINK-3-UPDOWN: Interface GigabitEthernet0/2, changed state to
  down

SW1# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      N - not in use, no aggregation
        f - failed to allocate aggregator

        M - not in use, minimum links not met
        m - not in use, port not aggregated due to minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
```

```
        d - default port

        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators: 1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SD)           -        Gi0/1(D) Gi0/2(D)
```

The messages at the top of the example specifically state what the switch does when determining whether the interface settings match. In this case, SW1 detects the different STP costs. SW1 does not use G0/1, does not use G0/2, and even places them into an err-disabled state. The switch also puts the PortChannel into err-disabled state. As a result, the PortChannel is not operational, and the physical interfaces are also not operational.

To solve this problem, you must reconfigure the physical interfaces to use the same STP settings. In addition, the PortChannel and physical interfaces must be **shutdown**, and then **no shutdown**, to recover from the err-disabled state. (Note that when a switch applies the **shutdown** and **no shutdown** commands to a PortChannel, it applies those same commands to the physical interfaces, as well; so, just do the **shutdown/no shutdown** on the PortChannel interface.)

EtherChannel Load Distribution

When using Layer 2 EtherChannels, a switch's MAC learning process associates MAC addresses with the PortChannel interfaces and not the underlying physical ports. Later, when a switch makes a forwarding decision to send a frame out a PortChannel interface, the switch must do more work: to decide out which specific physical port to use to forward the frame. IOS documentation refers to those rules as *EtherChannel load distribution* or *load balancing*. Figure 10-8 shows the main idea.

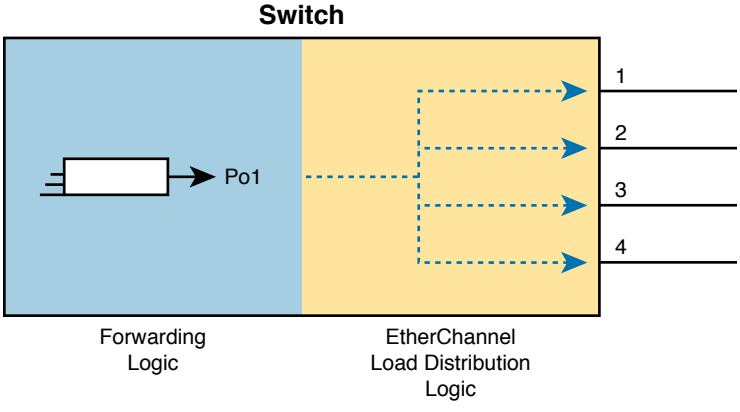


Figure 10-8 *Correct EtherChannel Configuration Combinations*

Configuration Options for EtherChannel Load Distribution

EtherChannel load distribution makes the choice for each frame based on various numeric values found in the Layer 2, 3, and 4 headers. The process uses one configurable setting as input: the load distribution method as defined with the **port-channel load-balance** *method* global command. The process then performs some match against the fields identified by the configured method.

Table 10-4 lists the most common methods. However, note that some switches may support only MAC-based methods, or only MAC- and IP-based methods, depending on the model and software version.

Table 10-4 EtherChannel Load Distribution Methods

Configuration Keyword	Math Uses...	Layer
src-mac	Source MAC address	2
dst-mac	Destination MAC address	2
src-dst-mac	Both source and destination MAC	2
src-ip	Source IP address	3
dst-ip	Destination IP address	3
src-dst-ip	Both source and destination IP	3
src-port	Source TCP or UDP port	4
dst-port	Destination TCP or UDP port	4
src-dst-port	Both source and destination TCP or UDP port	4

To appreciate why you might want to use different methods, you need to consider the results of how switches make their choice. (The discussion here focuses on the result, and not the logic, because the logic remains internal to the switch, and Cisco does not document how each switch model or IOS version works internally.) However, the various load distribution algorithms do share some common goals:

- To cause all messages in a single application flow to use the same link in the channel, rather than being sent over different links. Doing so means that the switch will not inadvertently reorder the messages sent in that application flow by sending one message over a busy link that has a queue of waiting messages, while immediately sending the next message out an unused link.
- To integrate the load distribution algorithm work into the hardware forwarding ASIC so that load distribution works just as quickly as the work to forward any other frame.
- To use all the active links in the EtherChannel, adjusting to the addition and removal of active links over time.
- Within the constraints of the other goals, balance the traffic across those active links.

In short, the algorithms first intend to avoid message reordering, make use of the switch forwarding ASICs, and use all the active links. However, the algorithm does not attempt to send the exact same number of bits over each link over time. The algorithm does try to balance the traffic, but always within the constraints of the other goals.

Whatever load distribution method you choose, the method identifies fields in the message headers. Any messages in the same application flow will then have the same values in the fields used by the load distribution algorithm and will always be forwarded over the same link. For example, when a user connects to a website, that web server may return thousands of packets to the client. Those thousands of packets should flow over the same link in the EtherChannel.

For instance, with the load distribution method of **src-mac** (meaning source MAC address), all frames with the same MAC address flow over one link. Figure 10-9 shows the idea with pseudo MAC addresses, with the load distribution sending frames with source MAC 1 over link 1, source MAC 2 over link 2, and source MAC 3 over link 3.

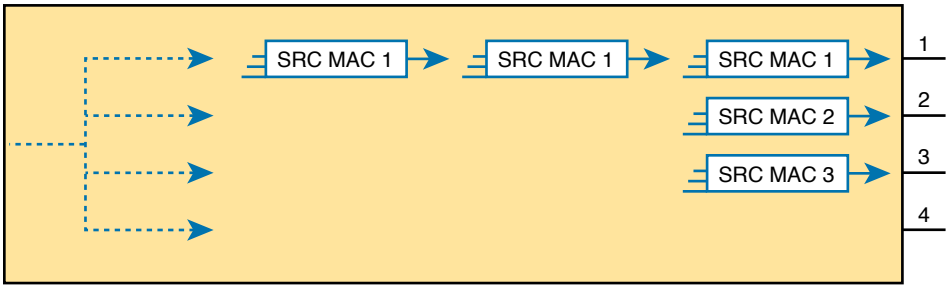


Figure 10-9 *Distributing All Frames with Same Mac Out Same Interface*

Cisco provides a variety of load distribution options so that the engineer can examine the flows in the network with the idea of finding which fields have the most variety in their values: source and destination MAC, or IP address, or transport layer port numbers. The more variety in the values in the fields, the better the balancing effects, and the lower the chance of sending disproportionate amounts of traffic over one link.

NOTE The algorithm focuses on the low-order bits in the fields in the headers because the low-order bits typically differ the most in real networks, while the high-order bits do not differ much. By focusing on the lower-order bits, the algorithm achieves better balancing of traffic over the links.

10

The Effects of the EtherChannel Load Distribution Algorithm

Figure 10-10 details a new EtherChannel that will be used in two examples to show the effects of load distribution. The examples will focus on frames sent by switch SW1 in the figure, showing the use of the **test etherchannel load-balance** EXEC command. That command asks the switch to consider some addresses or ports and answer the question: which link would you use when forwarding a message with those address/port values?

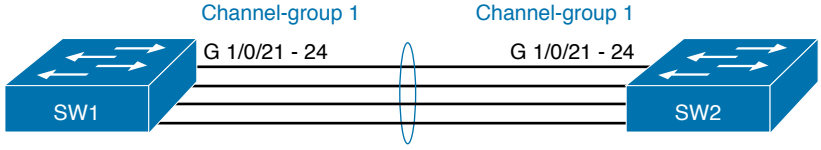


Figure 10-10 *Four-Link EtherChannel*

Example 10-7 shows how switch SW1 distributes traffic when using **src-mac** load distribution. The example lists the output from three of the **test etherchannel load-balance** commands, but note that all three commands use the same source MAC address. As a result, the answer from each command references the same interface (G1/0/22 in this case).

Example 10-7 *Testing with Identical Source MACs When Using* **src-mac** *Balancing*

```
SW1# show etherchannel load-balance
EtherChannel Load-Balancing Configuration:
        src-mac

EtherChannel Load-Balancing Addresses Used Per-Protocol:
Non-IP: Source MAC address
  IPv4: Source MAC address
  IPv6: Source MAC address

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1111
Would select Gi1/0/22 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1112
Would select Gi1/0/22 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1113
Would select Gi1/0/22 of Po1
```

Example 10-7 makes two important points:

- All three tests list the same outgoing physical interface because (1) the method uses only the source MAC address, and (2) all three tests use the same MAC addresses.
- All three tests use a different destination MAC address, with different low-order bits, but that had no impact on the choice because the method—**src-mac**—does not consider the destination MAC address.

In contrast on that first point, Example 10-8 repeats the **test** commands from Example 10-7. The switch still uses the **src-mac** balancing method, but now with different source MAC addresses in each test. Notice that the source MAC addresses used in the tests differ by just a few bit values in the low-order bits, so as a result, each test shows a different interface choice by SW1.

Example 10-8 *Testing with Source MACs with Low-Order Bit Differences*

```
SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1111
Would select Gi1/0/22 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0002 0200.1111.1111
Would select Gi1/0/24 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0003 0200.1111.1111
Would select Gi1/0/23 of Po1
```

Example 10-9 shows yet a third variation, this time changing the load distribution method to **src-dst-mac**, which means that the switch will consider both source and destination MAC. The example repeats the exact same **test etherchannel** commands as Example 10-7, with the exact same MAC addresses: the source MAC addresses remain the same in all three tests, but the destination MAC addresses differ in the low-order bits. With the chosen destination MAC values differing slightly, switch SW1 happens to choose three different interfaces.

Example 10-9 *Evidence of Source and Destination MAC Load Distribution*

```
SW1# config t
Enter configuration commands, one per line.  End with CNTL/Z.
SW1(config)# port-channel load-balance src-dst-mac
SW1(config)# ^Z
SW1#
SW1# show etherchannel load-balance
EtherChannel Load-Balancing Configuration:
        src-dst-mac

EtherChannel Load-Balancing Addresses Used Per-Protocol:
Non-IP: Source XOR Destination MAC address
  IPv4: Source XOR Destination MAC address
  IPv6: Source XOR Destination MAC address

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1111
Would select Gi1/0/22 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1112
Would select Gi1/0/24 of Po1

SW1# test etherchannel load-balance interface po1 mac 0200.0000.0001 0200.1111.1113
Would select Gi1/0/23 of Po1
```


Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 10-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 10-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review config checklists		Book, website
Review command tables		Book
Review memory tables		Website
Do labs		Blog

Review All the Key Topics

Key Topic

Table 10-6 Key Topics for Chapter 10

Key Topic Element	Description	Page Number
Figure 10-1	Typical design choice for which switches should be made to be root	241
Figure 10-2	Conceptual view of load-balancing benefits of PVST+	242
Table 10-2	STP Standards and Configuration Options	243
Figure 10-4	Shows the format of the system ID extension of the STP priority field	243
List	Facts about RPVST+'s methods versus RSTP	246
List	Steps to manually configure an EtherChannel	248
List	Items a switch compares in a new physical port's configuration to the existing ports in the channel	252

Key Terms You Should Know

Rapid PVST+, PVST+, system ID extension, PAgP, LACP, PortChannel, Channel-group, EtherChannel, EtherChannel Load Distribution, primary root, secondary root

Command References

Tables 10-7 and 10-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 10-7 Chapter 10 Configuration Command Reference

Command	Description
spanning-tree mode {**pvst** | **rapid-pvst** | **mst**}	Global configuration command to set the STP mode.
spanning-tree [**vlan** *vlan-number*] **root primary**	Global configuration command that changes this switch to the root switch. The switch's priority is changed to the lower of either 24,576 or 4096 less than the priority of the current root bridge when the command was issued.
spanning-tree [**vlan** *vlan-number*] **root secondary**	Global configuration command that sets this switch's STP base priority to 28,672.
spanning-tree vlan *vlan-id* **priority** *priority*	Global configuration command that changes the bridge priority of this switch for the specified VLAN.
spanning-tree [**vlan** *vlan-number*] **cost** *cost*	Interface subcommand that changes the STP cost to the configured value.
spanning-tree [**vlan** *vlan-number*] **port-priority** *priority*	Interface subcommand that changes the STP port priority in that VLAN (0 to 240, in increments of 16).
channel-group *channel-group-number* **mode** {**auto** | **desirable** | **active** | **passive** | **on**}	Interface subcommand that enables EtherChannel on the interface.

Table 10-8 Chapter 10 EXEC Command Reference

Command	Description
show spanning-tree	Lists details about the state of STP on the switch, including the state of each port.
show spanning-tree vlan *vlan-id*	Lists STP information for the specified VLAN.
show etherchannel [*channel-group-number*] {**brief** | **detail** | **port** | **port-channel** | **summary**}	Lists information about the state of EtherChannels on this switch.

Part III Review

Keep track of your part review progress with the checklist shown in Table P3-1. Details on each task follow the table.

Table P3-1 Part III Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Review Appendices		
Videos		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book, using the PCPT software.

Answer Part Review Questions

For this task, answer the Part Review questions for this part of the book, using the PTP software.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson ICND1 or CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.) Note that the Sim Lite that comes with this book also has a couple of labs about VLANs.

Blog: Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book, as found at http://blog.certskills.com. Then navigate to the Hands-on Config labs.

Other: If using other lab tools, as a few suggestions: Make sure and experiment heavily with VLAN configuration and VLAN trunking configuration.

Dig Deeper with Appendices on the Companion Website

The chapters in Part III of the book recommended the following appendices for extra reading. If you care to read further, consider:

- **Appendix K, "Analyzing Ethernet LAN Designs":** A chapter from the previous edition that discusses design topologies and LAN design with two-tier and three-tier designs, including access and distribution switches.
- **Appendix O, "Spanning Tree Protocol Implementation":** A chapter that works through the configuration and verification commands for STP and RSTP.
- **Appendix P, "LAN Troubleshooting":** A chapter from the previous edition of the ICND2 Cert Guide. This chapter includes topics about VLANs, trunks, and STP and how to troubleshoot each.

Watch Videos

Chapter 8 recommends two videos, one about VLANs and another about the VLAN allowed list on trunks. If you have not watched those videos yet, take a moment to scan back to Chapter 8 on the companion website and watch the videos.

The book makes a big transition at this point. Part I gave you a broad introduction to networking, and Parts II and III went into some detail about the dominant LAN technology today: Ethernet. Part IV transitions from Ethernet to the network layer details that sit above Ethernet and WAN technology, specifically IP Version 4 (IPv4).

Thinking about the network layer requires engineers to shift how they think about addressing. Ethernet allows the luxury of using universal MAC addresses, assigned by the manufacturers, with no need to plan or configure addresses. Although the network engineer needs to understand MAC addresses, MAC already exists on each Ethernet NIC, and switches learn the Ethernet MAC addresses dynamically without even needing to be configured to do so. As a result, most people operating the network can ignore the specific MAC address values for most tasks.

Conversely, IP addressing gives us flexibility and allows choice, but those features require planning, along with a much deeper understanding of the internal structure of the addresses. People operating the network must be more aware of the network layer addresses when doing many tasks. To better prepare you for these Layer 3 addressing details, this part breaks down the addressing details into four chapters, with an opportunity to learn more in preparation for the CCNP Enterprise certification.

Part IV examines most of the basic details of IPv4 addressing and subnetting, mostly from the perspective of operating an IP network. Chapter 11 takes a grand tour of IPv4 addressing as implemented inside a typical enterprise network. Chapters 12, 13, and 14 look at some of the specific questions people must ask themselves when operating an IPv4 network.

Part IV

IPv4 Addressing

CHAPTER 11

Perspectives on IPv4 Subnetting

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

1.7 Describe the need for private IPv4 addressing

Most entry-level networking jobs require you to operate and troubleshoot a network using a preexisting IP addressing and subnetting plan. The CCNA exam assesses your readiness to use preexisting IP addressing and subnetting information to perform typical operations tasks, such as monitoring the network, reacting to possible problems, configuring addresses for new parts of the network, and troubleshooting those problems.

However, you also need to understand how networks are designed and why. Anyone monitoring a network must continually ask the question, "Is the network working *as designed*?" If a problem exists, you must consider questions such as "What happens when the network works normally, and what is different right now?" Both questions require you to understand the intended design of the network, including details of the IP addressing and subnetting design.

This chapter provides some perspectives and answers for the bigger issues in IPv4 addressing. What addresses can be used so that they work properly? What addresses should be used? When told to use certain numbers, what does that tell you about the choices made by some other network engineer? How do these choices impact the practical job of configuring switches, routers, hosts, and operating the network on a daily basis? This chapter hopes to answer these questions while revealing details of how IPv4 addresses work.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 11-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Analyze Requirements	1–3
Make Design Choices	4–7

1. Host A is a PC, connected to switch SW1 and assigned to VLAN 1. Which of the following are typically assigned an IP address in the same subnet as host A? (Choose two answers.)
   - **a.** The local router's WAN interface
   - **b.** The local router's LAN interface
   - **c.** All other hosts attached to the same switch
   - **d.** Other hosts attached to the same switch and also in VLAN 1
2. Why does the formula for the number of hosts per subnet ($2^H - 2$) require the subtraction of two hosts?
   - **a.** To reserve two addresses for redundant default gateways (routers)
   - **b.** To reserve the two addresses required for DHCP operation
   - **c.** To reserve addresses for the subnet ID and default gateway (router)
   - **d.** To reserve addresses for the subnet broadcast address and subnet ID
3. A Class B network needs to be subnetted such that it supports 100 subnets and 100 hosts/subnet. Which of the following answers list a workable combination for the number of network, subnet, and host bits? (Choose two answers.)
   - **a.** Network = 16, subnet = 7, host = 7
   - **b.** Network = 16, subnet = 8, host = 8
   - **c.** Network = 16, subnet = 9, host = 7
   - **d.** Network = 8, subnet = 7, host = 17
4. Which of the following are private IP networks? (Choose two answers.)
   - **a.** 172.31.0.0
   - **b.** 172.32.0.0
   - **c.** 192.168.255.0
   - **d.** 192.1.168.0
   - **e.** 11.0.0.0
5. Which of the following are public IP networks? (Choose three answers.)
   - **a.** 9.0.0.0
   - **b.** 172.30.0.0
   - **c.** 192.168.255.0
   - **d.** 192.1.168.0
   - **e.** 1.0.0.0

6. Before Class B network 172.16.0.0 is subnetted by a network engineer, what parts of the structure of the IP addresses in this network already exist, with a specific size? (Choose two answers.)
   a. Network
   b. Subnet
   c. Host
   d. Broadcast

7. A network engineer spends time thinking about the entire Class B network 172.16.0.0 and how to subnet that network. He then chooses how to subnet this Class B network and creates an addressing and subnetting plan, on paper, showing his choices. If you compare his thoughts about this network before subnetting the network to his thoughts about this network after mentally subnetting the network, which of the following occurred to the parts of the structure of addresses in this network?
   a. The subnet part got smaller.
   b. The host part got smaller.
   c. The network part got smaller.
   d. The host part was removed.
   e. The network part was removed.

Foundation Topics

Introduction to Subnetting

Say you just happened to be at the sandwich shop when it was selling the world's longest sandwich. You're pretty hungry, so you go for it. Now you have one sandwich, but because it's over 2 kilometers long, you realize it's a bit more than you need for lunch all by yourself. To make the sandwich more useful (and more portable), you chop the sandwich into meal-size pieces and give the pieces to other folks around you who are also ready for lunch.

Huh? Well, subnetting, at least the main concept, is similar to this sandwich story. You start with one network, but it is just one large network. As a single large entity, it might not be useful, and it is probably far too large. To make it useful, you chop it into smaller pieces, called subnets, and assign those subnets to be used in different parts of the enterprise internetwork.

This short first section of the chapter introduces IP subnetting. First, it shows the general ideas behind a completed subnet design that indeed chops (or subnets) one network into subnets. The rest of this section describes the many design steps that you would take to create just such a subnet design. By the end of this section, you should have the right context to then read through the subnetting design steps introduced throughout the rest of this chapter.

NOTE All the chapters from this chapter up until Chapter 22, "Fundamentals of IP Version 6," focus on IPv4 rather than IPv6. All references to *IP* refer to IPv4 unless otherwise stated.

Subnetting Defined Through a Simple Example

An IP network—in other words, a Class A, B, or C network—is simply a set of consecutively numbered IP addresses that follows some preset rules. These Class A, B, and C rules define that for a given network, all the addresses in the network have the same value in some of the octets of the addresses. For example, Class B network 172.16.0.0 consists of all IP addresses that begin with 172.16: 172.16.0.0, 172.16.0.1, 172.16.0.2, and so on, through 172.16.255.255. Another example: Class A network 10.0.0.0 includes all addresses that begin with 10.

An IP subnet is simply a subset of a Class A, B, or C network. In fact, the word *subnet* is a shortened version of the phrase *subdivided network*. For example, one subnet of Class B network 172.16.0.0 could be the set of all IP addresses that begin with 172.16.1, and would include 172.16.1.0, 172.16.1.1, 172.16.1.2, and so on, up through 172.16.1.255. Another subnet of that same Class B network could be all addresses that begin with 172.16.2.

To give you a general idea, Figure 11-1 shows some basic documentation from a completed subnet design that could be used when an engineer subnets Class B network 172.16.0.0.

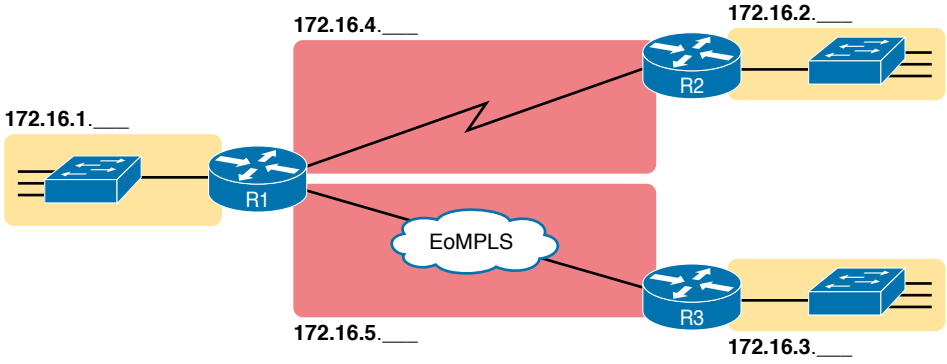


Figure 11-1 *Subnet Plan Document*

The design shows five subnets—one for each of the three LANs and one each for the two WAN links. The small text note shows the rationale used by the engineer for the subnets: each subnet includes addresses that have the same value in the first three octets. For example, for the LAN on the left, the number shows 172.16.1.__, meaning "all addresses that begin with 172.16.1." Also, note that the design, as shown, does not use all the addresses in Class B network 172.16.0.0, so the engineer has left plenty of room for growth.

Operational View Versus Design View of Subnetting

Most IT jobs require you to work with subnetting from an operational view. That is, someone else, before you got the job, designed how IP addressing and subnetting would work for that particular enterprise network. You need to interpret what someone else has already chosen.

To fully understand IP addressing and subnetting, you need to think about subnetting from both a design and operational perspective. For example, Figure 11-1 simply states that in all these subnets, the first three octets must be equal. Why was that convention chosen? What

alternatives exist? Would those alternatives be better for your internetwork today? All these questions relate more to subnetting design rather than to operation.

To help you see both perspectives, this chapter focuses more on design issues by moving through the entire design process for the purpose of introducing the bigger picture of IP subnetting. The next three chapters each take one topic from this chapter and examine it more closely but more from an operational perspective: how to use those ideas in real networks.

The remaining three main sections of this chapter examine each of the steps listed in Figure 11-2, in sequence.

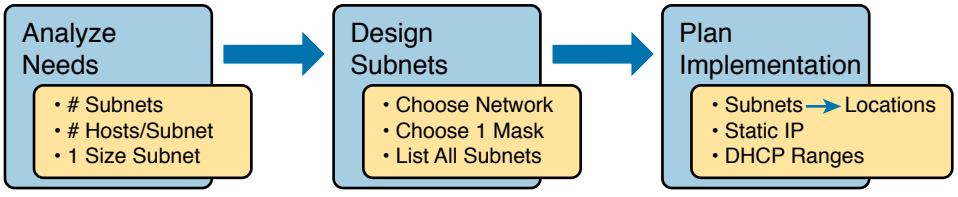


Figure 11-2 *Subnet Planning, Design, and Implementation Tasks*

Analyze Subnetting and Addressing Needs

This section discusses the meaning of four basic questions that can be used to analyze the addressing and subnetting needs for any new or changing enterprise network:

1. Which hosts should be grouped together into a subnet?
2. How many subnets does this internetwork require?
3. How many host IP addresses does each subnet require?
4. Will we use a single subnet size for simplicity, or not?

Rules About Which Hosts Are in Which Subnet

Every device that connects to an IP internetwork needs to have an IP address. These devices include computers used by end users, servers, mobile phones, laptops, IP phones, tablets, and networking devices like routers, switches, and firewalls. In short, any device that uses IP to send and receive packets needs an IP address.

NOTE In a discussion of IP addressing, the term *network* has specific meaning: a Class A, B, or C IP network. To avoid confusion with that use of the term *network*, this book uses the terms *internetwork* and *enterprise network* when referring to a collection of hosts, routers, switches, and so on.

Answers to the "Do I Know This Already?" quiz:

1 B, D **2** D **3** B, C **4** A, C **5** A, D, E **6** A, C **7** B

The IP addresses must be assigned according to some basic rules—and for good reasons. To make routing work efficiently, IP addressing rules group addresses into groups called subnets. The rules are as follows:

Key Topic

- Addresses in the same subnet are not separated by a router.
- Addresses in different subnets are separated by at least one router.

Figure 11-3 shows the general concept, with hosts A and B in one subnet and host C in another. In particular, note that hosts A and B are not separated from each other by any routers. However, host C, separated from A and B by at least one router, must be in a different subnet.

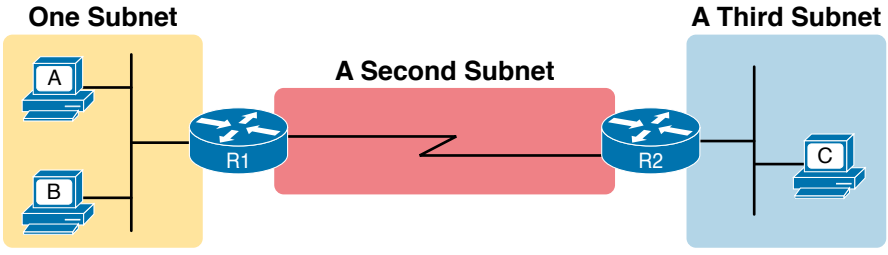


Figure 11-3 *PC A and B in One Subnet and PC C in a Different Subnet*

The idea that hosts on the same link must be in the same subnet is much like the postal code concept. All mailing addresses in the same town use the same postal code (ZIP codes in the United States). Addresses in another town, whether relatively nearby or on the other side of the country, have a different postal code. The postal code gives the postal service a better ability to automatically sort the mail to deliver it to the right location. For the same general reasons, hosts on the same LAN are in the same subnet, and hosts in different LANs are in different subnets.

Note that the point-to-point WAN link in the figure also needs a subnet. Figure 11-3 shows Router R1 connected to the LAN subnet on the left and to a WAN subnet on the right. Router R2 connects to that same WAN subnet. To do so, both R1 and R2 will have IP addresses on their WAN interfaces, and the addresses will be in the same subnet. (An Ethernet WAN link has the same IP addressing needs, with each of the two routers having an IP address in the same subnet.)

The Ethernet LANs in Figure 11-3 also show a slightly different style of drawing, using simple lines with no Ethernet switch. Drawings of Ethernet LANs when the details of the LAN switches do not matter simply show each device connected to the same line, as shown in Figure 11-3. (This kind of drawing mimics the original Ethernet cabling before switches and hubs existed.)

Finally, because the routers' main job is to forward packets from one subnet to another, routers typically connect to multiple subnets. For example, in this case, Router R1 connects to one LAN subnet on the left and one WAN subnet on the right. To do so, R1 will be configured with two different IP addresses, one per interface. These addresses will be in different subnets because the interfaces connect the router to different subnets.

Determining the Number of Subnets

To determine the number of subnets required, the engineer must think about the internetwork as documented and count the locations that need a subnet. To do so, the engineer requires access to network diagrams, VLAN configuration details, and details about WAN links. For the types of links discussed in this book, you should plan for one subnet for every

Key Topic

- VLAN
- Point-to-point serial link
- Ethernet WAN (Ethernet Line Service)

NOTE Other WAN technologies outside the scope of the CCNA exam topics allow subnetting options other than one subnet per pair of routers on the WAN (as shown here). However, this book only uses point-to-point WAN technologies—serial links and Ethernet WAN links—that have one subnet for each point-to-point WAN connection between two routers.

For example, imagine that the network planner has only Figure 11-4 on which to base the subnet design.

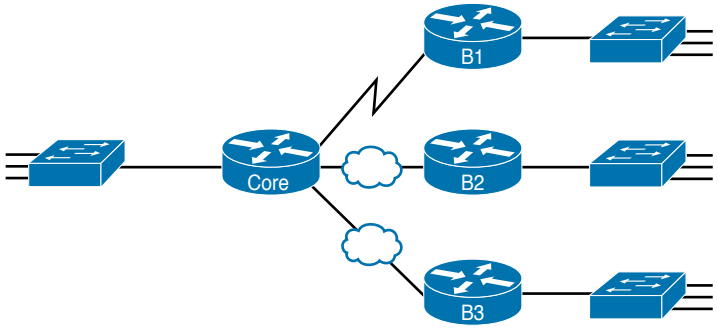


Figure 11-4 *Four-Site Internetwork with Small Central Site*

The number of subnets required cannot be fully predicted with only this figure. Certainly, three subnets will be needed for the WAN links, one per link. However, each LAN switch can be configured with a single VLAN or with multiple VLANs. You can be certain that you need at least one subnet for the LAN at each site, but you might need more.

Next, consider the more detailed version of the same figure shown in Figure 11-5. In this case, the figure shows VLAN counts in addition to the same Layer 3 topology (the routers and the links connected to the routers). It also shows that the central site has many more switches, but the key fact on the left, regardless of how many switches exist, is that the central site has a total of 12 VLANs. Similarly, the figure lists each branch as having two VLANs. Along with the same three WAN subnets, this internetwork requires 21 subnets.

Finally, in a real job, you would consider the needs today as well as how much growth you expect in the internetwork over time. Any subnetting plan should include a reasonable estimate of the number of subnets you need to meet future needs.

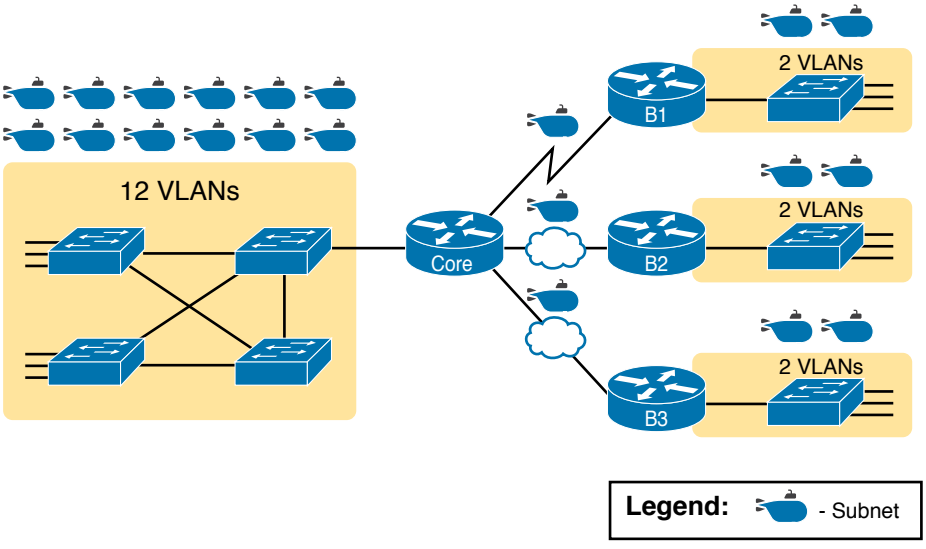


Figure 11-5 *Four-Site Internetwork with Larger Central Site*

Determining the Number of Hosts per Subnet

Determining the number of hosts per subnet requires knowing a few simple concepts and then doing a lot of research and questioning. Every device that connects to a subnet needs an IP address. For a totally new network, you can look at business plans—numbers of people at the site, devices on order, and so on—to get some idea of the possible devices. When expanding an existing network to add new sites, you can use existing sites as a point of comparison and then find out which sites will get bigger or smaller. And don't forget to count the router interface IP address in each subnet and the switch IP address used to remotely manage the switch.

Instead of gathering data for each and every site, planners often just use a few typical sites for planning purposes. For example, maybe you have some large sales offices and some small sales offices. You might dig in and learn a lot about only one large sales office and only one small sales office. Add that analysis to the fact that point-to-point links need a subnet with just two addresses, plus any analysis of more one-of-a-kind subnets, and you have enough information to plan the addressing and subnetting design.

For example, in Figure 11-6, the engineer has built a diagram that shows the number of hosts per LAN subnet in the largest branch, B1. For the two other branches, the engineer did not bother to dig to find out the number of required hosts. As long as the number of required IP addresses at sites B2 and B3 stays below the estimate of 50, based on larger site B1, the engineer can plan for 50 hosts in each branch LAN subnet and have plenty of addresses per subnet.

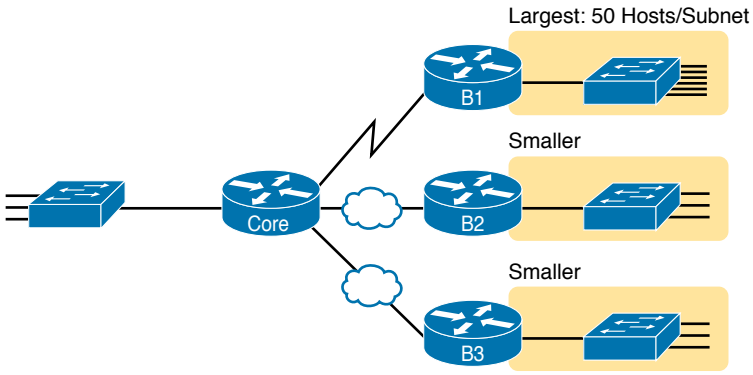


Figure 11-6 *Large Branch B1 with 50 Hosts/Subnet*

One Size Subnet Fits All—Or Not

The final choice in the initial planning step is to decide whether you will use a simpler design by using a one-size-subnet-fits-all philosophy. A subnet's size, or length, is simply the number of usable IP addresses in the subnet. A subnetting design can either use one size subnet or varied sizes of subnets, with pros and cons for each choice.

Defining the Size of a Subnet

Before you finish this book, you will learn all the details of how to determine the size of the subnet. For now, you just need to know a few specific facts about the size of subnets. Chapter 12, "Analyzing Classful IPv4 Networks," and Chapter 13, "Analyzing Subnet Masks," give you a progressively deeper knowledge of the details.

The engineer assigns each subnet a *subnet mask*, and that mask, among other things, defines the size of that subnet. The mask sets aside a number of *host bits* whose purpose is to number different host IP addresses in that subnet. Because you can number 2^x things with x bits, if the mask defines H host bits, the subnet contains 2^H unique numeric values.

However, the subnet's size is not 2^H. It's $2^H - 2$ because two numbers in each subnet are reserved for other purposes. Each subnet reserves the numerically lowest value for the *subnet number* and the numerically highest value as the *subnet broadcast address*. As a result, the number of usable IP addresses per subnet is $2^H - 2$.

NOTE The terms *subnet number*, *subnet ID*, and *subnet address* all refer to the number that represents or identifies a subnet.

Figure 11-7 shows the general concept behind the three-part structure of an IP address, focusing on the host part and the resulting subnet size.

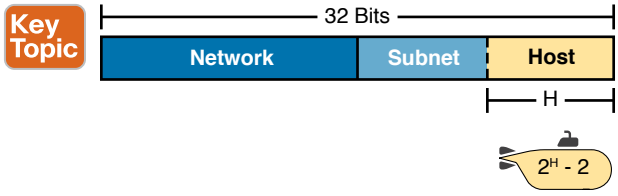


Figure 11-7 *Subnet Size Concepts*

One Size Subnet Fits All

To choose to use a single-size subnet in an enterprise network, you must use the same mask for all subnets because the mask defines the size of the subnet. But which mask?

One requirement to consider when choosing that one mask is this: that one mask must provide enough host IP addresses to support the largest subnet. To do so, the number of host bits (H) defined by the mask must be large enough so that $2^H - 2$ is larger than (or equal to) the number of host IP addresses required in the largest subnet.

For example, consider Figure 11-8. It shows the required number of hosts per LAN subnet. (The figure ignores the subnets on the WAN links, which require only two IP addresses each.) The branch LAN subnets require only 50 host addresses, but the main site LAN subnet requires 200 host addresses. To accommodate the largest subnet, you need at least 8 host bits. Seven host bits would not be enough because $2^7 - 2 = 126$. Eight host bits would be enough because $2^8 - 2 = 254$, which is more than enough to support 200 hosts in a subnet.

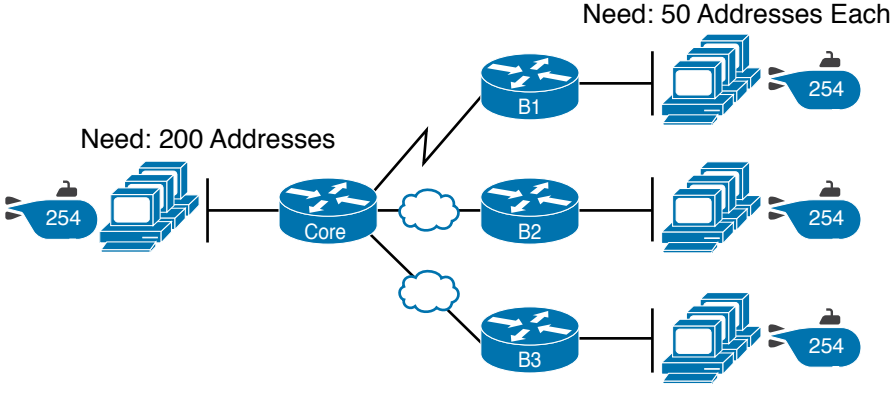


Figure 11-8 *Network Using One Subnet Size*

What's the big advantage when using a single-size subnet? Operational simplicity. In other words, keeping it simple. Everyone on the IT staff who has to work with networking can get used to working with one mask—and one mask only. Staff members will be able to answer all subnetting questions more easily because everyone gets used to doing subnetting math with that one mask.

The big disadvantage for using a single-size subnet is that it wastes IP addresses. For example, in Figure 11-8, all the branch LAN subnets support 254 addresses, while the largest branch subnet needs only 50 addresses. The WAN subnets only need two IP addresses, but each supports 254 addresses, again wasting more IP addresses.

The wasted IP addresses do not actually cause a problem in most cases, however. Most organizations use private IP networks in their enterprise internetworks, and a single Class A or Class B private network can supply plenty of IP addresses, even with the waste.

Multiple Subnet Sizes (Variable-Length Subnet Masks)

To create multiple sizes of subnets in one Class A, B, or C network, the engineer must create some subnets using one mask, some with another, and so on. Different masks mean different numbers of host bits, and a different number of hosts in some subnets based on the $2^H - 2$ formula.

For example, consider the requirements listed earlier in Figure 11-8. It showed one LAN subnet on the left that needs 200 host addresses, three branch subnets that need 50 addresses, and three WAN links that need two addresses. To meet those needs, but waste fewer IP addresses, three subnet masks could be used, creating subnets of three different sizes, as shown in Figure 11-9.

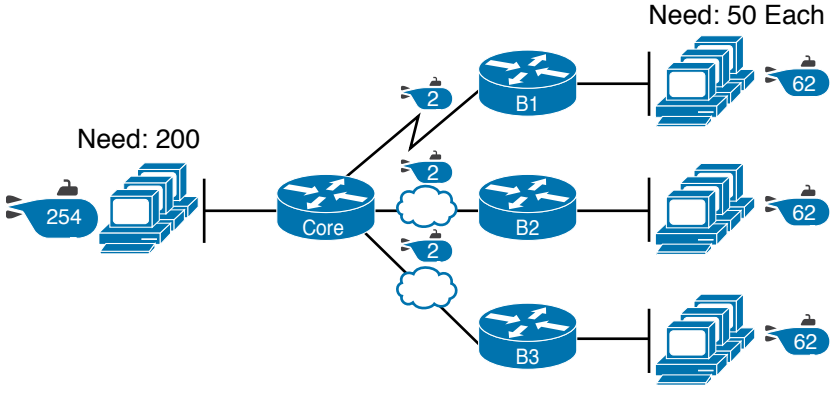


Figure 11-9 *Three Masks, Three Subnet Sizes*

The smaller subnets now waste fewer IP addresses compared to the design shown earlier in Figure 11-8. The subnets on the right that need 50 IP addresses have subnets with 6 host bits, for $2^6 - 2 = 62$ available addresses per subnet. The WAN links use masks with 2 host bits, for $2^2 - 2 = 2$ available addresses per subnet.

However, some are still wasted because you cannot set the size of the subnet as some arbitrary size. All subnets will be a size based on the $2^H - 2$ formula, with H being the number of host bits defined by the mask for each subnet.

One Mask for All Subnets, or More Than One

For the most part, this book explains subnetting using designs that use a single mask, creating a single subnet size for all subnets. Why? First, it makes the process of learning subnetting easier. Second, some types of analysis that you can do about a network—specifically, calculating the number of subnets in the classful network—only make sense when a single mask is used.

However, you still need to be ready to work with designs that use more than one mask in different subnets of the same Class A, B, or C network. In fact, a design that does just that is said to be using *variable-length subnet masks (VLSM)*. For example, the internetwork in Figure 11-10 shows 11 subnets, two with a mask of /30, and nine with a mask of /24. By using more than one mask among all the subnets of one Class A network (10.0.0.0), the design uses VLSM.

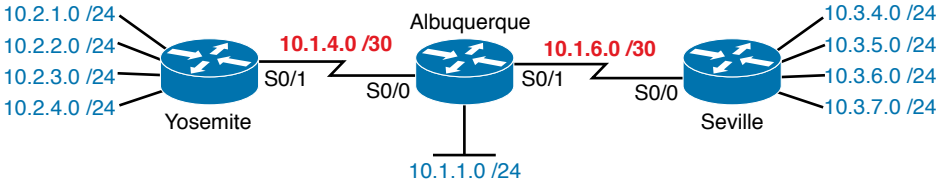


Figure 11-10 *Internetwork with VLSM: Network 10.0.0.0, >1 Mask*

For the current CCNA 200-301 exam, using VLSM causes no issues, although it does cause problems with some older routing protocols. The only routing protocol included in the CCNA blueprint (OSPF) works the same regardless of whether the design uses VLSM. Just be aware of the term and what it means and that it should not impact the features included in the current CCNA exam.

NOTE VLSM has been featured in the CCNA exam topics in the past. If you want to read a little more about VLSM, check out Appendix N, "Variable-Length Subnet Masks," on the companion website for this book.

Make Design Choices

Now that you know how to analyze the IP addressing and subnetting needs, the next major step examines how to apply the rules of IP addressing and subnetting to those needs and make some choices. In other words, now that you know how many subnets you need and how many host addresses you need in the largest subnet, how do you create a useful subnetting design that meets those requirements? The short answer is that you need to do the three tasks shown on the right side of Figure 11-11.

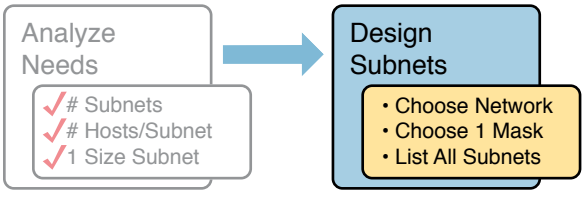


Figure 11-11 *Input to the Design Phase, and Design Questions to Answer*

Choose a Classful Network

In the original design for what we know of today as the Internet, companies used registered *public classful IP networks* when implementing TCP/IP inside the company. By the

mid-1990s, an alternative became more popular: *private IP networks*. This section discusses the background behind these two choices because it impacts the choice of what IP network a company will then subnet and implement in its enterprise internetwork.

Public IP Networks

The original design of the Internet required that any company that connected to the Internet had to use a *registered public IP network*. To do so, the company would complete some paperwork, describing the enterprise's internetwork and the number of hosts existing, plus plans for growth. After submitting the paperwork, the company would receive an assignment of either a Class A, B, or C network.

Public IP networks—and the administrative processes surrounding them—ensure that all the companies that connect to the Internet all use unique IP addresses. In particular, after a public IP network is assigned to a company, only that company should use the addresses in that network. That guarantee of uniqueness means that Internet routing can work well because there are no duplicate public IP addresses.

For example, consider the example shown in Figure 11-12. Company 1 has been assigned public Class A network 1.0.0.0, and company 2 has been assigned public Class A network 2.0.0.0. Per the original intent for public addressing in the Internet, after these public network assignments have been made, no other companies can use addresses in Class A networks 1.0.0.0 or 2.0.0.0.

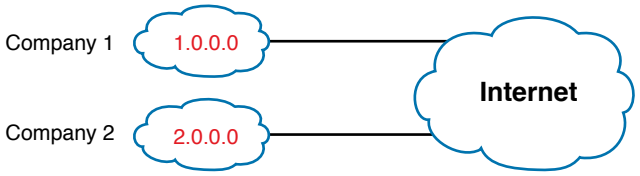


Figure 11-12 *Two Companies with Unique Public IP Networks*

This original address assignment process ensured unique IP addresses across the entire planet. The idea is much like the fact that your telephone number should be unique in the universe, your postal mailing address should also be unique, and your email address should also be unique. If someone calls you, your phone rings, but no one else's phone rings. Similarly, if company 1 is assigned Class A network 1.0.0.0, and the engineers at Company 1 assign address 1.1.1.1 to a particular PC, that address should be unique in the universe. A packet sent through the Internet to destination 1.1.1.1 should only arrive at this one PC inside company 1, instead of being delivered to some other host.

Growth Exhausts the Public IP Address Space

By the early 1990s, the world was running out of public IP networks that could be assigned. During most of the 1990s, the number of hosts newly connected to the Internet was growing at a double-digit pace *per month*. Companies kept following the rules, asking for public IP networks, and it was clear that the current address-assignment scheme could not continue without some changes. Simply put, the number of Class A, B, and C networks supported by the 32-bit address in IP version 4 (IPv4) was not enough to support one public classful network per organization, while also providing enough IP addresses in each company.

NOTE The universe has run out of public IPv4 addresses in a couple of significant ways. IANA, which assigns public IPv4 address blocks to the five Regional Internet Registries (RIR) around the globe, assigned the last of the IPv4 address spaces in early 2011. By 2015, ARIN, the RIR for North America, exhausted its supply of IPv4 addresses, so companies must return unused public IPv4 addresses to ARIN before they have more to assign to new companies. Try an online search for "ARIN depletion" to see pages about the current status of available IPv4 address space for just one RIR example.

The Internet community worked hard during the 1990s to solve this problem, coming up with several solutions, including the following:

Key Topic

- A new version of IP (IPv6), with much larger addresses (128 bit)
- Assigning a subset of a public IP network to each company, instead of an entire public IP network, to reduce waste, using a feature called "Classless Interdomain Routing" (CIDR)
- Network Address Translation (NAT), which allows the use of private IP networks

These three solutions matter to real networks today. However, to stay focused on the topic of subnet design, this chapter focuses on the third option, and in particular, the private IP networks that can be used by an enterprise when also using NAT. (Be aware that Chapter 10, "Network Address Translation" in *CCNA 200-301 Official Cert Guide*, *Volume 2*, gives more detail about the last two bullets in the list, while Part VII of this book discusses the first bullet item (IPv6) in more depth.

Focusing on the third item in the bullet list, NAT allows multiple companies to use the exact same *private IP network*, using the same IP addresses as other companies while still connecting to the Internet. For example, Figure 11-13 shows the same two companies connecting to the Internet as in Figure 11-12, but now with both using the same private Class A network 10.0.0.0.

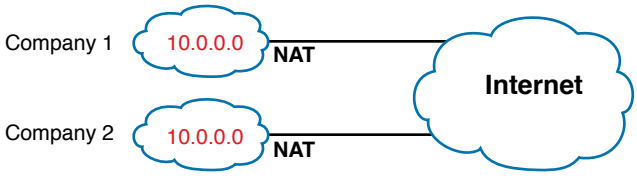


Figure 11-13 *Reusing the Same Private Network 10.0.0.0 with NAT*

Both companies use the same classful IP network (10.0.0.0). Both companies can implement their subnet design internal to their respective enterprise internetworks, without discussing their plans. The two companies can even use the exact same IP addresses inside network 10.0.0.0. And amazingly, at the same time, both companies can even communicate with each other through the Internet.

The technology called Network Address Translation makes it possible for companies to reuse the same IP networks, as shown in Figure 11-13. NAT does this by translating the IP addresses inside the packets as they go from the enterprise to the Internet, using a small number of public IP addresses to support tens of thousands of private IP addresses. That one bit of information is not enough to understand how NAT works; however, to keep the focus

on subnetting, the book defers the discussion of how NAT works until *CCNA 200-301 Official Cert Guide, Volume 2*. For now, accept that most companies use NAT, and therefore, they can use private IP networks for their internetworks.

Private IP Networks

When using NAT—and almost every organization that connects to the Internet uses NAT—the company can simply pick one or more of the private IP networks from the list of reserved private IP network numbers. RFC 1918 defines the list of available private IP networks, which is summarized in Table 11-2.

Table 11-2 RFC 1918 Private Address Space

Class of Networks	Private IP Networks	Number of Networks
A	10.0.0.0	1
B	172.16.0.0 through 172.31.0.0	16
C	192.168.0.0 through 192.168.255.0	256

NOTE According to an informal survey I ran on my blog a few years back, about half of the respondents said that their networks use private Class A network 10.0.0.0, as opposed to other private networks or public networks.

From the perspective of making IPv4 work for the entire world, private IP networks have helped preserve and extend IPv4 and its use in every enterprise and throughout the Internet. In particular, private networks have improved IPv4's implementation worldwide by

Key Topic

- **Avoiding Using Another Organization's Public Address Range for Private Networks:** Some organizations have a part of their networks that need zero Internet access. The hosts in that part of their network need IP addresses. RFC 1918 suggests that truly private networks—that is, networks with no need for Internet connectivity—use addresses from the RFC 1918 list of private networks.
- **Avoiding/Delaying IPv4 Address Exhaustion:** To delay the day in which all public IPv4 addresses were assigned to organizations as public addresses, RFC 1918 calls for the use of NAT along with private networks for the addresses internal to an organization.
- **Reducing Internet Routers' Routing Table Size:** Using private networks also helps reduce the size of the IP routing tables in Internet routers. For instance, routers in the Internet do not need routes for the private IP networks used inside organizations (in fact, ISPs filter those routes).

Choosing an IP Network During the Design Phase

Today, some organizations use private IP networks along with NAT, and some use public IP networks. Most new enterprise internetworks use private IP addresses throughout the network, along with NAT, as part of the connection to the Internet. Those organizations that already have registered public IP networks—often obtained before the addresses started

running short in the early 1990s—can continue to use those public addresses throughout their enterprise networks.

After the choice to use a private IP network has been made, just pick one that has enough IP addresses. You can have a small internetwork and still choose to use private Class A network 10.0.0.0. It might seem wasteful to choose a Class A network that has over 16 million IP addresses, especially if you need only a few hundred. However, there's no penalty or problem with using a private network that is too large for your current or future needs.

For the purposes of this book, most examples use private IP network numbers. For the design step to choose a network number, just choose a private Class A, B, or C network from the list of RFC 1918 private networks.

Regardless, from a math and concept perspective, the methods to subnet a public IP network versus a private IP network are the same.

Choose the Mask

If a design engineer followed the topics in this chapter so far, in order, he would know the following:

- The number of subnets required
- The number of hosts/subnet required
- That a choice was made to use only one mask for all subnets so that all subnets are the same size (same number of hosts/subnet)
- The classful IP network number that will be subnetted

This section completes the design process, at least the parts described in this chapter, by discussing how to choose that one mask to use for all subnets. First, this section examines default masks, used when a network is not subnetted, as a point of comparison. Next, the concept of borrowing host bits to create subnet bits is explored. Finally, this section ends with an example of how to create a subnet mask based on the analysis of the requirements.

Classful IP Networks Before Subnetting

Before an engineer subnets a classful network, the network is a single group of addresses. In other words, the engineer has not yet subdivided the network into many smaller subsets called *subnets*.

When thinking about an unsubnetted classful network, the addresses in a network have only two parts: the network part and host part. Comparing any two addresses in the classful network:

- The addresses have the same value in the network part.
- The addresses have different values in the host part.

The actual sizes of the network and host part of the addresses in a network can be easily predicted, as shown in Figure 11-14.

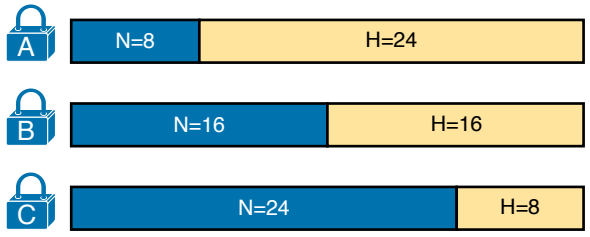


Figure 11-14 *Format of Unsubnetted Class A, B, and C Networks*

In Figure 11-14, N and H represent the number of network and host bits, respectively. Class rules define the number of network octets (1, 2, or 3) for Classes A, B, and C, respectively; the figure shows these values as a number of bits. The number of host octets is 3, 2, or 1, respectively.

Continuing the analysis of classful network before subnetting, the number of addresses in one classful IP network can be calculated with the same $2^H - 2$ formula previously discussed. In particular, the size of an unsubnetted Class A, B, or C network is as follows:

- **Class A:** $2^{24} - 2 = 16{,}777{,}214$
- **Class B:** $2^{16} - 2 = 65{,}534$
- **Class C:** $2^8 - 2 = 254$

Borrowing Host Bits to Create Subnet Bits

To subnet a network, the designer thinks about the network and host parts, as shown in Figure 11-15, and then the engineer adds a third part in the middle: the subnet part. However, the designer cannot change the size of the network part or the size of the entire address (32 bits). To create a subnet part of the address structure, the engineer borrows bits from the host part. Figure 11-15 shows the general idea.

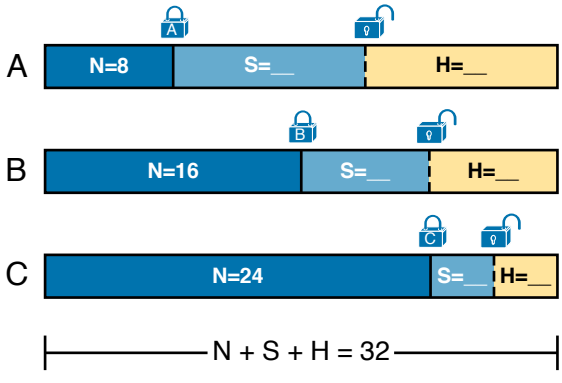


Figure 11-15 *Concept of Borrowing Host Bits*

Figure 11-15 shows a rectangle that represents the subnet mask. N, representing the number of network bits, remains locked at 8, 16, or 24, depending on the class. Conceptually, the designer moves a (dashed) dividing line into the host field, with subnet bits (S) between the

network and host parts, and the remaining host bits (H) on the right. The three parts must add up to 32 because IPv4 addresses consist of 32 bits.

Choosing Enough Subnet and Host Bits

The design process requires a choice of where to place the dashed line shown in Figure 11-15. But what is the right choice? How many subnet and host bits should the designer choose? The answers hinge on the requirements gathered in the early stages of the planning process:

- Number of subnets required
- Number of hosts/subnet

The bits in the subnet part create a way to uniquely number the different subnets that the design engineer wants to create. With 1 subnet bit, you can number 2^1 or 2 subnets. With 2 bits, 2^2 or 4 subnets, with 3 bits, 2^3 or 8 subnets, and so on. The number of subnet bits must be large enough to uniquely number all the subnets, as determined during the planning process.

At the same time, the remaining number of host bits must also be large enough to number the host IP addresses in the largest subnet. Remember, in this chapter, we assume the use of a single mask for all subnets. This single mask must support both the required number of subnets and the required number of hosts in the largest subnet. Figure 11-16 shows the concept.

Key Topic

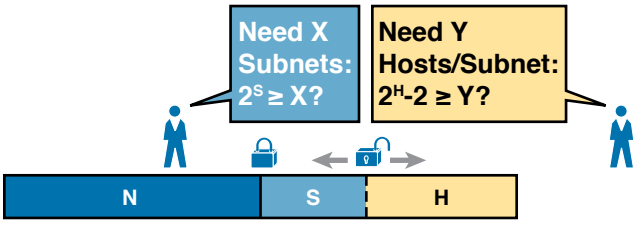


Figure 11-16 *Borrowing Enough Subnet and Host Bits*

Figure 11-16 shows the idea of the designer choosing a number of subnet (S) and host (H) bits and then checking the math. 2^S must be more than the number of required subnets, or the mask will not supply enough subnets in this IP network. Also, $2^H - 2$ must be more than the required number of hosts/subnet.

NOTE The idea of calculating the number of subnets as 2^S applies only in cases where a single mask is used for all subnets of a single classful network, as is being assumed in this chapter.

To effectively design masks, or to interpret masks that were chosen by someone else, you need a good working memory of the powers of 2. Appendix A, "Numeric Reference Tables," lists a table with powers of 2 up through 2^{32} for your reference.

Example Design: 172.16.0.0, 200 Subnets, 200 Hosts

To help make sense of the theoretical discussion so far, consider an example that focuses on the design choice for the subnet mask. In this case, the planning and design choices so far tell us the following:

- Use a single mask for all subnets.
- Plan for 200 subnets.
- Plan for 200 host IP addresses per subnet.
- Use private Class B network 172.16.0.0.

To choose the mask, the designer asks this question:

How many subnet (S) bits do I need to number 200 subnets?

You can see that S = 7 is not large enough ($2^7 = 128$), but S = 8 is enough ($2^8 = 256$). So, you need *at least* 8 subnet bits.

Next, the designer asks a similar question, based on the number of hosts per subnet:

How many host (H) bits do I need to number 200 hosts per subnet?

The math is basically the same, but the formula subtracts 2 when counting the number of hosts/subnet. You can see that H = 7 is not large enough ($2^7 - 2 = 126$), but H = 8 is enough ($2^8 - 2 = 254$).

Only one possible mask meets all the requirements in this case. First, the number of network bits (N) must be 16 because the design uses a Class B network. The requirements tell us that the mask needs at least 8 subnet bits and at least 8 host bits. The mask only has 32 bits in it; Figure 11-17 shows the resulting mask.

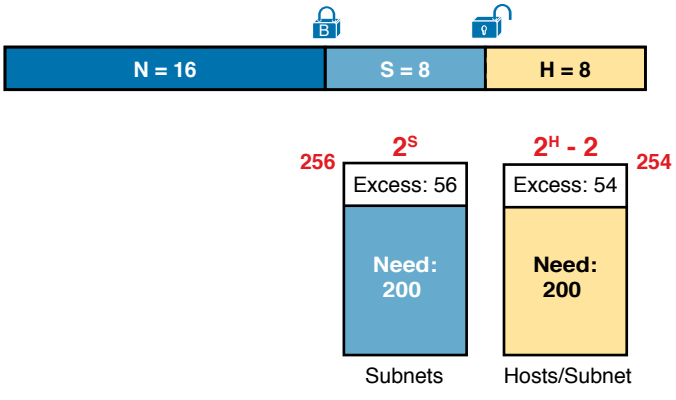


Figure 11-17 *Example Mask Choice, N = 16, S = 8, H = 8*

Masks and Mask Formats

Although engineers think about IP addresses in three parts when making design choices (network, subnet, and host), the subnet mask gives the engineer a way to communicate those design choices to all the devices in the subnet.

The subnet mask is a 32-bit binary number with a number of binary 1s on the left and with binary 0s on the right. By definition, the number of binary 0s equals the number of host bits; in fact, that is exactly how the mask communicates the idea of the size of the host part of the addresses in a subnet. The beginning bits in the mask equal binary 1, with those bit positions representing the combined network and subnet parts of the addresses in the subnet.

Because the network part always comes first, then the subnet part, and then the host part, the subnet mask, in binary form, cannot have interleaved 1s and 0s. Each subnet mask has one unbroken string of binary 1s on the left, with the rest of the bits as binary 0s.

After the engineer chooses the classful network and the number of subnet and host bits in a subnet, creating the binary subnet mask is easy. Just write down N 1s, S 1s, and then H 0s (assuming that N, S, and H represent the number of network, subnet, and host bits). Figure 11-18 shows the mask based on the previous example, which subnets a Class B network by creating 8 subnet bits, leaving 8 host bits.

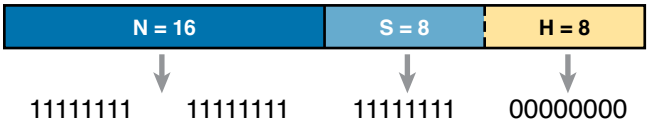


Figure 11-18 *Creating the Subnet Mask—Binary—Class B Network*

In addition to the binary mask shown in Figure 11-18, masks can also be written in two other formats: the familiar *dotted-decimal notation* (DDN) seen in IP addresses and an even briefer *prefix* notation. Chapter 13, "Analyzing Subnet Masks," discusses these formats and how to convert between the different formats.

Build a List of All Subnets

Building a list of all subnets, the final task of the subnet design step, determines the actual subnets that can be used, based on all the earlier choices. The earlier design work determined the Class A, B, or C network to use, and the (one) subnet mask to use that supplies enough subnets and enough host IP addresses per subnet. But what are those subnets? How do you identify or describe a subnet? This section answers these questions.

A subnet consists of a group of consecutive numbers. Most of these numbers can be used as IP addresses by hosts. However, each subnet reserves the first and last numbers in the group, and these two numbers cannot be used as IP addresses. In particular, each subnet contains the following:

Key Topic

- **Subnet number:** Also called the *subnet ID* or *subnet address*, this number identifies the subnet. It is the numerically smallest number in the subnet. It cannot be used as an IP address by a host.
- **Subnet broadcast:** Also called the *subnet broadcast address* or *directed broadcast address*, this is the last (numerically highest) number in the subnet. It also cannot be used as an IP address by a host.
- **IP addresses:** All the numbers between the subnet ID and the subnet broadcast address can be used as a host IP address.

For example, consider the earlier case in which the design results were as follows:

Network 172.16.0.0 (Class B)

Mask 255.255.255.0 (for all subnets)

With some math, the facts about each subnet that exists in this Class B network can be calculated. In this case, Table 11-3 shows the first 10 such subnets. It then skips many subnets and shows the last two (numerically largest) subnets.

Table 11-3 First 10 Subnets, Plus the Last Few, from 172.16.0.0, 255.255.255.0

Subnet Number	IP Addresses	Broadcast Address
172.16.0.0	172.16.0.1 – 172.16.0.254	172.16.0.255
172.16.1.0	172.16.1.1 – 172.16.1.254	172.16.1.255
172.16.2.0	172.16.2.1 – 172.16.2.254	172.16.2.255
172.16.3.0	172.16.3.1 – 172.16.3.254	172.16.3.255
172.16.4.0	172.16.4.1 – 172.16.4.254	172.16.4.255
172.16.5.0	172.16.5.1 – 172.16.5.254	172.16.5.255
172.16.6.0	172.16.6.1 – 172.16.6.254	172.16.6.255
172.16.7.0	172.16.7.1 – 172.16.7.254	172.16.7.255
172.16.8.0	172.16.8.1 – 172.16.8.254	172.16.8.255
172.16.9.0	172.16.9.1 – 172.16.9.254	172.16.9.255
Skipping many…		
172.16.254.0	172.16.254.1 – 172.16.254.254	172.16.254.255
172.16.255.0	172.16.255.1 – 172.16.255.254	172.16.255.255

After you have the network number and the mask, calculating the subnet IDs and other details for all subnets requires some math. In real life, most people use subnet calculators or subnet-planning tools. For the CCNA exam, you need to be ready to find this kind of information.

If you want to dig a little deeper in preparation for CCNP Enterprise or other studies related to IP routing, consider using Appendix L, "Subnet Design," on the book's companion website, which shows you how to find all the subnets of a given network.

Plan the Implementation

The next step, planning the implementation, is the last step before actually configuring the devices to create a subnet. The engineer first needs to choose where to use each subnet. For example, at a branch office in a particular city, which subnet from the subnet planning chart (Table 11-3) should be used for each VLAN at that site? Also, for any interfaces that require static IP addresses, which addresses should be used in each case? Finally, what range of IP addresses from inside each subnet should be configured in the DHCP server, to be

dynamically leased to hosts for use as their IP address? Figure 11-19 summarizes the list of implementation planning tasks.

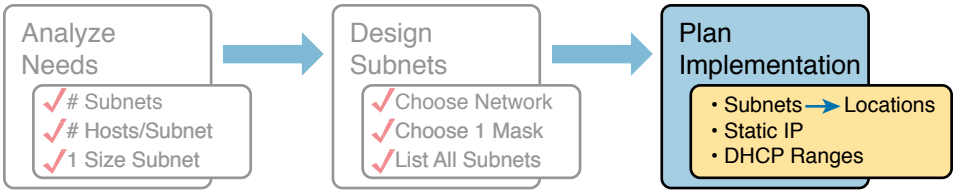


Figure 11-19 *Facts Supplied to the Plan Implementation Step*

Assigning Subnets to Different Locations

The job is simple: Look at your network diagram, identify each location that needs a subnet, and pick one from the table you made of all the possible subnets. Then, track it so that you know which ones you use where, using a spreadsheet or some other purpose-built subnet-planning tool. That's it! Figure 11-20 shows a sample of a completed design using Table 11-3, which happens to match the initial design sample shown way back in Figure 11-1.

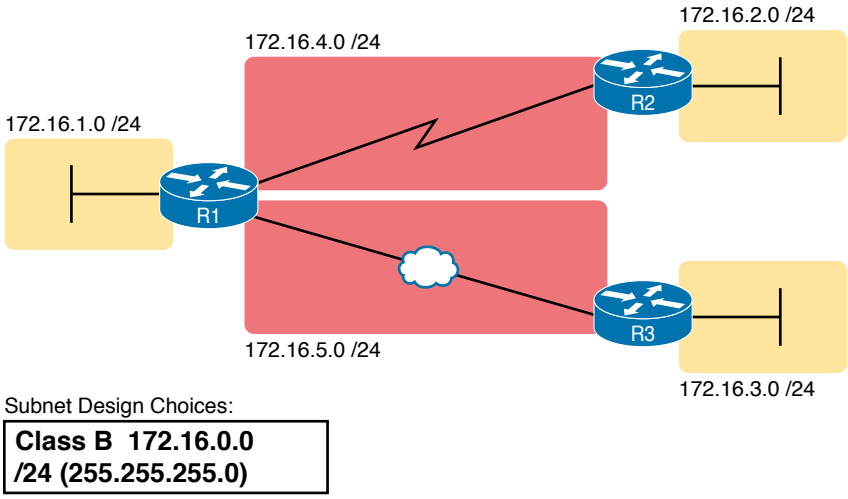


Figure 11-20 *Example of Subnets Assigned to Different Locations*

Although this design could have used any five subnets from Table 11-3, in real networks, engineers usually give more thought to some strategy for assigning subnets. For example, you might assign all LAN subnets lower numbers and WAN subnets higher numbers. Or you might slice off large ranges of subnets for different divisions of the company. Or you might follow that same strategy but ignore organizational divisions in the company, paying more attention to geographies.

For example, for a U.S.-based company with a smaller presence in both Europe and Asia, you might plan to reserve ranges of subnets based on continent. This kind of choice is particularly useful when later trying to use a feature called route summarization. Figure 11-21 shows the general benefit of placing addressing in the network for easier route summarization, using the same subnets from Table 11-3 again.

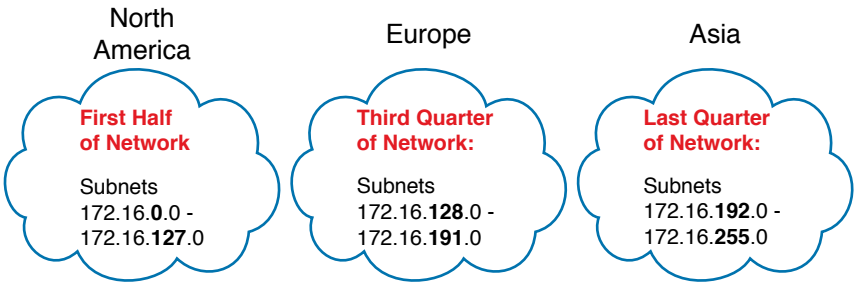


Figure 11-21 *Reserving 50 Percent of Subnets for North America and 25 Percent Each for Europe and Asia*

Choose Static and Dynamic Ranges per Subnet

Devices receive their IP address and mask assignment in one of two ways: dynamically by using Dynamic Host Configuration Protocol (DHCP) or statically through configuration. For DHCP to work, the network engineer must tell the DHCP server the subnets for which it must assign IP addresses. In addition, that configuration limits the DHCP server to only a subset of the addresses in the subnet. For static addresses, you simply configure the device to tell it what IP address and mask to use.

To keep things as simple as possible, most shops use a strategy to separate the static IP addresses on one end of each subnet, and the DHCP-assigned dynamic addresses on the other. It does not really matter whether the static addresses sit on the low end of the range of addresses or the high end.

For example, imagine that the engineer decides that, for the LAN subnets in Figure 11-20, the DHCP pool comes from the high end of the range, namely, addresses that end in .101 through .254. (The address that ends in .255 is, of course, reserved.) The engineer also assigns static addresses from the lower end, with addresses ending in .1 through .100. Figure 11-22 shows the idea.

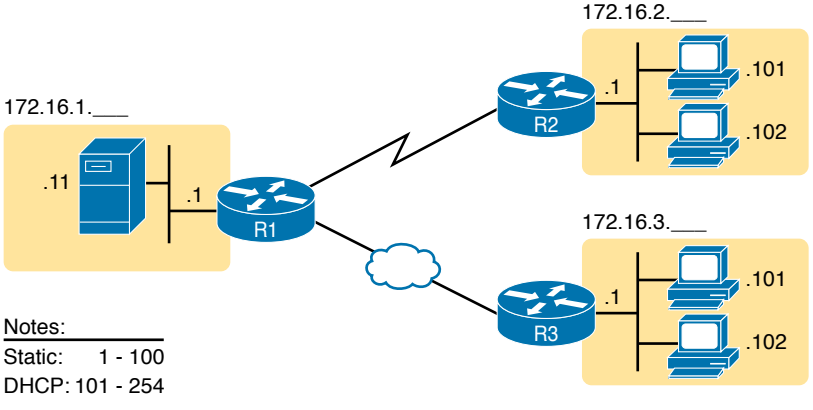


Figure 11-22 *Static from the Low End and DHCP from the High End*

Figure 11-22 shows all three routers with statically assigned IP addresses that end in .1. The only other static IP address in the figure is assigned to the server on the left, with address 172.16.1.11 (abbreviated simply as .11 in the figure).

On the right, each LAN has two PCs that use DHCP to dynamically lease their IP addresses. DHCP servers often begin by leasing the addresses at the bottom of the range of addresses, so in each LAN, the hosts have leased addresses that end in .101 and .102, which are at the low end of the range chosen by design.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 11-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 11-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Key Topic

Table 11-5 Key Topics for Chapter 11

Key Topic Element	Description	Page Number
List	Key facts about subnets	269
List	Rules about what places in a network topology need a subnet	270
Figure 11-7	Locations of the network, subnet, and host parts of an IPv4 address	273
List	Features that extended the life of IPv4	277
List	Motivations for using private IP networks	278
Figure 11-14	Formats of Class A, B, and C addresses when not subnetted	280
Figure 11-15	Formats of Class A, B, and C addresses when subnetted	280
Figure 11-16	General logic when choosing the size of the subnet and host parts of addresses in a subnet	281
List	Items that together define a subnet	283

Key Terms You Should Know

subnet, network, classful IP network, variable-length subnet masks (VLSM), network part, subnet part, host part, public IP network, private IP network, subnet mask

CHAPTER 12

Analyzing Classful IPv4 Networks

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

When operating a network, you often start investigating a problem based on an IP address and mask. Based on the IP address alone, you should be able to determine several facts about the Class A, B, or C network in which the IP address resides.

This chapter lists the key facts about classful IP networks and explains how to discover these facts. Following that, this chapter lists some practice problems. Before moving to the next chapter, you should practice until you can consistently determine all these facts, quickly and confidently, based on an IP address.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 12-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Classful Network Concepts	1–5

1. Which of the following are not valid Class A network IDs? (Choose two answers.)
   a. 1.0.0.0
   b. 130.0.0.0
   c. 127.0.0.0
   d. 9.0.0.0
2. Which of the following are not valid Class B network IDs?
   a. 130.0.0.0
   b. 191.255.0.0
   c. 128.0.0.0
   d. 150.255.0.0
   e. All are valid Class B network IDs.

3. Which of the following are true about IP address 172.16.99.45's IP network? (Choose two answers.)
   a. The network ID is 172.0.0.0.
   b. The network is a Class B network.
   c. The default mask for the network is 255.255.255.0.
   d. The number of host bits in the unsubnetted network is 16.
4. Which of the following are true about IP address 192.168.6.7's IP network? (Choose two answers.)
   a. The network ID is 192.168.6.0.
   b. The network is a Class B network.
   c. The default mask for the network is 255.255.255.0.
   d. The number of host bits in the unsubnetted network is 16.
5. Which of the following is a network broadcast address?
   a. 10.1.255.255
   b. 192.168.255.1
   c. 224.1.1.255
   d. 172.30.255.255

Foundation Topics

Classful Network Concepts

Imagine that you have a job interview for your first IT job. As part of the interview, you're given an IPv4 address and mask: 10.4.5.99, 255.255.255.0. What can you tell the interviewer about the classful network (in this case, the Class A network) in which the IP address resides?

This section, the first of two major sections in this chapter, reviews the concepts of *classful IP networks* (in other words, Class A, B, and C networks). In particular, this chapter examines how to begin with a single IP address and then determine the following facts:

- Class (A, B, or C)
- Default mask
- Number of network octets/bits
- Number of host octets/bits
- Number of host addresses in the network
- Network ID
- Network broadcast address
- First and last usable address in the network

IPv4 Network Classes and Related Facts

IP version 4 (IPv4) defines five address classes. Three of the classes, Classes A, B, and C, consist of unicast IP addresses. Unicast addresses identify a single host or interface so that the address uniquely identifies the device. Class D addresses serve as multicast addresses, so that one packet sent to a Class D multicast IPv4 address can actually be delivered to multiple hosts. Finally, Class E addresses were originally intended for experimentation but were changed to simply be reserved for future use. The class can be identified based on the value of the first octet of the address, as shown in Table 12-2.

Key Topic

Table 12-2 IPv4 Address Classes Based on First Octet Values

Class	First Octet Values	Purpose
A	1–126	Unicast (large networks)
B	128–191	Unicast (medium-sized networks)
C	192–223	Unicast (small networks)
D	224–239	Multicast
E	240–255	Reserved (formerly experimental)

After you identify the class of a unicast address as either A, B, or C, many other related facts can be derived just through memorization. Table 12-3 lists that information for reference and later study; each of these concepts is described in this chapter.

Key Topic

Table 12-3 Key Facts for Classes A, B, and C

	Class A	Class B	Class C
First octet range	1–126	128–191	192–223
Valid network numbers	1.0.0.0–126.0.0.0	128.0.0.0–191.255.0.0	192.0.0.0–223.255.255.0
Total networks	$2^7 - 2 = 126$	$2^{14} = 16{,}384$	$2^{21} = 2{,}097{,}152$
Hosts per network	$2^{24} - 2$	$2^{16} - 2$	$2^8 - 2$
Octets (bits) in network part	1 (8)	2 (16)	3 (24)
Octets (bits) in host part	3 (24)	2 (16)	1 (8)
Default mask	255.0.0.0	255.255.0.0	255.255.255.0

Note that the address ranges of all addresses that begin with 0 and all addresses that begin with 127 are reserved. Had they not been reserved since the creation of Class A networks, as listed in RFC 791 (published in 1981), then they might have been known as class A networks 0.0.0.0 and 127.0.0.0. Because they are reserved, however, the address space has 126 class A networks, and not 128. Also, note that there are no similar reserved ranges to begin/end the class B and C ranges.

Answers to the "Do I Know This Already?" quiz:

1 B, C **2** E **3** B, D **4** A, C **5** D

In addition to the reservation of what would be class A networks 0.0.0.0 and 127.0.0.0 for other purposes, other newer RFCs have also reserved small pieces of the Class A, B, and C address space. So, tables like Table 12-3, with the count of the numbers of Class A, B, and C networks, are a good place to get a sense of the size of the number; however, the number of reserved networks does change slightly over time (albeit slowly) based on these other reserved address ranges.

NOTE If you are interested in seeing all the reserved IPv4 address ranges, just do an Internet search on "IANA IPv4 special-purpose address registry."

The Number and Size of the Class A, B, and C Networks

Table 12-3 lists the range of Class A, B, and C network numbers; however, some key points can be lost just referencing a table of information. This section examines the Class A, B, and C network numbers, focusing on the more important points and the exceptions and unusual cases.

First, the number of networks from each class significantly differs. Only 126 Class A networks exist: network 1.0.0.0, 2.0.0.0, 3.0.0.0, and so on, up through network 126.0.0.0. However, 16,384 Class B networks exist, with more than 2 million Class C networks.

Next, note that the size of networks from each class also significantly differs. Each Class A network is relatively large—over 16 million host IP addresses per network—so they were originally intended to be used by the largest companies and organizations. Class B networks are smaller, with over 65,000 hosts per network. Finally, Class C networks, intended for small organizations, have 254 hosts in each network. Figure 12-1 summarizes those facts.

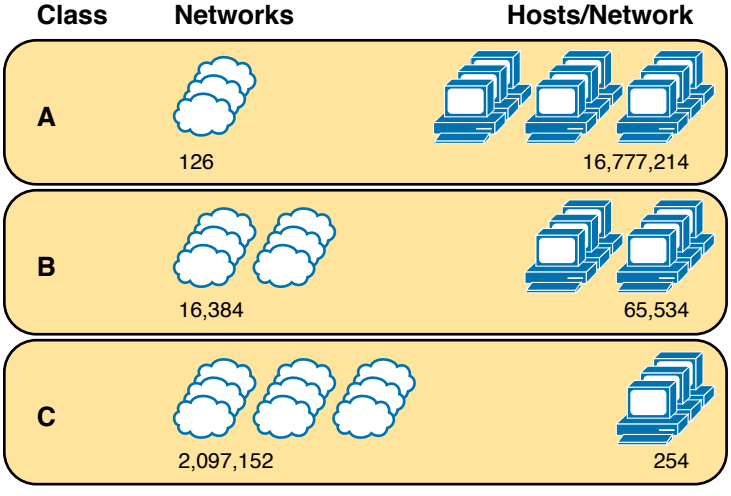


Figure 12-1 *Numbers and Sizes of Class A, B, and C Networks*

Address Formats

In some cases, an engineer might need to think about a Class A, B, or C network as if the network has not been subdivided through the subnetting process. In such a case, the

addresses in the classful network have a structure with two parts: the *network part* (sometimes called the *prefix*) and the *host part*. Then, comparing any two IP addresses in one network, the following observations can be made:

Key Topic

The addresses in the same network have the same values in the network part.

The addresses in the same network have different values in the host part.

For example, in Class A network 10.0.0.0, by definition, the network part consists of the first octet. As a result, all addresses have an equal value in the network part, namely a 10 in the first octet. If you then compare any two addresses in the network, the addresses have a different value in the last three octets (the host octets). For example, IP addresses 10.1.1.1 and 10.1.1.2 have the same value (10) in the network part, but different values in the host part.

Figure 12-2 shows the format and sizes (in number of bits) of the network and host parts of IP addresses in Class A, B, and C networks, before any subnetting has been applied.

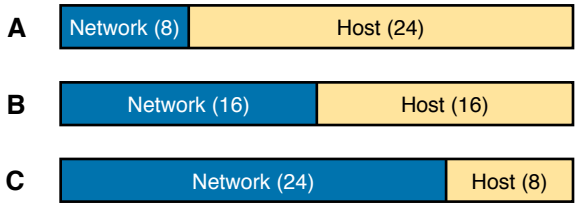


Figure 12-2 *Sizes (Bits) of the Network and Host Parts of Unsubnetted Classful Networks*

Default Masks

Although we humans can easily understand the concepts behind Figure 12-2, computers prefer numbers. To communicate those same ideas to computers, each network class has an associated *default mask* that defines the size of the network and host parts of an unsubnetted Class A, B, and C network. To do so, the mask lists binary 1s for the bits considered to be in the network part and binary 0s for the bits considered to be in the host part.

For example, Class A network 10.0.0.0 has a network part of the first single octet (8 bits) and a host part of the last three octets (24 bits). As a result, the Class A default mask is 255.0.0.0, which in binary is

11111111 00000000 00000000 00000000

Figure 12-3 shows default masks for each network class, both in binary and dotted-decimal format.

NOTE Decimal 255 converts to the binary value 11111111. Decimal 0, converted to 8-bit binary, is 00000000. See Appendix A, "Numeric Reference Tables," for a conversion table.

Key Topic

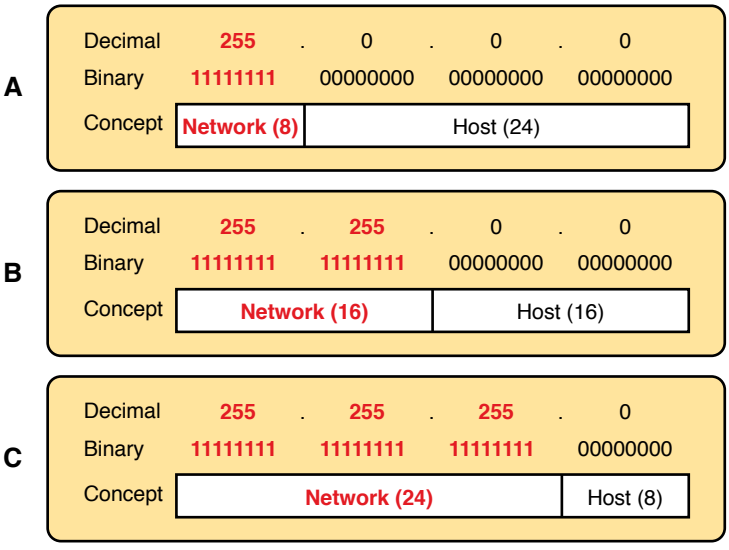


Figure 12-3 *Default Masks for Classes A, B, and C*

Number of Hosts per Network

Calculating the number of hosts per network requires some basic binary math. First, consider a case where you have a single binary digit. How many unique values are there? There are, of course, two values: 0 and 1. With 2 bits, you can make four combinations: 00, 01, 10, and 11. As it turns out, the total combination of unique values you can make with N bits is 2^N.

Host addresses—the IP addresses assigned to hosts—must be unique. The host bits exist for the purpose of giving each host a unique IP address by virtue of having a different value in the host part of the addresses. So, with H host bits, 2^H unique combinations exist.

However, the number of hosts in a network is not 2^H; instead, it is $2^H - 2$. Each network reserves two numbers that would have otherwise been useful as host addresses but have instead been reserved for special use: one for the network ID and one for the network broadcast address. As a result, the formula to calculate the number of host addresses per Class A, B, or C network is

Key Topic

$$2^H - 2$$

where H is the number of host bits.

Deriving the Network ID and Related Numbers

Each classful network has four key numbers that describe the network. You can derive these four numbers if you start with just one IP address in the network. The numbers are as follows:

- Network number
- First (numerically lowest) usable address
- Last (numerically highest) usable address
- Network broadcast address

First, consider both the network number and first usable IP address. The *network number*, also called the *network ID* or *network address*, identifies the network. By definition, the network number is the numerically lowest number in the network. However, to prevent any ambiguity, the people that made up IP addressing added the restriction that the network number cannot be assigned as an IP address. So, the lowest number in the network is the network ID. Then, the first (numerically lowest) host IP address is *one larger than* the network number.

Next, consider the network broadcast address along with the last (numerically highest) usable IP address. The TCP/IP RFCs define a network broadcast address as a special address in each network. This broadcast address could be used as the destination address in a packet, and the routers would forward a copy of that one packet to all hosts in that classful network. Numerically, a network broadcast address is always the highest (last) number in the network. As a result, the highest (last) number usable as an IP address is the address that is *one less than* the network broadcast address.

Simply put, if you can find the network number and network broadcast address, finding the first and last usable IP addresses in the network is easy. For the exam, you should be able to find all four values with ease; the process is as follows:

Key Topic

Step 1. Determine the class (A, B, or C) based on the first octet.

Step 2. Mentally divide the network and host octets based on the class.

Step 3. To find the network number, change the IP address's host octets to 0.

Step 4. To find the first address, add 1 to the fourth octet of the network ID.

Step 5. To find the broadcast address, change the network ID's host octets to 255.

Step 6. To find the last address, subtract 1 from the fourth octet of the network broadcast address.

The written process actually looks harder than it is. Figure 12-4 shows an example of the process, using Class A IP address 10.17.18.21, with the circled numbers matching the process.

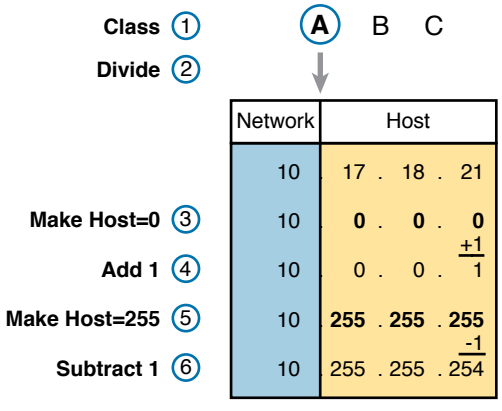


Figure 12-4 *Example of Deriving the Network ID and Other Values from 10.17.18.21*

Figure 12-4 shows the identification of the class as Class A (Step 1) and the number of network/host octets as 1 and 3, respectively. So, to find the network ID at Step 3, the figure copies only the first octet, setting the last three (host) octets to 0. At Step 4, just copy the network ID and add 1 to the fourth octet. Similarly, to find the broadcast address at Step 5, copy the network octets, but set the host octets to 255. Then, at Step 6, subtract 1 from the fourth octet to find the last (numerically highest) usable IP address.

Just to show an alternative example, consider IP address 172.16.8.9. Figure 12-5 shows the process applied to this IP address.

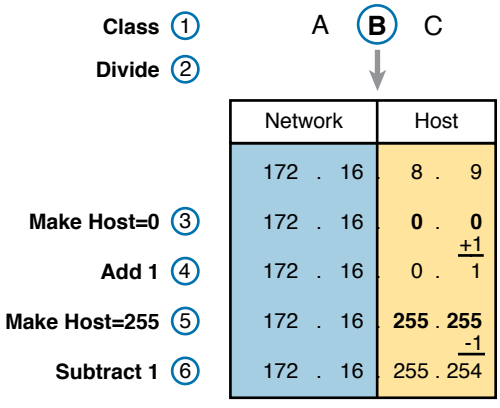


Figure 12-5 *Example Deriving the Network ID and Other Values from 172.16.8.9*

Figure 12-5 shows the identification of the class as Class B (Step 1) and the number of network/host octets as 2 and 2, respectively. So, to find the network ID at Step 3, the figure copies only the first two octets, setting the last two (host) octets to 0. Similarly, Step 5 shows the same action, but with the last two (host) octets being set to 255.

Unusual Network IDs and Network Broadcast Addresses

Some of the more unusual numbers in and around the range of Class A, B, and C network numbers can cause some confusion. This section lists some examples of numbers that make many people make the wrong assumptions about the meaning of the number.

For Class A, the first odd fact is that the range of values in the first octet omits the numbers 0 and 127. As it turns out, what would be Class A network 0.0.0.0 was originally reserved for some broadcasting requirements, so all addresses that begin with 0 in the first octet are reserved. What would be Class A network 127.0.0.0 is still reserved because of a special address used in software testing, called the loopback address (127.0.0.1).

For Class B (and C), some of the network numbers can look odd, particularly if you fall into a habit of thinking that 0s at the end means the number is a network ID, and 255s at the end means it's a network broadcast address. First, Class B network numbers range from 128.0.0.0 to 191.255.0.0, for a total of 2^{14} networks. However, even the very first (lowest number) Class B network number (128.0.0.0) looks a little like a Class A network number because it ends with three 0s. However, the first octet is 128, making it a Class B network with a two-octet network part (128.0).

For another Class B example, the high end of the Class B range also might look strange at first glance (191.255.0.0), but this is indeed the numerically highest of the valid Class B network numbers. This network's broadcast address, 191.255.255.255, might look a little like a Class A broadcast address because of the three 255s at the end, but it is indeed the broadcast address of a Class B network.

Similarly to Class B networks, some of the valid Class C network numbers do look strange. For example, Class C network 192.0.0.0 looks a little like a Class A network because of the last three octets being 0, but because it is a Class C network, it consists of all addresses that begin with three octets equal to 192.0.0. Similarly, 223.255.255.0, another valid Class C network, consists of all addresses that begin with 223.255.255.

Practice with Classful Networks

As with all areas of IP addressing and subnetting, you need to practice to be ready for the CCNA exam. You should practice some while reading this chapter to make sure that you understand the processes. At that point, you can use your notes and this book as a reference, with a goal of understanding the process. After that, keep practicing this and all the other subnetting processes. Before you take the exam, you should be able to always get the right answer, and with speed. Table 12-4 summarizes the key concepts and suggestions for this two-phase approach.

Table 12-4 Keep-Reading and Take-Exam Goals for This Chapter's Topics

	After Reading This Chapter	Before Taking the Exam
Focus on…	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	10 seconds

Practice Deriving Key Facts Based on an IP Address

Practice finding the various facts that can be derived from an IP address, as discussed throughout this chapter. To do so, complete Table 12-5.

Table 12-5 Practice Problems: Find the Network ID and Network Broadcast

	IP Address	Class	Network Octets	Host Octets	Network ID	Network Broadcast Address
1	1.1.1.1					
2	128.1.6.5					
3	200.1.2.3					
4	192.192.1.1					
5	126.5.4.3					

	IP Address	Class	Network Octets	Host Octets	Network ID	Network Broadcast Address
6	200.1.9.8					
7	192.0.0.1					
8	191.255.1.47					
9	223.223.0.1					

The answers are listed in the section "Answers to Earlier Practice Problems," later in this chapter.

Practice Remembering the Details of Address Classes

Tables 12-2 and 12-3, shown earlier in this chapter, summarized some key information about IPv4 address classes. Tables 12-6 and 12-7 show sparse versions of these same tables. To practice recalling those key facts, particularly the range of values in the first octet that identifies the address class, complete these tables. Then, refer to Tables 12-2 and 12-3 to check your answers. Repeat this process until you can recall all the information in the tables.

Table 12-6 Sparse Study Table Version of Table 12-2

Class	First Octet Values	Purpose
A		
B		
C		
D		
E		

Table 12-7 Sparse Study Table Version of Table 12-3

	Class A	Class B	Class C
First octet range			
Valid network numbers			
Total networks			
Hosts per network			
Octets (bits) in network part			
Octets (bits) in host part			
Default mask			

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 12-8 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 12-8 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website
Practice analyzing classful IPv4 networks		Website, Appendix D

Review All the Key Topics

Key Topic

Table 12-9 Key Topics for Chapter 12

Key Topic Elements	Description	Page Number
Table 12-2	Address classes	290
Table 12-3	Key facts about Class A, B, and C networks	290
List	Comparisons of network and host parts of addresses in the same classful network	292
Figure 12-3	Default masks	293
Paragraph	Function to calculate the number of hosts per network	294
List	Steps to find information about a classful network	294

Key Terms You Should Know

network, classful IP network, network number, network ID, network address, network broadcast address, network part, host part, default mask

Additional Practice for This Chapter's Processes

For additional practice with analyzing classful networks, you may do a set of practice problems using your choice of tools:

Application: Use the Analyzing Classful IPv4 Networks application on the companion website.

PDF: Alternatively, practice the same problems using companion website Appendix D, "Practice for Chapter 12: Analyzing Classful IPv4 Networks."

Answers to Earlier Practice Problems

Table 12-5, shown earlier, listed several practice problems. Table 12-10 lists the answers.

Table 12-10 Practice Problems: Find the Network ID and Network Broadcast

	IP Address	Class	Network Octets	Host Octets	Network ID	Network Broadcast
1	1.1.1.1	A	1	3	1.0.0.0	1.255.255.255
2	128.1.6.5	B	2	2	128.1.0.0	128.1.255.255
3	200.1.2.3	C	3	1	200.1.2.0	200.1.2.255
4	192.192.1.1	C	3	1	192.192.1.0	192.192.1.255
5	126.5.4.3	A	1	3	126.0.0.0	126.255.255.255
6	200.1.9.8	C	3	1	200.1.9.0	200.1.9.255
7	192.0.0.1	C	3	1	192.0.0.0	192.0.0.255
8	191.255.1.47	B	2	2	191.255.0.0	191.255.255.255
9	223.223.0.1	C	3	1	223.223.0.0	223.223.0.255

The class, number of network octets, and number of host octets all require you to look at the first octet of the IP address to determine the class. If a value is between 1 and 126, inclusive, the address is a Class A address, with one network and three host octets. If a value is between 128 and 191 inclusive, the address is a Class B address, with two network and two host octets. If a value is between 192 and 223, inclusive, it is a Class C address, with three network octets and one host octet.

The last two columns can be found based on Table 12-3, specifically the number of network and host octets along with the IP address. To find the network ID, copy the IP address, but change the host octets to 0. Similarly, to find the network broadcast address, copy the IP address, but change the host octets to 255.

The last three problems can be confusing and were included on purpose so that you could see an example of these unusual cases, as follows.

Answers to Practice Problem 7 (from Table 12-5)

Consider IP address 192.0.0.1. First, 192 is on the lower edge of the first octet range for Class C; as such, this address has three network and one host octet. To find the network ID, copy the address, but change the single host octet (the fourth octet) to 0, for a network ID of 192.0.0.0. It looks strange, but it is indeed the network ID.

The network broadcast address choice for problem 7 can also look strange. To find the broadcast address, copy the IP address (192.0.0.1), but change the last octet (the only host octet) to 255, for a broadcast address of 192.0.0.255. In particular, if you decide that the broadcast should be 192.255.255.255, you might have fallen into the trap of logic, like "Change all 0s in the network ID to 255s," which is not the correct logic. Instead, change all host octets in the IP address (or network ID) to 255s.

Answers to Practice Problem 8 (from Table 12-5)

The first octet of problem 8 (191.255.1.47) sits on the upper edge of the Class B range for the first octet (128–191). As such, to find the network ID, change the last two octets (host octets) to 0, for a network ID of 191.255.0.0. This value sometimes gives people problems because they are used to thinking that 255 somehow means the number is a broadcast address.

The broadcast address, found by changing the two host octets to 255, means that the broadcast address is 191.255.255.255. It looks more like a broadcast address for a Class A network, but it is actually the broadcast address for Class B network 191.255.0.0.

Answers to Practice Problem 9 (from Table 12-5)

Problem 9, with IP address 223.223.0.1, is near the high end of the Class C range. As a result, only the last (host) octet is changed to 0 to form the network ID 223.223.0.0. It looks a little like a Class B network number at first glance because it ends in two octets of 0. However, it is indeed a Class C network ID (based on the value in the first octet).

This page intentionally left blank

CHAPTER 13

Analyzing Subnet Masks

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

The subnet mask used in one or many subnets in an IP internetwork says a lot about the intent of the subnet design. First, the mask divides addresses into two parts: *prefix* and *host*, with the host part defining the size of the subnet. Then, the class (A, B, or C) further divides the structure of addresses in a subnet, breaking the prefix part into the *network* and *subnet* parts. The subnet part defines the number of subnets that could exist inside one classful IP network, assuming that one mask is used throughout the classful network.

The subnet mask holds the key to understanding several important subnetting design points. However, to analyze a subnet mask, you first need some basic math skills with masks. The math converts masks between the three different formats used to represent a mask:

- Binary
- Dotted-decimal notation (DDN)
- Prefix (also called classless interdomain routing [CIDR])

This chapter has two major sections. The first focuses on the mask formats and the math used to convert between the three formats. The second section explains how to take an IP address and its subnet mask and analyze those values. In particular, it shows how to determine the three-part format of the IPv4 address and describes the facts about the subnetting design that are implied by the mask.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 13-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Subnet Mask Conversion	1–3
Defining the Format of IPv4 Addresses	4–7

1. Which of the following answers lists the prefix (CIDR) format equivalent of 255.255.254.0?
   - **a.** /19
   - **b.** /20
   - **c.** /23
   - **d.** /24
   - **e.** /25
2. Which of the following answers lists the prefix (CIDR) format equivalent of 255.255.255.240?
   - **a.** /26
   - **b.** /28
   - **c.** /27
   - **d.** /30
   - **e.** /29
3. Which of the following answers lists the dotted-decimal notation (DDN) equivalent of /30?
   - **a.** 255.255.255.192
   - **b.** 255.255.255.252
   - **c.** 255.255.255.240
   - **d.** 255.255.254.0
   - **e.** 255.255.255.0
4. Working at the help desk, you receive a call and learn a user's PC IP address and mask (10.55.66.77, mask 255.255.255.0). When thinking about this using classful logic, you determine the number of network (N), subnet (S), and host (H) bits. Which of the following is true in this case?
   - **a.** N=12
   - **b.** S=12
   - **c.** H=8
   - **d.** S=8
   - **e.** N=24

5. Working at the help desk, you receive a call and learn a user's PC IP address and mask (192.168.9.1/27). When thinking about this using classful logic, you determine the number of network (N), subnet (S), and host (H) bits. Which of the following is true in this case?
   a. N=24
   b. S=24
   c. H=8
   d. H=7
6. Which of the following statements is true about classless IP addressing concepts?
   a. Uses a 128-bit IP address
   b. Applies only for Class A and B networks
   c. Separates IP addresses into network, subnet, and host parts
   d. Ignores Class A, B, and C network rules
7. Which of the following masks, when used as the only mask within a Class B network, would supply enough subnet bits to support 100 subnets? (Choose two.)
   a. /24
   b. 255.255.255.252
   c. /20
   d. 255.255.252.0

Foundation Topics

Subnet Mask Conversion

This section describes how to convert between different formats for the subnet mask. You can then use these processes when you practice. If you already know how to convert from one format to the other, go ahead and move to the section "Practice Converting Subnet Masks," later in this chapter.

Three Mask Formats

Subnet masks can be written as 32-bit binary numbers, but not just any binary number. In particular, the binary subnet mask must follow these rules:

- The value must not interleave 1s and 0s.
- If 1s exist, they are on the left.
- If 0s exist, they are on the right.

For example, the following values would be illegal. The first is illegal because the value interleaves 0s and 1s, and the second is illegal because it lists 0s on the left and 1s on the right:

```
10101010 01010101 11110000 00001111
00000000 00000000 00000000 11111111
```

The following two binary values meet the requirements, in that they have all 1s on the left, followed by all 0s, with no interleaving of 1s and 0s:

```
11111111 00000000 00000000 00000000
11111111 11111111 11111111 00000000
```

Two alternative subnet mask formats exist so that we humans do not have to work with 32-bit binary numbers. One format, dotted-decimal notation (DDN), converts each set of 8 bits into the decimal equivalent. For example, the two previous binary masks would convert to the following DDN subnet masks because binary 11111111 converts to decimal 255, and binary 00000000 converts to decimal 0:

255.0.0.0

255.255.255.0

Although the DDN format has been around since the beginning of IPv4 addressing, the third mask format was added later, in the early 1990s: the *prefix* format. This format takes advantage of the rule that the subnet mask starts with some number of 1s, and then the rest of the digits are 0s. Prefix format lists a slash (/) followed by the number of binary 1s in the binary mask. Using the same two examples as earlier in this section, the prefix format equivalent masks are as follows:

/8

/24

Note that although the terms *prefix* or *prefix mask* can be used, the terms *CIDR mask* or *slash mask* can also be used. This newer prefix style mask was created around the same time as the classless interdomain routing (CIDR) specification back in the early 1990s, and the acronym CIDR grew to be used for anything related to CIDR, including prefix-style masks. In addition, the term *slash mask* is sometimes used because the value includes a slash mark (/).

You need to get comfortable working with masks in different formats. The rest of this section examines how to convert between the three formats.

Converting Between Binary and Prefix Masks

Converting between binary and prefix masks should be relatively intuitive after you know that the prefix value is simply the number of binary 1s in the binary mask. For the sake of completeness, the processes to convert in each direction are

Binary to prefix: Count the number of binary 1s in the binary mask, and write the total, in decimal, after a /.

Prefix to binary: Write P binary 1s, where P is the prefix value, followed by as many binary 0s as required to create a 32-bit number.

Tables 13-2 and 13-3 show some examples.

Table 13-2 Example Conversions: Binary to Prefix

Binary Mask	Logic	Prefix Mask
11111111 11111111 11000000 00000000	Count 8 + 8 + 2 = 18 binary 1s	/18
11111111 11111111 11111111 11110000	Count 8 + 8 + 8 + 4 = 28 binary 1s	/28
11111111 11111000 00000000 00000000	Count 8 + 5 = 13 binary 1s	/13

Table 13-3 Example Conversions: Prefix to Binary

Prefix Mask	Logic	Binary Mask
/18	Write 18 1s, then 14 0s, total 32	11111111 11111111 11000000 00000000
/28	Write 28 1s, then 4 0s, total 32	11111111 11111111 11111111 11110000
/13	Write 13 1s, then 19 0s, total 32	11111111 11111000 00000000 00000000

Converting Between Binary and DDN Masks

By definition, a dotted-decimal number (DDN) used with IPv4 addressing contains four decimal numbers, separated by dots. Each decimal number represents 8 bits. So, a single DDN shows four decimal numbers that together represent some 32-bit binary number.

Conversion from a DDN mask to the binary equivalent is relatively simple to describe but can be laborious to perform. First, to do the conversion, the process is as follows:

For each octet, perform a decimal-to-binary conversion.

However, depending on your comfort level with doing decimal-to-binary conversions, that process can be difficult or time-consuming. If you want to think about masks in binary for the exam, consider picking one of the following methods to do the conversion and practicing until you can do it quickly and accurately:

- Do the decimal-binary conversions, but practice your decimal-binary conversions to become faster. If you choose this path, consider the Cisco Binary Game, which you can find by searching its name at the Cisco Learning Network (CLN) (http://learningnetwork.cisco.com).
- Use the decimal-binary conversion chart in Appendix A, "Numeric Reference Tables." This lets you find the answer more quickly now, but you cannot use the chart on exam day.
- Memorize the nine possible decimal values that can be in a decimal mask, and practice using a reference table with those values.

The third method, which is the method recommended in this book, takes advantage of the fact that any and every DDN mask octet must be one of only nine values. Why? Well, remember how a binary mask cannot interleave 1s and 0s, and the 0s must be on the right? It turns out that only nine different 8-bit binary numbers conform to these rules. Table 13-4 lists the values, along with other relevant information.

Answers to the "Do I Know This Already?" quiz:

1 C **2** B **3** B **4** C **5** A **6** D **7** A, B

Key Topic

Table 13-4 Nine Possible Values in One Octet of a Subnet Mask

Binary Mask Octet	Decimal Equivalent	Number of Binary 1s
00000000	0	0
10000000	128	1
11000000	192	2
11100000	224	3
11110000	240	4
11111000	248	5
11111100	252	6
11111110	254	7
11111111	255	8

Many subnetting processes can be done with or without binary math. Some of those processes—mask conversion included—use the information in Table 13-4. You should plan to memorize the information in the table. I recommend making a copy of the table to keep handy while you practice. (You will likely memorize the contents of this table simply by practicing the conversion process enough to get both good and fast at the conversion.)

Using the table, the conversion processes in each direction with binary and decimal masks are as follows:

Key Topic

Binary to decimal: Organize the bits into four sets of eight. For each octet, find the binary value in the table and write down the corresponding decimal value.

Decimal to binary: For each octet, find the decimal value in the table and write down the corresponding 8-bit binary value.

Tables 13-5 and 13-6 show some examples.

Table 13-5 Conversion Example: Binary to Decimal

Binary Mask	Logic	Decimal Mask
11111111 11111111 11000000 00000000	11111111 maps to 255 11000000 maps to 192 00000000 maps to 0	255.255.192.0
11111111 11111111 11111111 11110000	11111111 maps to 255 11110000 maps to 240	255.255.255.240
11111111 11111000 00000000 00000000	11111111 maps to 255 11111000 maps to 248 00000000 maps to 0	255.248.0.0

Table 13-6 Conversion Examples: Decimal to Binary

Decimal Mask	Logic	Binary Mask
255.255.192.0	255 maps to 11111111 192 maps to 11000000 0 maps to 00000000	11111111 11111111 11000000 00000000
255.255.255.240	255 maps to 11111111 240 maps to 11110000	11111111 11111111 11111111 11110000
255.248.0.0	255 maps to 11111111 248 maps to 11111000 0 maps to 00000000	11111111 11111000 00000000 00000000

Converting Between Prefix and DDN Masks

When you are learning, the best way to convert between the prefix and decimal formats is to first convert to binary. For example, to move from decimal to prefix, first convert decimal to binary and then from binary to prefix.

For the exams, set a goal to master these conversions doing the math in your head. While learning, you will likely want to use paper. To train yourself to do all this without writing it down, instead of writing each octet of binary, just write the number of binary 1s in that octet.

Figure 13-1 shows an example with a prefix-to-decimal conversion. The left side shows the conversion to binary as an interim step. For comparison, the right side shows the binary interim step in shorthand that just lists the number of binary 1s in each octet of the binary mask.

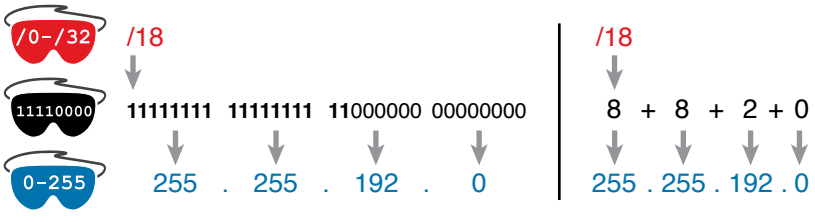


Figure 13-1 *Conversion from Prefix to Decimal: Full Binary Versus Shorthand*

Similarly, when converting from decimal to prefix, mentally convert to binary along the way, and as you improve, just think of the binary as the number of 1s in each octet. Figure 13-2 shows an example of such a conversion.

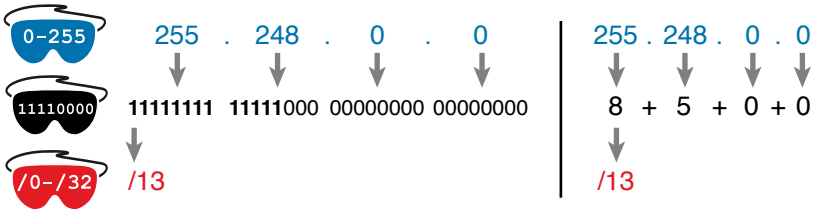


Figure 13-2 *Conversion from Decimal to Prefix: Full Binary Versus Shorthand*

Note that Appendix A has a table that lists all 33 legal subnet masks, with all three formats shown.

Practice Converting Subnet Masks

Before moving to the second half of this chapter, and thinking about what these subnet masks mean, first do some practice. Practice the processes discussed in this chapter until you get the right answer most of the time. Later, before taking the exam, practice more until you master the topics in this chapter and can move pretty fast, as outlined in the right column of Table 13-7.

Table 13-7 Keep-Reading and Take-Exam Goals for This Chapter's Topics

	Before Moving to the Next Section	Before Taking the Exam
Focus On…	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	10 seconds

Table 13-8 lists eight practice problems. The table has three columns, one for each mask format. Each row lists one mask, in one format. Your job is to find the mask's value in the other two formats for each row. Table 13-12, located in the section "Answers to Earlier Practice Problems," later in this chapter, lists the answers.

Table 13-8 Practice Problems: Find the Mask Values in the Other Two Formats

Prefix	Binary Mask	Decimal
	11111111 11111111 11000000 00000000	
		255.255.255.252
/25		
/16		
		255.0.0.0
	11111111 11111111 11111100 00000000	
		255.254.0.0
/27		

Identifying Subnet Design Choices Using Masks

Subnet masks have many purposes. In fact, if ten experienced network engineers were independently asked, "What is the purpose of a subnet mask?" the engineers would likely give a variety of true answers. The subnet mask plays several roles.

This chapter focuses on one particular use of a subnet mask: defining the prefix part of the IP addresses in a subnet. The prefix part must be the same value for all addresses in a subnet. In fact, a single subnet can be defined as all IPv4 addresses that have the same value in the prefix part of their IPv4 addresses.

While the previous paragraph might sound a bit formal, the idea is relatively basic, as shown in Figure 13-3. The figure shows a network diagram, focusing on two subnets: a subnet of all addresses that begin with 172.16.2 and another subnet made of all addresses that begin with 172.16.3. In this example, the prefix—the part that has the same value in all the addresses in the subnet—is the first three octets.

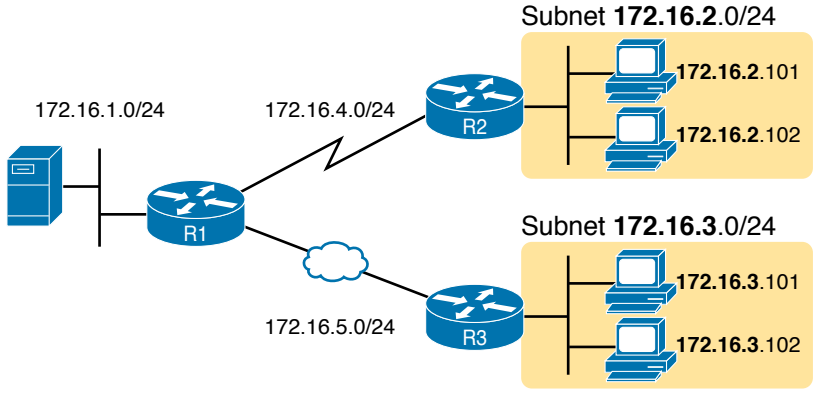


Figure 13-3 *Simple Subnet Design, with Mask /24*

While people can sit around a conference table and talk about how a prefix is three octets long, computers communicate that same concept using a subnet mask. In this case, the subnets use a subnet mask of /24, which means that the prefix part of the addresses is 24 bits (3 octets) long.

This section explains more about how to use a subnet mask to understand this concept of a prefix part of an IPv4 address, along with these other uses for a subnet mask. Note that this section discusses the first five items in the list.

Key Topic

- Defines the size of the prefix (combined network and subnet) part of the addresses in a subnet
- Defines the size of the host part of the addresses in the subnet
- Can be used to calculate the number of hosts in the subnet
- Provides a means for the network designer to communicate the design details—the number of subnet and host bits—to the devices in the network
- Under certain assumptions, can be used to calculate the number of subnets in the entire classful network
- Can be used in binary calculations of both the subnet ID and the subnet broadcast address

Masks Divide the Subnet's Addresses into Two Parts

The subnet mask subdivides the IP addresses in a subnet into two parts: the *prefix*, or *subnet part*, and the *host part*.

The prefix part identifies the addresses that reside in the same subnet because all IP addresses in the same subnet have the same value in the prefix part of their addresses. The idea is much like the postal code (ZIP codes in the United States) in mailing addresses. All mailing addresses in the same town have the same postal code. Likewise, all IP addresses in the same subnet have identical values in the prefix part of their addresses.

The host part of an address identifies the host uniquely inside the subnet. If you compare any two IP addresses in the same subnet, their host parts will differ, even though the prefix parts of their addresses have the same value. To summarize these key comparisons:

Key Topic

Prefix (subnet) part: Equal in all addresses in the same subnet.

Host part: Different in all addresses in the same subnet.

For example, imagine a subnet that, in concept, includes all addresses whose first three octets are 10.1.1. So, the following list shows several addresses in this subnet:

10.1.1.**1**

10.1.1.**2**

10.1.1.**3**

In this list, the prefix or subnet part (the first three octets of 10.1.1) are equal. The host part (the last octet [in bold]) is different. So, the prefix or subnet part of the address identifies the group, and the host part identifies the specific member of the group.

The subnet mask defines the dividing line between the prefix and the host part. To do so, the mask creates a conceptual line between the binary 1s in the binary mask and the binary 0s in the mask. In short, if a mask has P binary 1s, the prefix part is P bits long and the rest of the bits are host bits. Figure 13-4 shows the general concept.

Key Topic

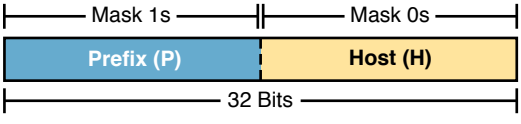


Figure 13-4 *Prefix (Subnet) and Host Parts Defined by Mask 1s and 0s*

The next figure, Figure 13-5, shows a specific example using mask 255.255.255.0. Mask 255.255.255.0 (/24) has 24 binary 1s, for a prefix length of 24 bits.

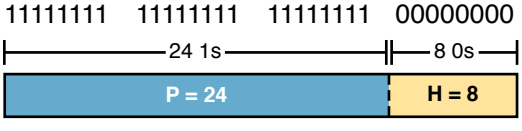


Figure 13-5 *Mask 255.255.255.0: P=24, H=8*

Masks and Class Divide Addresses into Three Parts

In addition to the two-part view of IPv4 addresses, you can also think about IPv4 addresses as having three parts. To do so, just apply Class A, B, and C rules to the address format to define the network part at the beginning of the address. This added logic divides the prefix into two parts: the *network* part and the *subnet* part. The class defines the length of the network part, with the subnet part simply being the rest of the prefix. Figure 13-6 shows the idea.

Key Topic

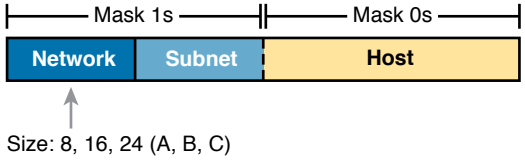


Figure 13-6 *Class Concepts Applied to Create Three Parts*

The combined network and subnet parts act like the prefix because all addresses in the same subnet must have identical values in the network and subnet parts. The size of the host part remains unchanged, whether viewing the addresses as having two parts or three parts.

To be complete, Figure 13-7 shows the same example as in the previous section, with the subnet of "all addresses that begin with 10.1.1." In that example, the subnet uses mask 255.255.255.0, and the addresses are all in Class A network 10.0.0.0. The class defines 8 network bits, and the mask defines 24 prefix bits, meaning that 24 – 8 = 16 subnet bits exist. The host part remains as 8 bits per the mask.

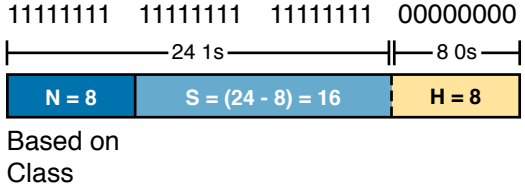


Figure 13-7 *Subnet 10.1.1.0, Mask 255.255.255.0: N=8, S=16, H=8*

Classless and Classful Addressing

The terms *classless addressing* and *classful addressing* refer to the two different ways to think about IPv4 addresses as described so far in this chapter. Classful addressing means that you think about Class A, B, and C rules, so the prefix is separated into the network and subnet parts, as shown in Figures 13-6 and 13-7. Classless addressing means that you ignore the Class A, B, and C rules and treat the prefix part as one part, as shown in Figures 13-4 and 13-5. The following more formal definitions are listed for reference and study:

Key Topic

Classless addressing: The concept that an IPv4 address has two parts—the prefix part plus the host part—as defined by the mask, with *no consideration of the class* (A, B, or C).

Classful addressing: The concept that an IPv4 address has three parts—network, subnet, and host—as defined by the mask *and Class A, B, and C rules*.

NOTE Unfortunately, the networking world uses the terms *classless* and *classful* in a couple of different ways. In addition to the classless and classful addressing described here, each routing protocol can be categorized as either a *classless routing protocol* or a *classful routing protocol*. In another use, the terms *classless routing* and *classful routing* refer to some details of how Cisco routers forward (route) packets using the default route in some cases. As a result, these terms can be easily confused and misused. So, when you see the words *classless* and *classful*, be careful to note the context: addressing, routing, or routing protocols.

Calculations Based on the IPv4 Address Format

After you know how to break an address down using both classless and classful addressing rules, you can easily calculate a couple of important facts using some basic math formulas.

First, for any subnet, after you know the number of host bits, you can calculate the number of host IP addresses in the subnet. Next, if you know the number of subnet bits (using classful addressing concepts) and you know that only one subnet mask is used throughout the network, you can also calculate the number of subnets in the network. The formulas just require that you know the powers of 2:

Hosts in the subnet: $2^H - 2$, where H is the number of host bits.

Subnets in the network: 2^S, where S is the number of subnet bits. Only use this formula if only one mask is used throughout the network.

NOTE The section "Choose the Mask" in Chapter 11, "Perspectives on IPv4 Subnetting," details many concepts related to masks, including comments about this assumption of one mask throughout a single Class A, B, or C network.

The sizes of the parts of IPv4 addresses can also be calculated. The math is basic, but the concepts are important. Keeping in mind that IPv4 addresses are 32 bits long, the two parts with classless addressing must add up to 32 (P + H = 32), and with classful addressing, the three parts must add up to 32 (N + S + H = 32). Figure 13-8 shows the relationships.

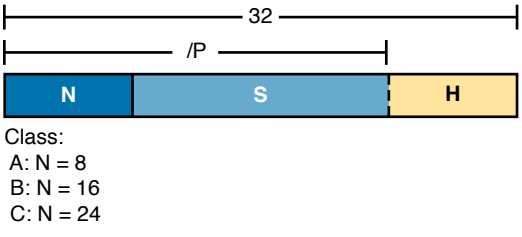


Figure 13-8 *Relationship Between /P, N, S, and H*

You often begin with an IP address and mask, both when answering questions on the CCNA exam and when examining problems that occur in real networks. Based on the information in this chapter and earlier chapters, you should be able to find all the information in Figure 13-8 and then calculate the number of hosts/subnet and the number of subnets in the network.

For reference, the following process spells out the steps:

Key Topic

Step 1. Convert the mask to prefix format (/P) as needed. (See the earlier section "Practice Converting Subnet Masks" for review.)

Step 2. Determine N based on the class. (See Chapter 12, "Analyzing Classful IPv4 Networks," for review.)

Step 3. Calculate S = P – N.

Step 4. Calculate H = 32 – P.

Step 5. Calculate hosts/subnet: $2^H - 2$.

Step 6. Calculate number of subnets: 2^S.

For example, consider the case of IP address 8.1.4.5 with mask 255.255.0.0 by following this process:

Step 1. 255.255.0.0 = /16, so P=16.

Step 2. 8.1.4.5 is in the range 1–126 in the first octet, so it is Class A; so N=8.

Step 3. S = P – N = 16 – 8 = 8.

Step 4. H = 32 – P = 32 – 16 = 16.

Step 5. $2^{16} - 2$ = 65,534 hosts/subnet.

Step 6. 2^8 = 256 subnets.

Figure 13-9 shows a visual analysis of the same problem.

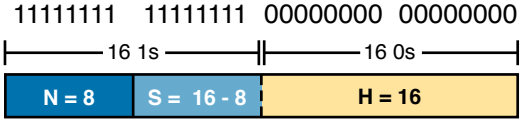


Figure 13-9 *Visual Representation of Problem: 8.1.4.5, 255.255.0.0*

For another example, consider address 200.1.1.1, mask 255.255.255.252 by following this process:

Step 1. 255.255.255.252 = /30, so P=30.

Step 2. 200.1.1.1 is in the range 192–223 in the first octet, so it is Class C; so N=24.

Step 3. S = P – N = 30 – 24 = 6.

Step 4. H = 32 – P = 32 – 30 = 2.

Step 5. $2^2 - 2$ = 2 hosts/subnet.

Step 6. 2^6 = 64 subnets.

This example uses a popular mask for serial links because serial links only require two host addresses, and the mask supports only two host addresses.

Practice Analyzing Subnet Masks

As with the other subnetting math in this book, using a two-phase approach may help. Take time now to practice until you feel as though you understand the process. Then, before the exam, make sure you master the math. Table 13-9 summarizes the key concepts and suggestions for this two-phase approach.

Table 13-9 Keep-Reading and Take-Exam Goals for This Chapter's Topics

	Before Moving to the Next Chapter	Before Taking the Exam
Focus On…	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	15 seconds

On a piece of scratch paper, answer the following questions. In each case:

- Determine the structure of the addresses in each subnet based on the class and mask, using classful IP addressing concepts. In other words, find the size of the network, subnet, and host parts of the addresses.
- Calculate the number of hosts in the subnet.
- Calculate the number of subnets in the network, assuming that the same mask is used throughout.

1. 8.1.4.5, 255.255.254.0
2. 130.4.102.1, 255.255.255.0
3. 199.1.1.100, 255.255.255.0
4. 130.4.102.1, 255.255.252.0
5. 199.1.1.100, 255.255.255.224

The answers are listed in the section "Answers to Earlier Practice Problems," later in this chapter.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 13-10 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 13-10 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website
Practice analyzing subnet masks		Website, Appendix E

Review All the Key Topics

Key Topic

Table 13-11 Key Topics for Chapter 13

Key Topic Element	Description	Page Number
List	Rules for binary subnet mask values	304
List	Rules to convert between binary and prefix masks	305
Table 13-4	Nine possible values in a decimal subnet mask	307
List	Rules to convert between binary and DDN masks	307
List	Some functions of a subnet mask	310
List	Comparisons of IP addresses in the same subnet	311
Figure 13-4	Two-part classless view of an IP address	311
Figure 13-6	Three-part classful view of an IP address	312
List	Definitions of classful addressing and classless addressing	312
List	Formal steps to analyze masks and calculate values	314

Key Terms You Should Know

binary mask, dotted-decimal notation (DDN), decimal mask, prefix mask, CIDR mask, classful addressing, classless addressing

Additional Practice for This Chapter's Processes

You can do more practice with the processes in this chapter with a pair of practice sets. One focuses on interpreting existing masks, while the other gives you practice with converting between mask formats. You may do each practice set using the following tools:

Application: Use the "Analyzing Subnet Masks" and "Converting Masks" applications on the companion website, listed under the Chapter Review for this chapter.

PDF: Alternatively, practice the same problems found in both these apps using companion website Appendix E, "Practice for Chapter 13: Analyzing Subnet Masks."

Answers to Earlier Practice Problems

Table 13-8, shown earlier, listed several practice problems for converting subnet masks; Table 13-12 lists the answers.

Table 13-12 Answers to Problems in Table 13-8

Prefix	Binary Mask	Decimal
/18	11111111 11111111 11000000 00000000	255.255.192.0
/30	11111111 11111111 11111111 11111100	255.255.255.252
/25	11111111 11111111 11111111 10000000	255.255.255.128
/16	11111111 11111111 00000000 00000000	255.255.0.0
/8	11111111 00000000 00000000 00000000	255.0.0.0
/22	11111111 11111111 11111100 00000000	255.255.252.0
/15	11111111 11111110 00000000 00000000	255.254.0.0
/27	11111111 11111111 11111111 11100000	255.255.255.224

Table 13-13 lists the answers to the practice problems from the earlier section "Practice Analyzing Subnet Masks."

Table 13-13 Answers to Problems from Earlier in the Chapter

	Problem	/P	Class	N	S	H	2^S	$2^H - 2$
1	8.1.4.5 255.255.254.0	23	A	8	15	9	32,768	510
2	130.4.102.1 255.255.255.0	24	B	16	8	8	256	254
3	199.1.1.100 255.255.255.0	24	C	24	0	8	N/A	254
4	130.4.102.1 255.255.252.0	22	B	16	6	10	64	1022
5	199.1.1.100 255.255.255.224	27	C	24	3	5	8	30

The following list reviews the problems:

1. For 8.1.4.5, the first octet (8) is in the 1–126 range, so it is a Class A address, with 8 network bits. Mask 255.255.254.0 converts to /23, so P – N = 15, for 15 subnet bits. H can be found by subtracting /P (23) from 32, for 9 host bits.
2. 130.4.102.1 is in the 128–191 range in the first octet, making it a Class B address, with N = 16 bits. 255.255.255.0 converts to /24, so the number of subnet bits is 24 – 16 = 8. With 24 prefix bits, the number of host bits is 32 – 24 = 8.
3. The third problem purposely shows a case where the mask does not create a subnet part of the address. The address, 199.1.1.100, has a first octet between 192 and 223, making it a Class C address with 24 network bits. The prefix version of the mask is /24, so the number of subnet bits is 24 – 24 = 0. The number of host bits is 32 minus

the prefix length (24), for a total of 8 host bits. So in this case, the mask shows that the network engineer is using the default mask, which creates no subnet bits and no subnets.

4. With the same address as the second problem, 130.4.102.1 is a Class B address with N = 16 bits. This problem uses a different mask, 255.255.252.0, which converts to /22. This makes the number of subnet bits 22 – 16 = 6. With 22 prefix bits, the number of host bits is 32 – 22 = 10.

5. With the same address as the third problem, 199.1.1.100 is a Class C address with N = 24 bits. This problem uses a different mask, 255.255.255.224, which converts to /27. This makes the number of subnet bits 27 – 24 = 3. With 27 prefix bits, the number of host bits is 32 – 27 = 5.

This page intentionally left blank

CHAPTER 14

Analyzing Existing Subnets

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

Often, a networking task begins with the discovery of the IP address and mask used by some host. Then, to understand how the internetwork routes packets to that host, you must find key pieces of information about the subnet, specifically the following:

- Subnet ID
- Subnet broadcast address
- Subnet's range of usable unicast IP addresses

This chapter discusses the concepts and math to take a known IP address and mask, and then fully describe a subnet by finding the values in this list. These specific tasks might well be the most important IP skills in the entire IP addressing and subnetting topics in this book because these tasks might be the most commonly used tasks when operating and troubleshooting real networks.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 14-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Defining a Subnet	1
Analyzing Existing Subnets: Binary	2
Analyzing Existing Subnets: Decimal	3–6

1. When you think about an IP address using classful addressing rules, an address can have three parts: network, subnet, and host. If you examined all the addresses in one subnet, in binary, which of the following answers correctly states which of the three parts of the addresses will be equal among all addresses? (Choose the best answer.)
   - **a.** Network part only
   - **b.** Subnet part only
   - **c.** Host part only
   - **d.** Network and subnet parts
   - **e.** Subnet and host parts

2. Which of the following statements are true regarding the binary subnet ID, subnet broadcast address, and host IP address values in any single subnet? (Choose two answers.)
   a. The host part of the broadcast address is all binary 0s.
   b. The host part of the subnet ID is all binary 0s.
   c. The host part of a usable IP address can have all binary 1s.
   d. The host part of any usable IP address must not be all binary 0s.
3. Which of the following is the resident subnet ID for IP address 10.7.99.133/24?
   a. 10.0.0.0
   b. 10.7.0.0
   c. 10.7.99.0
   d. 10.7.99.128
4. Which of the following is the resident subnet for IP address 192.168.44.97/30?
   a. 192.168.44.0
   b. 192.168.44.64
   c. 192.168.44.96
   d. 192.168.44.128
5. Which of the following is the subnet broadcast address for the subnet in which IP address 172.31.77.201/27 resides?
   a. 172.31.201.255
   b. 172.31.255.255
   c. 172.31.77.223
   d. 172.31.77.207
6. A fellow engineer tells you to configure the DHCP server to lease the last 100 usable IP addresses in subnet 10.1.4.0/23. Which of the following IP addresses could be leased as a result of your new configuration?
   a. 10.1.4.156
   b. 10.1.4.254
   c. 10.1.5.200
   d. 10.1.7.200
   e. 10.1.255.200

Foundation Topics

Defining a Subnet

An IP subnet is a subset of a classful network, created by choice of some network engineer. However, that engineer cannot pick just any arbitrary subset of addresses; instead, the engineer must follow certain rules, such as the following:

Key Topic

- The subnet contains a set of consecutive numbers.
- The subnet holds 2^H numbers, where H is the number of host bits defined by the subnet mask.
- Two special numbers in the range cannot be used as IP addresses:
  - The first (lowest) number acts as an identifier for the subnet (*subnet ID*).
  - The last (highest) number acts as a *subnet broadcast address*.
- The remaining addresses, whose values sit between the subnet ID and subnet broadcast address, are used as *unicast IP addresses*.

This section reviews and expands the basic concepts of the subnet ID, subnet broadcast address, and range of addresses in a subnet.

An Example with Network 172.16.0.0 and Four Subnets

Imagine that you work at the customer support center, where you receive all initial calls from users who have problems with their computer. You coach the user through finding her IP address and mask: 172.16.150.41, mask 255.255.192.0. One of the first and most common tasks you will do based on that information is to find the subnet ID of the subnet in which that address resides. (In fact, this subnet ID is sometimes called the *resident subnet* because the IP address exists in or resides in that subnet.)

Before getting into the math, examine the mask (255.255.192.0) and classful network (172.16.0.0) for a moment. From the mask, based on what you learned in Chapter 13, "Analyzing Subnet Masks," you can find the structure of the addresses in the subnet, including the number of host and subnet bits. That analysis tells you that two subnet bits exist, meaning that there should be four (2^2) subnets. Figure 14-1 shows the idea.

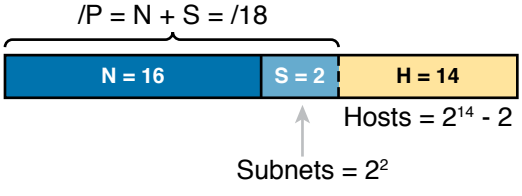


Figure 14-1 *Address Structure: Class B Network, /18 Mask*

Answers to the "Do I Know This Already?" quiz:

1 D **2** B, D **3** C **4** C **5** C **6** C

NOTE This chapter, like the others in this part of the book, assumes that one mask is used throughout an entire classful network.

Because each subnet uses a single mask, all subnets of this single IP network must be the same size, because all subnets have the same structure. In this example, all four subnets will have the structure shown in the figure, so all four subnets will have $2^{14} - 2$ host addresses.

Next, consider the big picture of what happens with this example subnet design: the one Class B network now has four subnets of equal size. Conceptually, if you represent the entire Class B network as a number line, each subnet consumes one-fourth of the number line, as shown in Figure 14-2. Each subnet has a subnet ID—the numerically lowest number in the subnet—so it sits on the left of the subnet. And each subnet has a subnet broadcast address—the numerically highest number in the subnet—so it sits on the right side of the subnet.

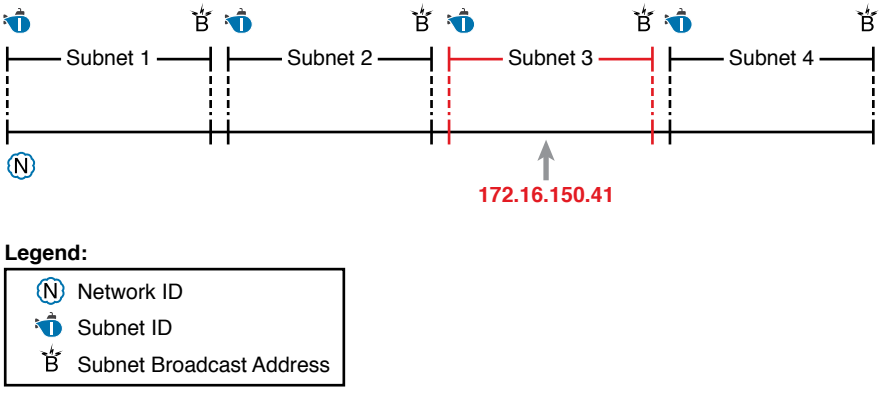


Figure 14-2 *Network 172.16.0.0, Divided into Four Equal Subnets*

The rest of this chapter focuses on how to take one IP address and mask and discover the details about that one subnet in which the address resides. In other words, you see how to find the resident subnet of an IP address. Again, using IP address 172.16.150.41 and mask 255.255.192.0 as an example, Figure 14-3 shows the resident subnet, along with the subnet ID and subnet broadcast address that bracket the subnet.

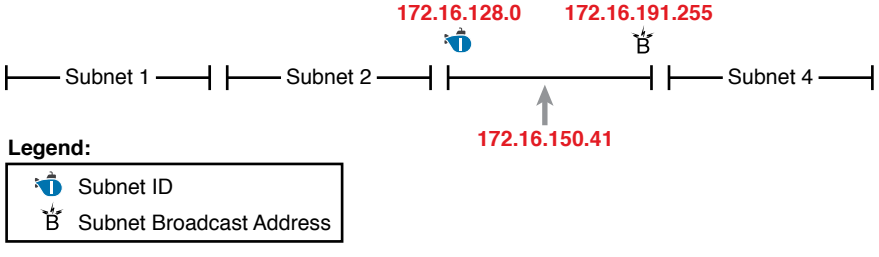


Figure 14-3 *Resident Subnet for 172.16.150.41, 255.255.192.0*

Subnet ID Concepts

A subnet ID is simply a number used to succinctly represent a subnet. When listed along with its matching subnet mask, the subnet ID identifies the subnet and can be used to derive the subnet broadcast address and range of addresses in the subnet. Rather than having to write down all these details about a subnet, you simply need to write down the subnet ID and mask, and you have enough information to fully describe the subnet.

The subnet ID appears in many places, but it is seen most often in IP routing tables. For example, when an engineer configures a router with its IP address and mask, the router calculates the subnet ID and puts a route into its routing table for that subnet. The router typically then advertises the subnet ID/mask combination to neighboring routers with some IP routing protocol. Eventually, all the routers in an enterprise learn about the subnet—again using the subnet ID and subnet mask combination—and display it in their routing tables. (You can display the contents of a router's IP routing table using the **show ip route** command.)

Unfortunately, the terminology related to subnets can sometimes cause problems. First, the terms *subnet ID*, *subnet number*, and *subnet address* are synonyms. In addition, people sometimes simply say *subnet* when referring to both the idea of a subnet and the number that is used as the subnet ID. When talking about routing, people sometimes use the term *prefix* instead of *subnet*. The term *prefix* refers to the same idea as *subnet*; it just uses terminology from the classless addressing way to describe IP addresses, as discussed in Chapter 13's section "Classless and Classful Addressing."

The biggest terminology confusion arises between the terms *network* and *subnet*. In the real world, people often use these terms synonymously, and that is perfectly reasonable in some cases. In other cases, the specific meaning of these terms, and their differences, matter to what is being discussed.

For example, people often might say, "What is the network ID?" when they really want to know the subnet ID. In another case, they might want to know the Class A, B, or C network ID. So, when one engineer asks something like, "What's the net ID for 172.16.150.41 slash 18?" use the context to figure out whether he wants the literal classful network ID (172.16.0.0, in this case) or the literal subnet ID (172.16.128.0, in this case).

For the exams, be ready to notice when the terms *subnet* and *network* are used, and then use the context to figure out the specific meaning of the term in that case.

Table 14-2 summarizes the key facts about the subnet ID, along with the possible synonyms, for easier review and study.

Key Topic

Table 14-2 Summary of Subnet ID Key Facts

Definition	Number that represents the subnet
Numeric Value	First (smallest) number in the subnet
Literal Synonyms	Subnet number, subnet address, prefix, resident subnet
Common-Use Synonyms	Network, network ID, network number, network address
Typically Seen In…	Routing tables, documentation

Subnet Broadcast Address

The subnet broadcast address has two main roles: to be used as a destination IP address for the purpose of sending packets to all hosts in the subnet, and as a means to find the high end of the range of addresses in a subnet.

The original purpose for the subnet broadcast address was to give hosts a way to send one packet to all hosts in a subnet and to do so efficiently. For example, a host in subnet A could send a packet with a destination address of subnet B's subnet broadcast address. The routers would forward this one packet just like a packet sent to a host in subnet B. After the packet arrives at the router connected to subnet B, that last router would then forward the packet to all hosts in subnet B, typically by encapsulating the packet in a data-link layer broadcast frame. As a result, all hosts in host B's subnet would receive a copy of the packet.

The subnet broadcast address also helps you find the range of addresses in a subnet because the broadcast address is the last (highest) number in a subnet's range of addresses. To find the low end of the range, calculate the subnet ID; to find the high end of the range, calculate the subnet broadcast address.

Table 14-3 summarizes the key facts about the subnet broadcast address, along with the possible synonyms, for easier review and study.

Key Topic

Table 14-3 Summary of Subnet Broadcast Address Key Facts

Definition	A reserved number in each subnet that, when used as the destination address of a packet, causes the device to forward the packet to all hosts in that subnet
Numeric Value	Last (highest) number in the subnet
Literal Synonyms	Directed broadcast address
Broader-Use Synonyms	Network broadcast
Typically Seen In…	In calculations of the range of addresses in a subnet

Range of Usable Addresses

The engineers implementing an IP internetwork need to know the range of unicast IP addresses in each subnet. Before you can plan which addresses to use as statically assigned IP addresses, which to configure to be leased by the DHCP server, and which to reserve for later use, you need to know the range of usable addresses.

To find the range of usable IP addresses in a subnet, first find the subnet ID and the subnet broadcast address. Then, just add 1 to the fourth octet of the subnet ID to get the first (lowest) usable address, and subtract 1 from the fourth octet of the subnet broadcast address to get the last (highest) usable address in the subnet.

For example, Figure 14-3 showed subnet ID 172.16.128.0, mask /18. The first usable address is simply one more than the subnet ID (in this case, 172.16.128.1). That same figure showed a subnet broadcast address of 172.16.191.255, so the last usable address is one less, or 172.16.191.254.

Now that this section has described the concepts behind the numbers that collectively define a subnet, the rest of this chapter focuses on the math used to find these values.

Analyzing Existing Subnets: Binary

What does it mean to "analyze a subnet"? For this book, it means that you should be able to start with an IP address and mask and then define key facts about the subnet in which that address resides. Specifically, that means discovering the subnet ID, subnet broadcast address, and range of addresses. The analysis can also include the calculation of the number of addresses in the subnet as discussed in Chapter 13, but this chapter does not review those concepts.

Many methods exist to calculate the details about a subnet based on the address/mask. This section begins by discussing some calculations that use binary math, with the next section showing alternatives that use only decimal math. Although many people prefer the decimal method for going fast on the exams, the binary calculations ultimately give you a better understanding of IPv4 addressing. In particular, if you plan to move on to attain Cisco certifications beyond CCNA, you should take the time to understand the binary methods discussed in this section, even if you use the decimal methods for the exams.

Finding the Subnet ID: Binary

The two following statements summarize the logic behind the binary value of any subnet ID:

All numbers in the subnet (subnet ID, subnet broadcast address, and all usable IP addresses) have the same value in the prefix part of the numbers.

The subnet ID is the lowest numeric value in the subnet, so its host part, in binary, is all 0s.

To find the subnet ID in binary, you take the IP address in binary and change all host bits to binary 0. To do so, you need to convert the IP address to binary. You also need to identify the prefix and host bits, which can be easily done by converting the mask (as needed) to prefix format. (Note that Appendix A, "Numeric Reference Tables," includes a decimal-binary conversion table.) Figure 14-4 shows the idea, using the same address/mask as in the earlier examples in this chapter: 172.16.150.41, mask /18.

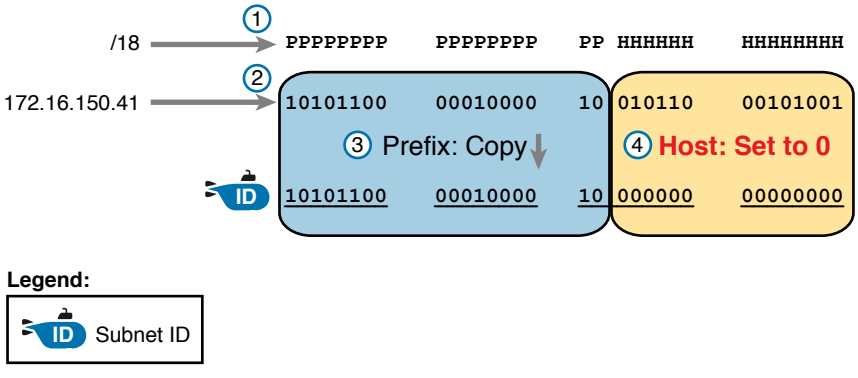


Figure 14-4 *Binary Concept: Convert the IP Address to the Subnet ID*

Starting at the top of Figure 14-4, the format of the IP address is represented with 18 prefix (P) and 14 host (H) bits in the mask (Step 1). The second row (Step 2) shows the binary version of the IP address, converted from the dotted-decimal notation (DDN) value 172.16.150.41. (If you have not yet used the conversion table in Appendix A, it might be useful to double-check the conversion of all four octets based on the table.)

The next two steps show the action to copy the IP address's prefix bits (Step 3) and give the host bits a value of binary 0 (Step 4). This resulting number is the subnet ID (in binary).

The last step, not shown in Figure 14-4, is to convert the subnet ID from binary to decimal. This book shows that conversion as a separate step, in Figure 14-5, mainly because many people make a mistake at this step in the process. When converting a 32-bit number (like an IP address or IP subnet ID) back to an IPv4 DDN, you must follow this rule:

> Convert 8 bits at a time from binary to decimal, regardless of the line between the prefix and host parts of the number.

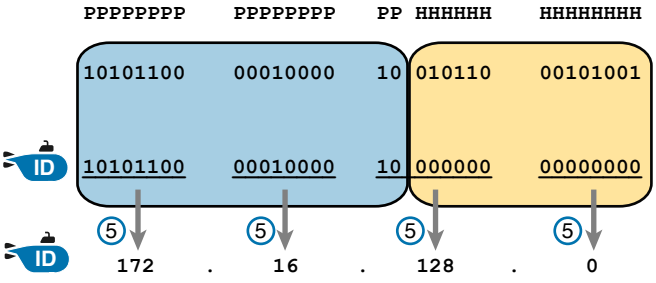


Figure 14-5 *Converting the Subnet ID from Binary to DDN*

Figure 14-5 shows this final step. Note that the third octet (the third set of 8 bits) has 2 bits in the prefix and 6 bits in the host part of the number, but the conversion occurs for all 8 bits.

NOTE You can do the numeric conversions in Figures 14-4 and 14-5 by relying on the conversion table in Appendix A. To convert from DDN to binary, for each octet, find the decimal value in the table and then write down the 8-bit binary equivalent. To convert from binary back to DDN, for each octet of 8 bits, find the matching binary entry in the table and write down the corresponding decimal value. For example, 172 converts to binary 10101100, and 00010000 converts to decimal 16.

Finding the Subnet Broadcast Address: Binary

Finding the subnet broadcast address uses a similar process. To find the subnet broadcast address, use the same binary process used to find the subnet ID, but instead of setting all the host bits to the lowest value (all binary 0s), set the host part to the highest value (all binary 1s). Figure 14-6 shows the concept.

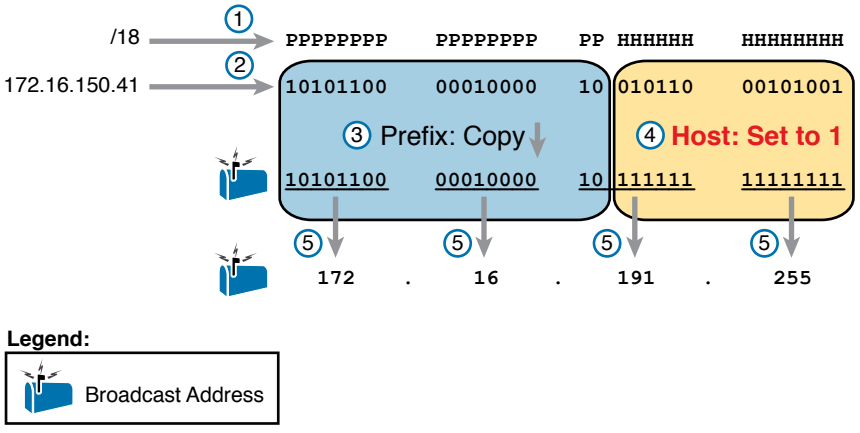


Figure 14-6 *Finding a Subnet Broadcast Address: Binary*

The process in Figure 14-6 demonstrates the same first three steps shown in Figure 14-4. Specifically, it shows the identification of the prefix and host bits (Step 1), the results of converting the IP address 172.16.150.41 to binary (Step 2), and the copying of the prefix bits (first 18 bits, in this case). The difference occurs in the host bits on the right, changing all host bits (the last 14, in this case) to the largest possible value (all binary 1s). The final step converts the 32-bit subnet broadcast address to DDN format. Also, remember that with any conversion from DDN to binary or vice versa, the process always converts using 8 bits at a time. In particular, in this case, the entire third octet of binary 10111111 is converted back to decimal 191.

Binary Practice Problems

Figures 14-4 and 14-5 demonstrate a process to find the subnet ID using binary math. The following process summarizes those steps in written form for easier reference and practice:

Key Topic

Step 1. Convert the mask to prefix format to find the length of the prefix (/P) and the length of the host part (32 – P).

Step 2. Convert the IP address to its 32-bit binary equivalent.

Step 3. Copy the prefix bits of the IP address.

Step 4. Write down 0s for the host bits.

Step 5. Convert the resulting 32-bit number, 8 bits at a time, back to decimal.

The process to find the subnet broadcast address is exactly the same, except in Step 4, you set the bits to 1s, as shown in Figure 14-6.

Take a few moments and run through the following five practice problems on scratch paper. In each case, find both the subnet ID and subnet broadcast address. Also, record the prefix style mask:

1. 8.1.4.5, 255.255.0.0
2. 130.4.102.1, 255.255.255.0
3. 199.1.1.100, 255.255.255.0

4. 130.4.102.1, 255.255.252.0
5. 199.1.1.100, 255.255.255.224

Tables 14-4 through 14-8 show the results for the five different examples. The tables show the host bits in bold, and they include the binary version of the address and mask and the binary version of the subnet ID and subnet broadcast address.

Table 14-4 Subnet Analysis for Subnet with Address 8.1.4.5, Mask 255.255.0.0

Prefix Length	/16	11111111 11111111 **00000000 00000000**
Address	8.1.4.5	00001000 00000001 **00000100 00000101**
Subnet ID	8.1.0.0	00001000 00000001 **00000000 00000000**
Broadcast Address	8.1.255.255	00001000 00000001 **11111111 11111111**

Table 14-5 Subnet Analysis for Subnet with Address 130.4.102.1, Mask 255.255.255.0

Prefix Length	/24	11111111 11111111 11111111 **00000000**
Address	130.4.102.1	10000010 00000100 01100110 **00000001**
Subnet ID	130.4.102.0	10000010 00000100 01100110 **00000000**
Broadcast Address	130.4.102.255	10000010 00000100 01100110 **11111111**

Table 14-6 Subnet Analysis for Subnet with Address 199.1.1.100, Mask 255.255.255.0

Prefix Length	/24	11111111 11111111 11111111 **00000000**
Address	199.1.1.100	11000111 00000001 00000001 **01100100**
Subnet ID	199.1.1.0	11000111 00000001 00000001 **00000000**
Broadcast Address	199.1.1.255	11000111 00000001 00000001 **11111111**

Table 14-7 Subnet Analysis for Subnet with Address 130.4.102.1, Mask 255.255.252.0

Prefix Length	/22	11111111 11111111 111111**00** **00000000**
Address	130.4.102.1	10000010 00000100 011001**10** **00000001**
Subnet ID	130.4.100.0	10000010 00000100 011001**00** **00000000**
Broadcast Address	130.4.103.255	10000010 00000100 011001**11** **11111111**

Table 14-8 Subnet Analysis for Subnet with Address 199.1.1.100, Mask 255.255.255.224

Prefix Length	/27	11111111 11111111 11111111 **111**00000
Address	199.1.1.100	11000111 00000001 00000001 **011**00100
Subnet ID	199.1.1.96	11000111 00000001 00000001 **011**00000
Broadcast Address	199.1.1.127	11000111 00000001 00000001 **011**11111

Shortcut for the Binary Process

The binary process described in this section so far requires that all four octets be converted to binary and then back to decimal. However, you can easily predict the results in at least three of the four octets, based on the DDN mask. You can then avoid the binary math in all but one octet and reduce the number of binary conversions you need to do.

First, consider an octet, and that octet only, whose DDN mask value is 255. The mask value of 255 converts to binary 11111111, which means that all 8 bits are prefix bits. Thinking through the steps in the process, at Step 2, you convert the address to some number. At Step 3, you copy the number. At Step 4, you convert the same 8-bit number back to decimal. All you did in those three steps, in this one octet, is convert from decimal to binary and convert the same number back to the same decimal value!

In short, the subnet ID (and subnet broadcast address) are equal to the IP address in octets for which the mask is 255.

For example, the resident subnet ID for 172.16.150.41, mask 255.255.192.0 is 172.16.128.0. The first two mask octets are 255. Rather than think about the binary math, you could just start by copying the address's value in those two octets: 172.16.

Another shortcut exists for octets whose DDN mask value is decimal 0, or binary 00000000. With a decimal mask value of 0, the math always results in a decimal 0 for the subnet ID, no matter the beginning value in the IP address. Specifically, just look at Steps 4 and 5 in this case: At Step 4, you would write down 8 binary 0s, and at Step 5, you would convert 00000000 back to decimal 0.

The following revised process steps take these two shortcuts into account. However, when the mask is neither 0 nor 255, the process requires the same conversions. At most, you have to do only one octet of the conversions. To find the subnet ID, apply the logic in these steps for each of the four octets:

Step 1. If the mask = 255, copy the decimal IP address for that octet.

Step 2. If the mask = 0, write down a decimal 0 for that octet.

Step 3. If the mask is neither 0 nor 255 in this octet, use the same binary logic as shown in the section "Finding the Subnet ID: Binary," earlier in this chapter.

Figure 14-7 shows an example of this process, again using 172.16.150.41, 255.255.192.0.

To find the subnet broadcast address, you can use a decimal shortcut similar to the one used to find the subnet ID: for DDN mask octets equal to decimal 0, set the decimal subnet broadcast address value to 255 instead of 0, as noted in the following list:

Step 1. If the mask = 255, copy the decimal IP address for that octet.

Step 2. If the mask = 0, write down a decimal 255 for that octet.

Step 3. If the mask is neither 0 nor 255 in this octet, use the same binary logic as shown in the section "Finding the Subnet Broadcast Address: Binary," earlier in this chapter.

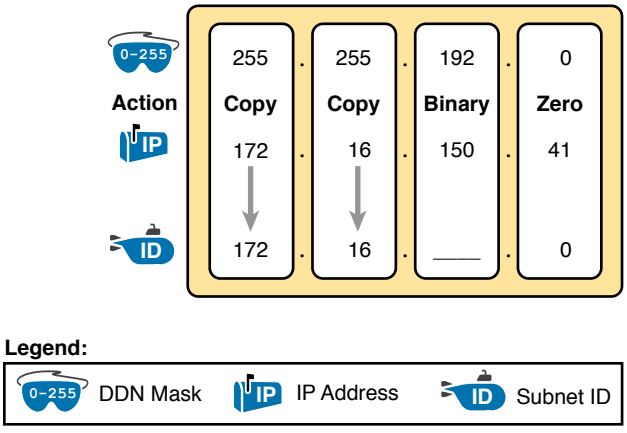


Figure 14-7 *Binary Shortcut Example*

Brief Note About Boolean Math

So far, this chapter has described how humans can use binary math to find the subnet ID and subnet broadcast address. However, computers typically use an entirely different binary process to find the same values, using a branch of mathematics called *Boolean algebra*. Computers already store the IP address and mask in binary form, so they do not have to do any conversions to and from decimal. Then, certain Boolean operations allow the computers to calculate the subnet ID and subnet broadcast address with just a few CPU instructions.

You do not need to know Boolean math to have a good understanding of IP subnetting. However, in case you are interested, computers use the following Boolean logic to find the subnet ID and subnet broadcast address, respectively:

Perform a *Boolean AND* of the IP address and mask. This process converts all host bits to binary 0.

Invert the mask, and then perform a *Boolean OR* of the IP address and inverted subnet mask. This process converts all host bits to binary 1s.

Finding the Range of Addresses

Finding the range of usable addresses in a subnet, after you know the subnet ID and subnet broadcast address, requires only simple addition and subtraction. To find the first (lowest) usable IP address in the subnet, simply add 1 to the fourth octet of the subnet ID. To find the last (highest) usable IP address, simply subtract 1 from the fourth octet of the subnet broadcast address.

Analyzing Existing Subnets: Decimal

Analyzing existing subnets using the binary process works well. However, some of the math takes time for most people, particularly the decimal-binary conversions. And you need to do the math quickly for the Cisco CCNA exam. For the exam, you really should be able to take an IP address and mask, and calculate the subnet ID and range of usable addresses within about 15 seconds. When using binary methods, most people require a lot of practice to be able to find these answers, even when using the abbreviated binary process.

This section discusses how to find the subnet ID and subnet broadcast address using only decimal math. Most people can find the answers more quickly using this process, at least after a little practice, as compared with the binary process. However, the decimal process does not tell you anything about the meaning behind the math. So, if you have not read the earlier section "Analyzing Existing Subnets: Binary," it is worthwhile to read it for the sake of understanding subnetting. This section focuses on getting the right answer using a method that, after you have practiced, should be faster.

Analysis with Easy Masks

With three easy subnet masks in particular, finding the subnet ID and subnet broadcast address requires only easy logic and literally no math. Three easy masks exist:

255.0.0.0

255.255.0.0

255.255.255.0

These easy masks have only 255 and 0 in decimal. In comparison, difficult masks have one octet that has neither a 255 nor a 0 in the mask, which makes the logic more challenging.

NOTE The terms *easy mask* and *difficult mask* are terms created for use in this book to describe the masks and the level of difficulty when working with each.

When the problem uses an easy mask, you can quickly find the subnet ID based on the IP address and mask in DDN format. Just use the following process for each of the four octets to find the subnet ID:

Step 1. If the mask octet = 255, copy the decimal IP address.

Step 2. If the mask octet = 0, write a decimal 0.

A similar simple process exists to find the subnet broadcast address, as follows:

Step 1. If the mask octet = 255, copy the decimal IP address.

Step 2. If the mask octet = 0, write a decimal 255.

Before moving to the next section, take some time to fill in the blanks in Table 14-9. Check your answers against Table 14-15 in the section "Answers to Earlier Practice Problems," later in this chapter. Complete the table by listing the subnet ID and subnet broadcast address.

Table 14-9 Practice Problems: Find Subnet ID and Broadcast Address, Easy Masks

	IP Address	Mask	Subnet ID	Broadcast Address
1	10.77.55.3	255.255.255.0		
2	172.30.99.4	255.255.255.0		
3	192.168.6.54	255.255.255.0		
4	10.77.3.14	255.255.0.0		
5	172.22.55.77	255.255.0.0		
6	1.99.53.76	255.0.0.0		

Predictability in the Interesting Octet

Although three masks are easier to work with (255.0.0.0, 255.255.0.0, and 255.255.255.0), the rest make the decimal math a little more difficult, so we call these masks difficult masks. With difficult masks, one octet is neither a 0 nor a 255. The math in the other three octets is easy and boring, so this book calls the one octet with the more difficult math the *interesting octet*.

If you take some time to think about different problems and focus on the interesting octet, you will begin to see a pattern. This section takes you through that examination so that you can learn how to predict the pattern, in decimal, and find the subnet ID.

First, the subnet ID value has a predictable decimal value because of the assumption that a single subnet mask is used for all subnets of a single classful network. The chapters in this part of the book assume that, for a given classful network, the design engineer chooses to use a single subnet mask for all subnets. (See the section "One Size Subnet Fits All—Or Not" in Chapter 11, "Perspectives on IPv4 Subnetting," for more details.)

To see that predictability, consider some planning information written down by a network engineer, as shown in Figure 14-8. The figure shows four different masks the engineer is considering using in an IPv4 network, along with Class B network 172.16.0.0. The figure shows the third-octet values for the subnet IDs that would be created when using mask 255.255.128.0, 255.255.192.0, 255.255.224.0, and 255.255.240.0, from top to bottom in the figure.

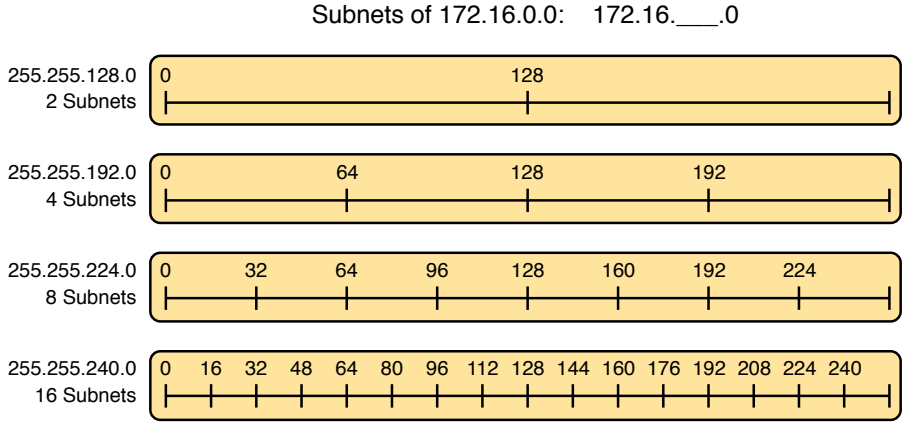


Figure 14-8 *Numeric Patterns in the Interesting Octet*

First, to explain the figure further, look at the top row of the figure. If the engineer uses 255.255.128.0 as the mask, the mask creates two subnets, with subnet IDs 172.16.0.0 and 172.16.128.0. If the engineer uses mask 255.255.192.0, the mask creates four subnets, with subnet IDs 172.16.0.0, 172.16.64.0, 172.16.128.0, and 172.16.192.0.

If you take the time to look at the figure, the patterns become obvious. In this case:

Mask: 255.255.128.0 Pattern: Multiples of 128

Mask: 255.255.192.0 Pattern: Multiples of 64

Mask: 255.255.224.0 Pattern: Multiples of 32

Mask: 255.255.240.0 Pattern: Multiples of 16

To find the subnet ID, you just need a way to figure out what the pattern is. If you start with an IP address and mask, just find the subnet ID closest to the IP address, without going over, as discussed in the next section.

Finding the Subnet ID: Difficult Masks

The following written process lists all the steps to find the subnet ID, using only decimal math. This process adds to the earlier process used with easy masks. For each octet:

Step 1. If the mask octet = 255, copy the decimal IP address.

Step 2. If the mask octet = 0, write a decimal 0.

Step 3. If the mask is neither, refer to this octet as the *interesting octet*:

A. Calculate the *magic number* as 256 – mask.

B. Set the subnet ID's value to the multiple of the magic number that is closest to the IP address without going over.

The process uses two new terms created for this book: *magic number* and *interesting octet*. The term *interesting octet* refers to the octet identified at Step 3 in the process; in other words, it is the octet with the mask that is neither 255 nor 0. Step 3A then uses the term *magic number*, which is derived from the DDN mask. Conceptually, the magic number is the number you add to one subnet ID to get the next subnet ID in order, as shown in Figure 14-8. Numerically, it can be found by subtracting the DDN mask's value, in the interesting octet, from 256, as mentioned in Step 3A.

The best way to learn this process is to see it happen. In fact, if you can, stop reading now, use the companion website for this book, and watch the videos about finding the subnet ID with a difficult mask. These videos demonstrate this process. You can also use the examples on the next few pages that show the process being used on paper. Then follow the practice opportunities outlined in the section "Practice Analyzing Existing Subnets," later in this chapter.

Resident Subnet Example 1

For example, consider the requirement to find the resident subnet for IP address 130.4.102.1, mask 255.255.240.0. The process does not require you to think about prefix bits versus host bits, convert the mask, think about the mask in binary, or convert the IP address to and from

binary. Instead, for each of the four octets, choose an action based on the value in the mask. Figure 14-9 shows the results; the circled numbers in the figure refer to the step numbers in the written process to find the subnet ID, as listed in the previous few pages.

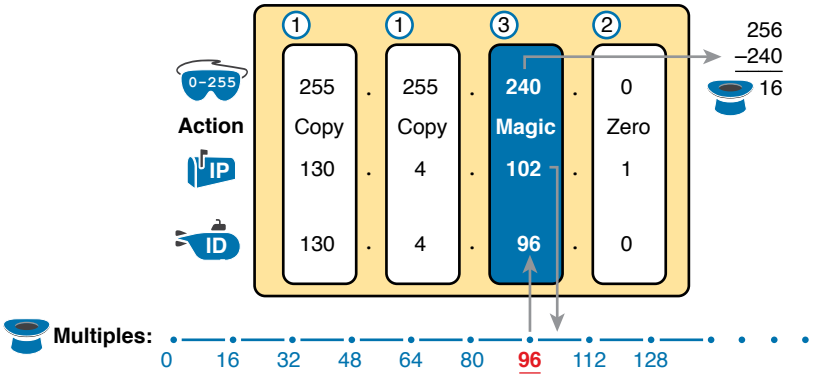


Figure 14-9 *Find the Subnet ID: 130.4.102.1, 255.255.240.0*

First, examine the three uninteresting octets (1, 2, and 4, in this example). The process keys on the mask, and the first two octets have a mask value of 255, so simply copy the IP address to the place where you intend to write down the subnet ID. The fourth octet has a mask value of 0, so write down a 0 for the fourth octet of the subnet ID.

The most challenging logic occurs in the interesting octet, which is the third octet in this example, because of the mask value 240 in that octet. For this octet, Step 3A asks you to calculate the magic number as 256 – mask. That means you take the mask's value in the interesting octet (240, in this case) and subtract it from 256: 256 – 240 = 16. The subnet ID's value in this octet must be a multiple of decimal 16, in this case.

Step 3B then asks you to find the multiples of the magic number (16, in this case) and choose the one closest to the IP address without going over. Specifically, that means that you should mentally calculate the multiples of the magic number, starting at 0. (Do not forget to start at 0!) Count, starting at 0: 0, 16, 32, 48, 64, 80, 96, 112, and so on. Then, find the multiple closest to the IP address value in this octet (102, in this case), without going over 102. So, as shown in Figure 14-9, you make the third octet's value 96 to complete the subnet ID of 130.4.96.0.

Resident Subnet Example 2

Consider another example: 192.168.5.77, mask 255.255.255.224. Figure 14-10 shows the results.

The three uninteresting octets (1, 2, and 3, in this case) require only a little thought. For each octet, each with a mask value of 255, just copy the IP address.

For the interesting octet, at Step 3A, the magic number is 256 – 224 = 32. The multiples of the magic number are 0, 32, 64, 96, and so on. Because the IP address value in the fourth octet is 77, in this case, the multiple must be the number closest to 77 without going over; therefore, the subnet ID ends with 64, for a value of 192.168.5.64.

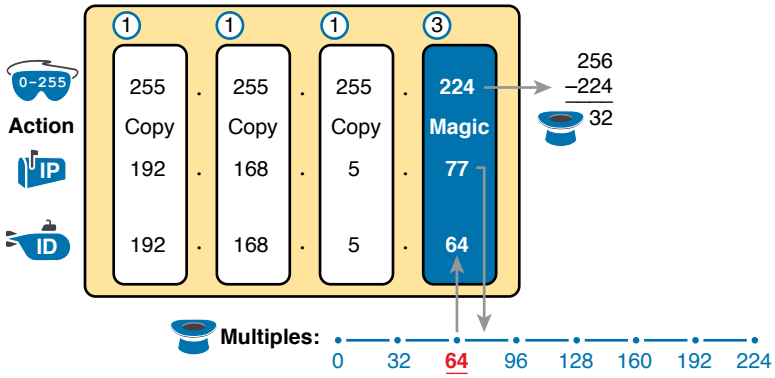


Figure 14-10 *Resident Subnet for 192.168.5.77, 255.255.255.224*

Resident Subnet Practice Problems

Before moving to the next section, take some time to fill in the blanks in Table 14-10. Check your answers against Table 14-16 in the section "Answers to Earlier Practice Problems," later in this chapter. Complete the table by listing the subnet ID in each case. The text following Table 14-16 also lists explanations for each problem.

Table 14-10 Practice Problems: Find Subnet ID, Difficult Masks

Problem	IP Address	Mask	Subnet ID
1	10.77.55.3	255.248.0.0	
2	172.30.99.4	255.255.192.0	
3	192.168.6.54	255.255.255.252	
4	10.77.3.14	255.255.128.0	
5	172.22.55.77	255.255.254.0	
6	1.99.53.76	255.255.255.248	

Finding the Subnet Broadcast Address: Difficult Masks

To find a subnet's broadcast address, a similar process can be used. For simplicity, this process begins with the subnet ID, rather than the IP address. If you happen to start with an IP address instead, use the processes in this chapter to first find the subnet ID, and then use the following process to find the subnet broadcast address for that same subnet. For each octet:

Key Topic

Step 1. If the mask octet = 255, copy the subnet ID.

Step 2. If the mask octet = 0, write 255.

Step 3. If the mask is neither, identify this octet as the *interesting octet*:

A. Calculate the *magic number* as 256 – mask.

B. Take the subnet ID's value, add the magic number, and subtract 1 (ID + magic – 1).

As with the similar process used to find the subnet ID, you have several options for how to best learn and internalize the process. If you can, stop reading now, use the companion website for this book, and watch the videos listed for this chapter. Also, look at the examples in this section, which show the process being used on paper. Then, follow the practice opportunities outlined in the section "Additional Practice for This Chapter's Processes."

Subnet Broadcast Example 1

The first example continues the first example from the section "Finding the Subnet ID: Difficult Masks," earlier in this chapter, as demonstrated in Figure 14-9. That example started with the IP address/mask of 130.4.102.1, 255.255.240.0, and showed how to find subnet ID 130.4.96.0. Figure 14-11 now begins with that subnet ID and the same mask.

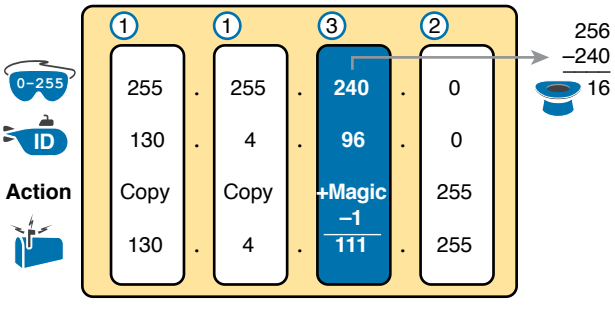


Figure 14-11 *Find the Subnet Broadcast: 130.4.96.0, 255.255.240.0*

First, examine the three uninteresting octets (1, 2, and 4). The process keys on the mask, and the first two octets have a mask value of 255, so simply copy the subnet ID to the place where you intend to write down the subnet broadcast address. The fourth octet has a mask value of 0, so write down a 255 for the fourth octet.

The logic related to the interesting octet occurs in the third octet in this example because of the mask value 240. First, Step 3A asks you to calculate the magic number, as 256 – mask. (If you had already calculated the subnet ID using the decimal process in this book, you should already know the magic number.) At Step 3B, you take the subnet ID's value (96), add the magic number (16), and subtract 1, for a total of 111. That makes the subnet broadcast address 130.4.111.255.

Subnet Broadcast Example 2

Again, this example continues an earlier example, from the section "Resident Subnet Example 2," as demonstrated in Figure 14-10. That example started with the IP address/mask of 192.168.5.77, mask 255.255.255.224 and showed how to find subnet ID 192.168.5.64. Figure 14-12 now begins with that subnet ID and the same mask.

First, examine the three uninteresting octets (1, 2, and 3). The process keys on the mask, and the first three octets have a mask value of 255, so simply copy the subnet ID to the place where you intend to write down the subnet broadcast address.

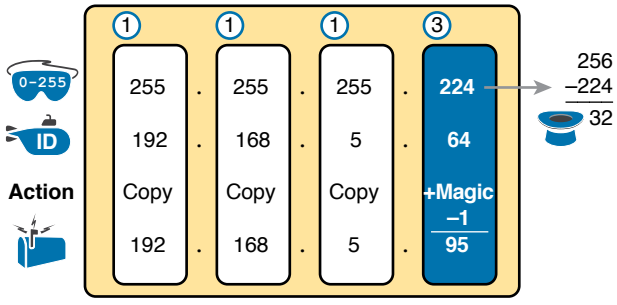


Figure 14-12 *Find the Subnet Broadcast: 192.168.5.64, 255.255.255.224*

The interesting logic occurs in the interesting octet, the fourth octet in this example, because of the mask value 224. First, Step 3A asks you to calculate the magic number, as 256 – mask. (If you had already calculated the subnet ID, it is the same magic number because the same mask is used.) At Step 3B, you take the subnet ID's value (64), add magic (32), and subtract 1, for a total of 95. That makes the subnet broadcast address 192.168.5.95.

Subnet Broadcast Address Practice Problems

Before moving to the next section, take some time to do several practice problems on a scratch piece of paper. Go back to Table 14-10, which lists IP addresses and masks, and practice by finding the subnet broadcast address for all the problems in that table. Then check your answers against Table 14-17 in the section "Answers to Earlier Practice Problems," later in this chapter.

Practice Analyzing Existing Subnets

As with the other subnetting math in this book, using a two-phase approach may help. Take time now to practice until you feel like you understand the process. Then, before the exam, make sure you master the math. Table 14-11 summarizes the key concepts and suggestions for this two-phase approach.

Table 14-11 Keep-Reading and Take-Exam Goals for This Chapter's Topics

	Before Moving to the Next Chapter	Before Taking the Exam
Focus On…	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	20–30 seconds

A Choice: Memorize or Calculate

As described in this chapter, the decimal processes to find the subnet ID and subnet broadcast address do require some calculation, including the calculation of the magic number (256 – mask). The processes also use a DDN mask, so if an exam question gives you a prefix-style mask, you need to convert to DDN format before using the process in this book.

Over the years, some people have told me they prefer to memorize a table to find the magic number. These tables could list the magic number for different DDN masks and prefix masks, so you avoid converting from the prefix mask to DDN. Table 14-12 shows an example of such a table. Feel free to ignore this table, use it, or make your own.

Table 14-12 Reference Table: DDN Mask Values, Binary Equivalent, Magic Numbers, and Prefixes

Prefix, interesting octet 2	/9	/10	/11	/12	/13	/14	/15	/16
Prefix, interesting octet 3	/17	/18	/19	/20	/21	/22	/23	/24
Prefix, interesting octet 4	/25	/26	/27	/28	/29	/30		
Magic number	128	64	32	16	8	4	2	1
DDN mask in the interesting octet	128	192	224	240	248	252	254	255

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 14-13 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 14-13 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website
Practice mask analysis		Website, Appendix F
Practice analyzing existing subnets		Website, Appendix F

Review All the Key Topics

Key Topic

Table 14-14 Key Topics for Chapter 14

Key Topic Element	Description	Page Number
List	Definition of a subnet's key numbers	322
Table 14-2	Key facts about the subnet ID	324
Table 14-3	Key facts about the subnet broadcast address	325
List	Steps to use binary math to find the subnet ID	328
List	General steps to use binary and decimal math to find the subnet ID	330
List	Steps to use decimal and binary math to find the subnet broadcast address	330
List	Steps to use only decimal math to find the subnet ID	334
List	Steps to use only decimal math to find the subnet broadcast address	336

Key Terms You Should Know

resident subnet, subnet ID, subnet number, subnet address, subnet broadcast address

Additional Practice for This Chapter's Processes

You can do more practice with the processes in this chapter with a pair of practice sets. Both give you practice at analyzing existing subnets. You may do each practice set using the following tools:

Application: Use the "Analyzing Existing Subnets" exercises 1 and 2 on the companion website, listed under the Chapter Review for this chapter.

PDF: Alternatively, practice the same problems found in these apps using companion website Appendix F, "Practice for Chapter 14: Analyzing Existing Subnets."

Answers to Earlier Practice Problems

This chapter includes practice problems spread around different locations in the chapter. The answers are located in Tables 14-15, 14-16, and 14-17.

Table 14-15 Answers to Problems in Table 14-9

	IP Address	Mask	Subnet ID	Broadcast Address
1	10.77.55.3	255.255.255.0	10.77.55.0	10.77.55.255
2	172.30.99.4	255.255.255.0	172.30.99.0	172.30.99.255
3	192.168.6.54	255.255.255.0	192.168.6.0	192.168.6.255
4	10.77.3.14	255.255.0.0	10.77.0.0	10.77.255.255
5	172.22.55.77	255.255.0.0	172.22.0.0	172.22.255.255
6	1.99.53.76	255.0.0.0	1.0.0.0	1.255.255.255

Table 14-16 Answers to Problems in Table 14-10

	IP Address	Mask	Subnet ID
1	10.77.55.3	255.248.0.0	10.72.0.0
2	172.30.99.4	255.255.192.0	172.30.64.0
3	192.168.6.54	255.255.255.252	192.168.6.52
4	10.77.3.14	255.255.128.0	10.77.0.0
5	172.22.55.77	255.255.254.0	172.22.54.0
6	1.99.53.76	255.255.255.248	1.99.53.72

The following list explains the answers for Table 14-16:

1. The second octet is the interesting octet, with magic number 256 – 248 = 8. The multiples of 8 include 0, 8, 16, 24, …, 64, 72, and 80. 72 is closest to the IP address value in that same octet (77) without going over, making the subnet ID 10.72.0.0.
2. The third octet is the interesting octet, with magic number 256 – 192 = 64. The multiples of 64 include 0, 64, 128, and 192. 64 is closest to the IP address value in that same octet (99) without going over, making the subnet ID 172.30.64.0.
3. The fourth octet is the interesting octet, with magic number 256 – 252 = 4. The multiples of 4 include 0, 4, 8, 12, 16, …, 48, 52, and 56. 52 is the closest to the IP address value in that same octet (54) without going over, making the subnet ID 192.168.6.52.
4. The third octet is the interesting octet, with magic number 256 – 128 = 128. Only two multiples exist that matter: 0 and 128. 0 is the closest to the IP address value in that same octet (3) without going over, making the subnet ID 10.77.0.0.
5. The third octet is the interesting octet, with magic number 256 – 254 = 2. The multiples of 2 include 0, 2, 4, 6, 8, and so on—essentially all even numbers. 54 is closest to the IP address value in that same octet (55) without going over, making the subnet ID 172.22.54.0.
6. The fourth octet is the interesting octet, with magic number 256 – 248 = 8. The multiples of 8 include 0, 8, 16, 24, …, 64, 72, and 80. 72 is closest to the IP address value in that same octet (76) without going over, making the subnet ID 1.99.53.72.

Table 14-17 Answers to Problems in the Section "Subnet Broadcast Address Practice Problems"

	Subnet ID	Mask	Broadcast Address
1	10.72.0.0	255.248.0.0	10.79.255.255
2	172.30.64.0	255.255.192.0	172.30.127.255
3	192.168.6.52	255.255.255.252	192.168.6.55
4	10.77.0.0	255.255.128.0	10.77.127.255
5	172.22.54.0	255.255.254.0	172.22.55.255
6	1.99.53.72	255.255.255.248	1.99.53.79

The following list explains the answers for Table 14-17:

1. The second octet is the interesting octet. Completing the three easy octets means that the broadcast address in the interesting octet will be 10.___.255.255. With magic number 256 – 248 = 8, the second octet will be 72 (from the subnet ID), plus 8, minus 1, or 79.
2. The third octet is the interesting octet. Completing the three easy octets means that the broadcast address in the interesting octet will be 172.30.___.255. With magic number 256 – 192 = 64, the interesting octet will be 64 (from the subnet ID), plus 64 (the magic number), minus 1, for 127.
3. The fourth octet is the interesting octet. Completing the three easy octets means that the broadcast address in the interesting octet will be 192.168.6.___. With magic number 256 – 252 = 4, the interesting octet will be 52 (the subnet ID value), plus 4 (the magic number), minus 1, or 55.
4. The third octet is the interesting octet. Completing the three easy octets means that the broadcast address will be 10.77.___.255. With magic number 256 – 128 = 128, the interesting octet will be 0 (the subnet ID value), plus 128 (the magic number), minus 1, or 127.
5. The third octet is the interesting octet. Completing the three easy octets means that the broadcast address will be 172.22.___.255. With magic number 256 – 254 = 2, the broadcast address in the interesting octet will be 54 (the subnet ID value), plus 2 (the magic number), minus 1, or 55.
6. The fourth octet is the interesting octet. Completing the three easy octets means that the broadcast address will be 1.99.53.___. With magic number 256 – 248 = 8, the broadcast address in the interesting octet will be 72 (the subnet ID value), plus 8 (the magic number), minus 1, or 79.

This page intentionally left blank

Part IV Review

Keep track of your part review progress with the checklist in Table P4-1. Details on each task follow the table.

Table P4-1 Part IV Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Subnetting Exercises in Appendices on Companion Website		
Videos on Companion Website		
Subnetting Exercises on Author's Blog		
Subnetting Exercises in IP Subnetting Practice Question Kit		
Subnetting Labs in Pearson Network Simulator		

Repeat All DIKTA Questions

For this task, use the PCPT software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PCPT to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Watch Videos

Chapter 14 recommends several videos as listed on this book's companion website. These videos help you understand how to use the process in the book to find facts about subnets, like the range of usable addresses in the subnet.

Subnetting Exercises

Chapters 12, 13, and 14 list some subnetting exercises, along with time and accuracy goals. Now is a good time to work on those goals. Some options include the following:

Practice from this book's appendices or web applications: The Chapter Review sections of Chapters 12, 13, and 14 mention addressing and subnetting exercises included with this book. Find all the related applications in the Part IV Review section of the companion website:

Appendix D, "Practice for Chapter 12: Analyzing Classful IPv4 Networks"

Appendix E, "Practice for Chapter 13: Analyzing Subnet Masks"

Appendix F, "Practice for Chapter 14: Analyzing Existing Subnets"

Pearson Network Simulator: The full Pearson ICND1 or CCNA simulator has subnetting math exercises that you can do by using CLI commands. Look for the labs with "IP Address Rejection" and "Subnet ID Calculation" in their names.

Author's blog: I've written a few dozen subnetting exercises on the blog over the years. Just look at the Questions menu item at the top of the page, and you will see a variety of IPv4 addressing and subnetting question types. Start at http://blog.certskills.com.

Parts V and VI work together to reveal the details of how to implement IPv4 routing in Cisco routers. To that end, Part V focuses on the most common features for Cisco routers, including IP address configuration, connected routes, and static routes. Part VI then goes into some detail about the one IP routing protocol discussed in this book: OSPF Version 2 (OSPFv2).

Part V follows a progression of topics. First, Chapter 15 examines the fundamentals of routers—the physical components, how to access the router command-line interface (CLI), and the configuration process. Chapter 15 makes a close comparison of the switch CLI and its basic administrative commands so that you have to learn only new commands that apply to routers but not to switches.

Chapter 16 then moves on to discuss how to configure routers to route IPv4 packets in the most basic designs. Those designs require a simple IP address/mask configuration on each interface, with the addition of a static route command—a command that directly configures a route into the IP routing table—for each destination subnet.

By the end of Chapter 16, you should have a solid understanding of how to enable IP addressing and routing in a Cisco router, so Chapter 17 continues the progression into more challenging but more realistic configurations related to routing between subnets in a LAN environment. Most LANs use many VLANs, with one subnet per VLAN. Cisco routers and switches can be configured to route packets between those subnets, with more than a few twists in the configuration.

Finally, Part V closes with a chapter about troubleshooting IPv4 routing. The chapter features the **ping** and **traceroute** commands, two commands that can help you discover not only whether a routing problem exists but also where the problem exists. Chapters 15, 16, and 17 show how to confirm whether a route has been added to one router's routing table, while the commands discussed in Chapter 18 teach you how to test the end-to-end routes from sending host to receiving host.

Part V

IPv4 Routing

Chapter 15: Operating Cisco Routers

Chapter 16: Configuring IPv4 Addressing and Static Routes

Chapter 17: IP Routing in the LAN

Chapter 18: Troubleshooting IPv4 Routing

Part V Review

CHAPTER 15

Operating Cisco Routers

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.a Routers

1.2 Describe characteristics of network topology architectures

1.2.e Small office/home office (SOHO)

1.6 Configure and verify IPv4 addressing and subnetting

Getting an IPv4 network up and working requires some basic steps: installing routers, installing cables, and ordering WAN services. The installation also requires some router configuration because routers often use defaults so that the router does not route IP packets until configuration has been added. You will need to configure IPv4 addresses, enable interfaces, and add IP routes—either through static configuration or by enabling some dynamic routing protocol. This chapter focuses on the first steps to creating a small working network: how to install an enterprise-class Cisco router and configure interfaces and IP addresses.

This chapter breaks the topics into two major headings. The first discusses the physical installation of an enterprise-class Cisco router. The second section looks at the command-line interface (CLI) on a Cisco router, which has the same look and feel as the Cisco switch CLI. This section first lists the similarities between a switch and router CLI and then introduces the configuration required to make the router start forwarding IP packets on its interfaces.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 15-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Installing Cisco Routers	1
Enabling IPv4 Support on Cisco Routers	2–6

1. Which of the following installation steps are more likely required on a Cisco router, but not typically required on a Cisco switch? (Choose two answers.)
   a. Connect Ethernet cables
   b. Connect serial cables
   c. Connect to the console port
   d. Connect the power cable
   e. Turn the on/off switch to "on"
2. Which of the following commands might you see associated with a router CLI, but not with a switch CLI?
   a. The **show mac address-table** command
   b. The **show ip route** command
   c. The **show running-config** command
   d. The **show interfaces status** command
3. Which answers list a task that could be helpful in making a router interface G0/0 ready to route packets? (Choose two answers.)
   a. Configuring the **ip address** *address mask* command in G0/0 configuration mode
   b. Configuring the **ip address** *address* and **ip mask** *mask* commands in G0/0 configuration mode
   c. Configuring the **no shutdown** command in G0/0 configuration mode
   d. Setting the interface **description** in G0/0 configuration mode
4. The output of the **show ip interface brief** command on R1 lists interface status codes of "down" and "down" for interface GigabitEthernet 0/0. The interface connects to a LAN switch with a UTP straight-through cable. Which of the following could be true?
   a. The **shutdown** command is currently configured for router interface G0/0.
   b. The **shutdown** command is currently configured for the switch interface on the other end of the cable.
   c. The router was never configured with an **ip address** command on the interface.
   d. The router was configured with the **no ip address** command.
5. Which of the following commands do not list the IP address and mask of at least one interface? (Choose two answers.)
   a. **show running-config**
   b. **show protocols** *type number*
   c. **show ip interface brief**
   d. **show interfaces**
   e. **show version**

6. Which of the following is different on the Cisco switch CLI for a Layer 2 switch as compared with the Cisco router CLI?

   a. The commands used to configure simple password checking for the console
   b. The number of IP addresses configured
   c. The configuration of the device's hostname
   d. The configuration of an interface description

Foundation Topics

Installing Cisco Routers

Routers collectively provide the main feature of the network layer—the capability to forward packets end to end through a network. As introduced in Chapter 3, "Fundamentals of WANs and IP Routing," routers forward packets by connecting to various physical network links, like Ethernet LAN, Ethernet WAN, and serial WAN links, then using Layer 3 routing logic to choose where to forward each packet. As a reminder, Chapter 2, "Fundamentals of Ethernet LANs," covered the details of making those physical connections to Ethernet networks, while Chapter 3 covered the basics of cabling with WAN links.

This section examines some of the details of router installation and cabling, first from the enterprise perspective and then from the perspective of connecting a typical small office/home office (SOHO) to an ISP using high-speed Internet.

Installing Enterprise Routers

A typical enterprise network has a few centralized sites as well as lots of smaller remote sites. To support devices at each site (the computers, IP phones, printers, and other devices), the network includes at least one LAN switch at each site. In addition, each site has a router, which connects to the LAN switch and to some WAN link. The WAN link provides connectivity from each remote site, back to the central site, and to other sites through the connection to the central site.

Figures 15-1 and 15-2 show a couple of different kinds of network diagrams that might be used to represent an enterprise network. The style of Figure 15-1 supports discussions about Layer 3 topics, showing the subnet IDs, masks, and interface IP addresses in shorthand. The figure also keeps the physical and data-link details to a minimum with these conventions:

Ethernet LAN: Simple straight lines with one or more LAN switches implied but not shown.

Ethernet WAN: Shown as a straight line, often with a cloud over it, with some kind of Ethernet interface identifier shown by the router (in this case, G0/1/0 and G0/0/0, which refers to GigabitEthernet interfaces).

Serial WAN: A line with a crooked part in the middle (a "lightning bolt") represents a typical point-to-point serial link as introduced in Chapter 3.

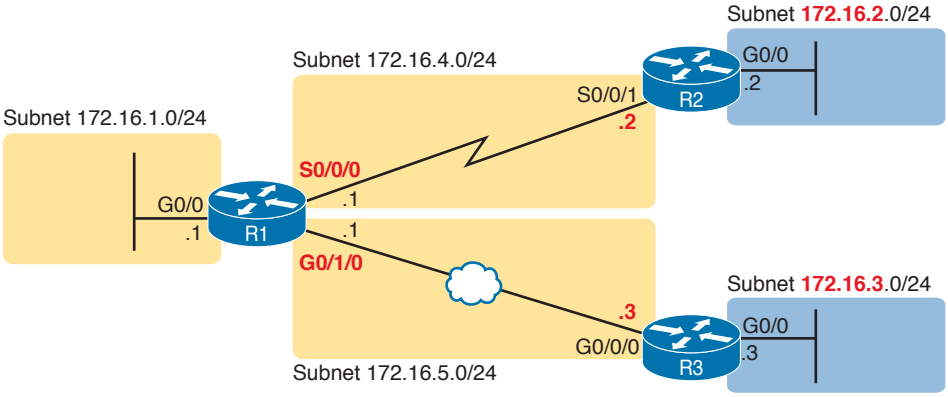


Figure 15-1 *Generic Enterprise Network Diagram*

In comparison, Figure 15-2 shows more detail about the physical cabling with less detail about the IP subnets and addresses. First, if the diagram needs to show physical details in the LAN, the diagram could show the LAN switches and related devices to the outside of the figure. The router Ethernet interfaces have an RJ-45 connector; just connect the appropriate UTP cable to both the router and the nearby LAN switch.

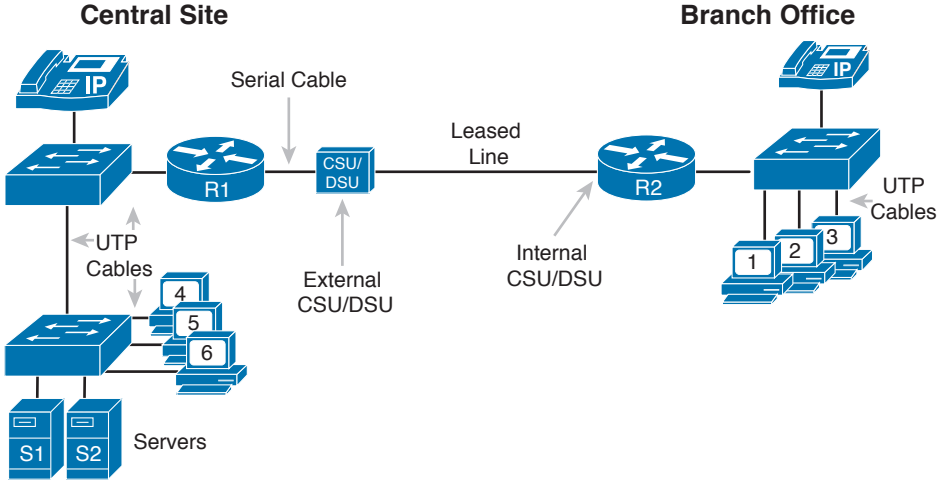


Figure 15-2 *More Detailed Cabling Diagram for the Same Enterprise Network*

Next, consider the hardware on the ends of the serial link, in particular where the channel service unit/data service unit (CSU/DSU) hardware resides on each end of the serial link. In a real serial link that runs through a service provider, the link terminates at a CSU/DSU. The CSU/DSU can either sit outside the router as a separate device (as shown on the left at router R1) or integrated into the router's serial interface hardware (as shown on the right).

As for cabling, the service provider will run the cable into the enterprise's wiring closet and often put an RJ-48 connector (same size as an RJ-45 connector) on the end of the cable. That cable should connect to the CSU/DSU. With an internal CSU/DSU (as with router R2 in Figure 15-2), the router serial port has an RJ-48 port to which the serial cable should

connect. With an external CSU/DSU, the CSU/DSU must be connected to the router's serial card via a short serial cable.

Cisco Integrated Services Routers

Product vendors, including Cisco, typically provide several different types of router hardware. Today, routers often do much more work than simply routing packets; in fact, they serve as a device or platform from which to provide many network services. Cisco even brands its enterprise routers not just as routers, but as "integrated services routers," emphasizing the multipurpose nature of the products.

As an example, consider the networking functions needed at a typical branch office. A typical enterprise branch office needs a router for WAN/LAN connectivity, and a LAN switch to provide a high-performance local network and connectivity into the router and WAN. Many branches also need voice-over-IP (VoIP) services to support IP phones, and several security services as well. Plus, it is hard to imagine a site with users that does not have Wi-Fi access today. So, rather than require multiple separate devices at one site, as shown in Figure 15-2, Cisco offers single devices that act as both router and switch and provide other functions as well.

For the sake of learning and understanding the different functions, this book focuses on using a separate switch and separate router, which provides a much cleaner path for learning the basics.

Figure 15-3 shows a photo of the Cisco 4321 ISR, with some of the more important features highlighted. The top part of the figure shows a full view of the back of the router. This model comes with two built-in Gigabit Ethernet interfaces and two modular slots that allow you to add small cards called Network Interface Modules (NIMs). The bottom of the figure shows one example NIM (a NIM that provides two serial interfaces). The router has other items as well, including both an RJ-45 and USB console port.

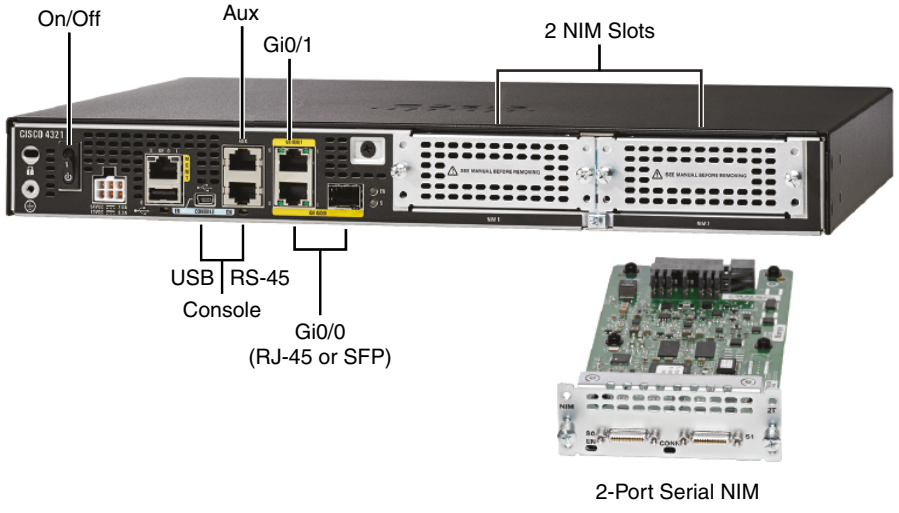


Figure 15-3 *Photos of a Model 4321 Cisco Integrated Services Router (ISR)*

Answers to the "Do I Know This Already?" quiz:

1 B, E **2** B **3** A, C **4** B **5** C, E **6** B

The figure shows an important feature for using routers to connect to both Ethernet LANs and Ethernet WAN services. Look closely at Figure 15-3's Gigabit interfaces. Gi0/1 refers to interface GigabitEthernet0/1 and is an RJ-45 port that supports UTP cabling only. However, interface Gi0/0 (short for GigabitEthernet0/0) has some interesting features:

- The router has two ports for one interface (Gi0/0).
- You can use one or the other at any point in time, but not both.
- One physical port is an RJ-45 port that supports copper cabling (implying that it is used to connect to a LAN).
- The other Gi0/0 physical port is a Small Form Pluggable (SFP) port that would support various fiber Ethernet standards, allowing the port to be used for Ethernet WAN purposes.

Cisco commonly makes one or more of the Ethernet ports on its Enterprise class routers support SFPs so that the engineer can choose an SFP that supports the type of Ethernet cabling provided by the Ethernet WAN service provider.

NOTE When building a lab network to study for CCNA or CCNP, because your devices will be in the same place, you can create Ethernet WAN links by using the RJ-45 ports and a UTP cable without the need to purchase an SFP for each router.

Physical Installation

Armed with the cabling details in images like Figure 15-2 and the router hardware details in photos like Figure 15-3, you can physically install a router. To install a router, follow these steps:

Key Topic

Step 1. For any Ethernet LAN interface, connect the RJ-45 connector of an appropriate copper Ethernet cable between the RJ-45 Ethernet port on the router and one of the LAN switch ports.

Step 2. For any serial WAN ports:

A. If using an external CSU/DSU, connect the router's serial interface to the CSU/DSU and the CSU/DSU to the line from the telco.

B. If using an internal CSU/DSU, connect the router's serial interface to the line from the telco.

Step 3. For any Ethernet WAN ports:

A. When ordering the Ethernet WAN service, confirm the required Ethernet standard and SFP type required to connect to the link, and order the SFPs.

B. Install the SFPs into the routers, and connect the Ethernet cable for the Ethernet WAN link to the SFP on each end of the link.

Step 4. Connect the router's console port to a PC (as discussed in Chapter 4, "Using the Command-Line Interface"), as needed, to configure the router.

Step 5. Connect a power cable from a power outlet to the power port on the router.

Step 6. Power on the router.

Note that Cisco enterprise routers typically have an on/off switch, while switches do not.

Installing SOHO Routers

The terms *enterprise router* and *small office/home office (SOHO) router* act as a pair of contrasting categories for routers, both in terms of how vendors like Cisco provide to the market, and how enterprises use and configure those devices. The term *enterprise router* typically refers to a router that a company would use in a permanent business location, while a *SOHO router* would reside at an employee's home or at a small permanent site with just a few people. However, as you might guess, the line between a router acting as an enterprise router and a SOHO router is blurry, so use these terms as general categories.

Even with that general comparison, SOHO routers typically have two features that an enterprise router would be less likely to have:

- SOHO routers almost always use the Internet and virtual private network (VPN) technology for their WAN connections to send data back and forth to the rest of the Enterprise.
- SOHO routers almost always use a multifunction device that does routing, LAN switching, VPN, wireless, and maybe other features.

For instance, at an enterprise business location, the building may contain enterprise routers, separate Ethernet switches, and separate wireless access points (AP), all connected together. At a permanent business site with four employees and 10 total devices in the network, one SOHO router could provide all those same features in one device.

For instance, Figure 15-4 shows a typical SOHO site. The three icons that represent a router, switch, and access point actually all exist inside one box; the figure just shows them separately to emphasize the fact that the one SOHO router provides several functions. On the left, the SOHO router provides wired and wireless LAN servers, and on the right, it provides WAN access through a cable Internet connection.

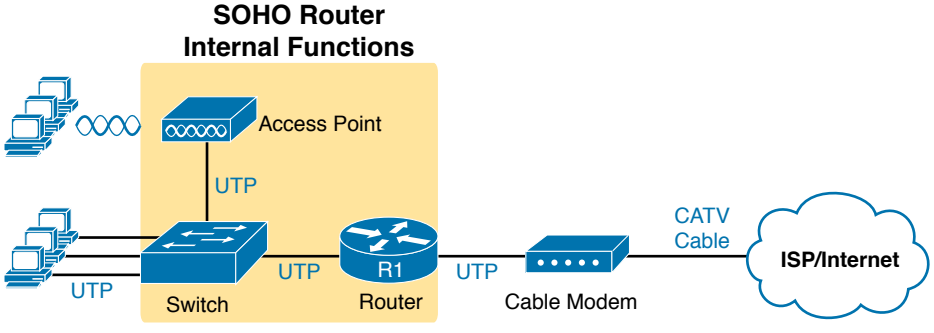


Figure 15-4 *Devices in a SOHO Network with High-Speed CATV Internet*

Figure 15-4 does not reflect the physical reality of a SOHO router, so Figure 15-5 shows one cabling example. The figure shows user devices on the left, connecting to the router via wireless or via Ethernet UTP cabling. On the right in this case, the router uses an external cable modem to connect to the coaxial cable provided by the ISP. Then the router must use a normal UTP Ethernet port to connect a short Ethernet cable between the SOHO router and the cable modem.

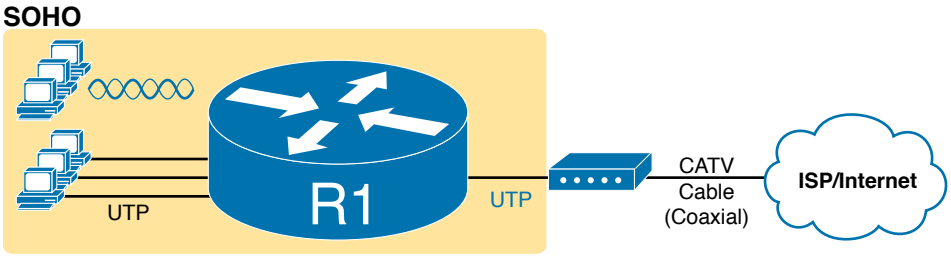


Figure 15-5 *SOHO Network, Using Cable Internet and an Integrated Device*

Enabling IPv4 Support on Cisco Router Interfaces

Routers support a relatively large number of features, with a large number of configuration and EXEC commands to support those features. You will learn about many of these features throughout the rest of this book.

NOTE For perspective, the Cisco router documentation includes a command reference, with an index to every single router command. A quick informal count of a recent IOS version listed around 5000 CLI commands.

This second section of the chapter focuses on commands related to router interfaces. To make routers work—that is, to route IPv4 packets—the interfaces must be configured. This section introduces the most common commands that configure interfaces, make them work, and give the interfaces IP addresses and masks.

Accessing the Router CLI

Accessing a router's command-line interface (CLI) works much like a switch. In fact, it works so much like accessing a Cisco switch CLI that this book relies on Chapter 4 instead of repeating the same details here. If the details from Chapter 4 are not fresh in your memory, it might be worthwhile to spend a few minutes briefly reviewing that chapter as well as Chapter 7, "Configuring and Verifying Switch Interfaces," before reading further.

Cisco switches and routers share many of the same CLI navigation features and many of the same configuration commands for management features. The following list mentions the highlights:

Key Topic

- User and Enable (privileged) mode
- Entering and exiting configuration mode, using the **configure terminal**, **end**, and **exit** commands and the Ctrl+Z key sequence
- Configuration of console, Telnet (vty), and enable secret passwords
- Configuration of Secure Shell (SSH) encryption keys and username/password login credentials
- Configuration of the hostname and interface description
- Configuration of Ethernet interfaces that can negotiate speed using the **speed** and **duplex** commands

- Configuration of an interface to be administratively disabled (**shutdown**) and administratively enabled (**no shutdown**)
- Navigation through different configuration mode contexts using commands like **line console 0** and **interface** *type number*
- CLI help, command editing, and command recall features
- The meaning and use of the startup-config (in NVRAM), running-config (in RAM), and external servers (like TFTP), along with how to use the **copy** command to copy the configuration files and IOS images

At first glance, this list seems to cover most everything you have read so far in this book about the switch CLI. However, a couple of topics do work differently with the router CLI as compared to the switch CLI, as follows:

- The configuration of IP addresses differs in some ways, with switches using a VLAN interface and routers using an IP address configured on each working interface.
- Many Cisco router models have an auxiliary (Aux) port, intended to be connected to an external modem and phone line to allow remote users to dial in to the router, and access the CLI, by making a phone call. Cisco switches do not have auxiliary ports.
- Router IOS defaults to disallow both Telnet and SSH into the router because of the typical router default setting of **transport input none** in vty configuration mode. (Cisco Catalyst LAN switches typically default to allow both Telnet and SSH.) Chapter 6, "Configuring Basic Switch Management," already discussed the various options on this command to enable Telnet (**transport input telnet**), SSH (**transport input ssh**), or both (**transport input all** or **transport input telnet ssh**).

The router CLI also differs from a switch CLI just because switches and routers do different things. For example:

- Cisco Layer 2 switches support the **show mac address-table** command, while Cisco routers do not.
- Cisco routers support the **show ip route** command, while Cisco Layer 2 switches do not.
- Cisco Layer 2 switches use the **show interfaces status** command to list one line of output per interface (and routers do not), while routers use the **show ip interface brief** command to list similar information (but switches do not).

Note also that some Cisco devices perform both Layer 2 switching and Layer 3 routing, and those devices support both router and switch commands. Chapter 17, "IP Routing in the LAN," discusses one such device, a Layer 3 switch, in more detail.

Router Interfaces

One minor difference between Cisco switches and routers is that routers support a much wider variety of interfaces. Today, LAN switches support Ethernet LAN interfaces of various speeds. Routers support a variety of other types of interfaces, including serial interfaces, cable TV, DSL, 3G/4G wireless, and others not mentioned in this book.

Most Cisco routers have at least one Ethernet interface of some type. Many of those Ethernet interfaces support multiple speeds and use autonegotiation, so for consistency, the

router IOS refers to these interfaces based on the fastest speed. For example, a 10-Mbps-only Ethernet interface would be configured with the **interface ethernet** *number* configuration command, a 10/100 interface with the **interface fastethernet** *number* command, and a 10/100/1000 interface with the **interface gigabitethernet** *number* command. However, when discussing these interfaces all together, engineers would simply call them *ethernet interfaces*, regardless of the maximum speed.

Some Cisco routers have serial interfaces. As you might recall from Chapter 3, Cisco routers use serial interfaces to connect to a serial link. Each point-to-point serial link can then use High-Level Data Link Control (HDLC, the default) or Point-to-Point Protocol (PPP).

Routers refer to interfaces in many commands, first by the type of interface (Ethernet, Fast Ethernet, Gigabit Ethernet, Serial, and so on) and then with a unique number of that router. Depending on the router model, the interface numbers might be a single number, two numbers separated by a slash, or three numbers separated by slashes. For example, all five of the following configuration commands are correct on at least one model of Cisco router:

```
interface ethernet 0
interface fastethernet 0/1
interface gigabitethernet 0/0
interface gigabitethernet 0/1/0
interface serial 1/0/1
```

Two of the most common commands to display the interfaces, and their status, are the **show ip interface brief** and **show interfaces** commands. The first of these commands displays a list with one line per interface, with some basic information, including the interface IP address and interface status. The second command lists the interfaces, but with a large amount of information per interface. Example 15-1 shows a sample of each command. The output comes from a 2900-series ISR router, used in many examples in this book; note that it has both a Gi0/0 interface and a Gi0/1/0 interface, showing a case with both two-digit and three-digit interface identifiers.

Example 15-1 *Listing the Interfaces in a Router*

```
R1# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
Embedded-Service-Engine0/0 unassigned      YES NVRAM  administratively down down
GigabitEthernet0/0         172.16.1.1      YES NVRAM  up                    up
GigabitEthernet0/1         unassigned      YES NVRAM  administratively down down
Serial0/0/0                172.16.4.1      YES manual up                    up
Serial0/0/1                unassigned      YES unset  administratively down down
GigabitEthernet0/1/0       172.16.5.1      YES NVRAM  up                    up

R1# show interfaces gigabitEthernet 0/1/0
GigabitEthernet0/1/0 is up, line protocol is up
  Hardware is EHWIC-1GE-SFP-CU, address is 0201.a010.0001 (bia 30f7.0d29.8570)
  Description: Link in lab to R3's G0/0/0
  Internet address is 172.16.5.1/24
```

```
MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
Full Duplex, 1Gbps, media type is RJ45
output flow-control is XON, input flow-control is XON
ARP type: ARPA, ARP Timeout 04:00:00
Last input 00:00:29, output 00:00:08, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
Queueing strategy: fifo
Output queue: 0/40 (size/max)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   12 packets input, 4251 bytes, 0 no buffer
   Received 12 broadcasts (0 IP multicasts)
   0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
   0 watchdog, 0 multicast, 0 pause input
   55 packets output, 8098 bytes, 0 underruns
   0 output errors, 0 collisions, 0 interface resets
   0 unknown protocol drops
   0 babbles, 0 late collision, 0 deferred
   0 lost carrier, 0 no carrier, 0 pause output
   0 output buffer failures, 0 output buffers swapped out
```

NOTE Commands that refer to router interfaces can be significantly shortened by truncating the words. For example, **sh int gi0/0** or **sh int g0/0** can be used instead of **show interfaces gigabitethernet 0/0**. In fact, many network engineers, when looking over someone's shoulder, would say something like "just do a show int G-i-oh-oh command" in this case, rather than speaking the long version of the command.

Also, note that the **show interfaces** command lists a text interface description on about the third line, if configured. In this case, interface G0/1/0 had been previously configured with the **description Link in lab to R3's G0/0/0** command in interface configuration mode for interface G0/1/0. The **description** interface subcommand provides an easy way to keep small notes about what router interfaces connect to which neighboring devices, with the **show interfaces** command listing that information.

Interface Status Codes

Each interface has two *interface status codes*. To be usable, the two interface status codes must be in an "up" state. The first status code refers essentially to whether Layer 1 is working, and the second status code mainly (but not always) refers to whether the data-link layer protocol is working. Table 15-2 summarizes these two status codes.

Key Topic

Table 15-2 Interface Status Codes and Their Meanings

Name	Location	General Meaning
Line status	First status code	Refers to the Layer 1 status. (For example, is the cable installed, is it the right/wrong cable, is the device on the other end powered on?)
Protocol status	Second status code	Refers generally to the Layer 2 status. It is always down if the line status is down. If the line status is up, a protocol status of down is usually caused by a mismatched data-link layer configuration.

Several combinations of interface status codes exist, as summarized in Table 15-3. The table lists the status codes in order, from being disabled on purpose by the configuration to a fully working state.

Key Topic

Table 15-3 Typical Combinations of Interface Status Codes

Line Status	Protocol Status	Typical Reasons
Administratively down	Down	The interface has a **shutdown** command configured on it.
Down	Down	The interface is not **shutdown**, but the physical layer has a problem. For example, no cable has been attached to the interface, or with Ethernet, the switch interface on the other end of the cable is shut down, or the switch is powered off, or the devices on the ends of the cable use a different transmission speed.
Up	Down	Almost always refers to data-link layer problems, most often configuration problems. For example, serial links have this combination when one router was configured to use PPP and the other defaults to use HDLC.
Up	Up	Layer 1 and Layer 2 of this interface are functioning.

For some examples, look back at Example 15-1's **show ip interface brief** command, to the three interfaces in the following list. The interfaces in this list each have a different combination of interface status codes; the list details the specific reasons for this status code in the lab used to create this example for the book.

G0/0: The interface is down/down, in this case because no cable was connected to the interface.

G0/1: The interface is administratively down/down, because the configuration includes the **shutdown** command under the G0/1 interface.

S0/0/0: The interface is up/up because a serial cable is installed, is connected to another router in a lab, and is working.

Router Interface IP Addresses

Cisco enterprise routers require at least some configuration beyond the default configuration before they will do their primary job: routing IP packets. The following facts tell us that to make a router ready to route IPv4 packets on an interface, you need to enable the interface and assign it an IPv4 address:

- Most Cisco router interfaces default to a disabled (**shutdown**) state and should be enabled with the **no shutdown** interface subcommand.
- Cisco routers do not route IP packets in or out an interface until an IP address and mask have been configured; by default, no interfaces have an IP address and mask.
- Cisco routers attempt to route IP packets for any interfaces that are in an up/up state and that have an IP address/mask assigned.

To configure the address and mask, simply use the **ip address** *address mask* interface subcommand. Figure 15-6 shows a simple IPv4 network with IPv4 addresses on Router R1, with Example 15-2 showing the matching configuration.

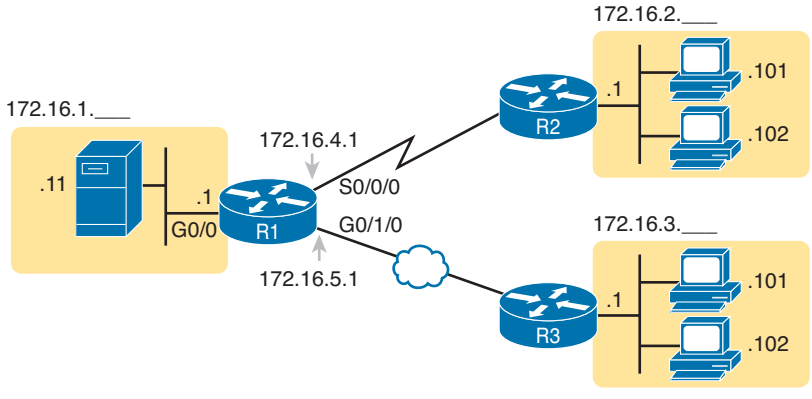


Figure 15-6 *IPv4 Addresses Used in Example 15-2*

Example 15-2 *Configuring IP Addresses on Cisco Routers*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1config)# interface G0/0
R1(config-if)# ip address 172.16.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface S0/0/0
R1(config-if)# ip address 172.16.4.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface G0/1/0
R1(config-if)# ip address 172.16.5.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# ^Z
R1#
```

Example 15-3 shows the output of the **show protocols** command. This command confirms the state of each of the three R1 interfaces in Figure 15-6 and the IP address and mask configured on those same interfaces.

Example 15-3 *Verifying IP Addresses on Cisco Routers*

```
R1# show protocols
Global values:
   Internet Protocol routing is enabled
Embedded-Service-Engine0/0 is administratively down, line protocol is down
GigabitEthernet0/0 is up, line protocol is up
   Internet address is 172.16.1.1/24
GigabitEthernet0/1 is administratively down, line protocol is down
Serial0/0/0 is up, line protocol is up
   Internet address is 172.16.4.1/24
Serial0/0/1 is administratively down, line protocol is down
GigabitEthernet0/1/0 is up, line protocol is up
   Internet address is 172.16.1.1/24
```

One of the first actions to take when verifying whether a router is working is to find the interfaces, check the interface status, and check to see whether the correct IP addresses and masks are used. Examples 15-1 and 15-3 showed samples of the key **show** commands, while Table 15-4 summarizes those commands and the types of information they display.

Key Topic

Table 15-4 Key Commands to List Router Interface Status

Command	Lines of Output per Interface	IP Configuration Listed	Interface Status Listed?
show ip interface brief	1	Address	Yes
show protocols [*type number*]	1 or 2	Address/mask	Yes
show interfaces [*type number*]	Many	Address/mask	Yes

Bandwidth and Clock Rate on Serial Interfaces

Cisco has included serial WAN topics in the CCNA exam topic list since its inception in 1998 until the CCNA 200-301 release in the year 2019. Because the CCNA 200-301 exam is the first to not mention serial technologies at all, this book includes some examples that show serial links. The exam might show them with the expectation that you at least understand basics, such as the fact that two routers can send data over a serial link if the router interfaces on both ends are up/up and the routers have IP addresses in the same subnet.

However, some of you will want to make serial links work in a lab because you have some serial interface cards in your lab. If so, take the time to look at a few pages in the section titled "Bandwidth and Clock Rate on Serial Interfaces," in Appendix J, "Topics from Previous Editions," which shows how to cable and configure a WAN serial link in the lab.

Router Auxiliary Port

Both routers and switches have a console port to allow administrative access, but most Cisco routers have an extra physical port called an auxiliary (Aux) port. The Aux port typically serves as a means to make a phone call to connect into the router to issue commands from the CLI.

The Aux port works like the console port, except that the Aux port is typically connected through a cable to an external analog modem, which in turn connects to a phone line. Then, the engineer uses a PC, terminal emulator, and modem to call the remote router. After being connected, the engineer can use the terminal emulator to access the router CLI, starting in user mode as usual.

Aux ports can be configured beginning with the **line aux 0** command to reach aux line configuration mode. From there, all the commands for the console line, covered mostly in Chapter 6, can be used. For example, the **login** and **password** *password* subcommands on the aux line could be used to set up simple password checking when a user dials in.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 15-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 15-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Review memory tables		Website
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 15-6 Key Topics for Chapter 15

Key Topic	Description	Page Number
List	Steps required to install a router	353
List	Similarities between a router CLI and a switch CLI	355
List	Items covered for switches in Chapters 4 and 6 that differ in some way on routers	356
Table 15-2	Router interface status codes and their meanings	359
Table 15-3	Combinations of the two interface status codes and the likely reasons for each combination	359
Table 15-4	Commands useful to display interface IPv4 addresses, masks, and interface status	361

Key Terms You Should Know

enterprise router, SOHO router, Integrated Services Router (ISR)

Command References

Tables 15-7 and 15-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 15-7 Chapter 15 Configuration Command Reference

Command	Description
interface *type number*	Global command that moves the user into configuration mode of the named interface.
ip address *address mask*	Interface subcommand that sets the router's IPv4 address and mask.
[no] shutdown	Interface subcommand that enables (**no shutdown**) or disables (**shutdown**) the interface.
duplex {**full** | **half** | **auto**}	Interface command that sets the duplex, or sets the use of IEEE autonegotiation, for router LAN interfaces that support multiple speeds.
speed {**10** | **100** | **1000**}	Interface command for router Gigabit (10/100/1000) interfaces that sets the speed at which the router interface sends and receives data.
description *text*	An interface subcommand with which you can type a string of text to document information about that particular interface.

Table 15-8 Chapter 15 EXEC Command Reference

Command	Purpose
show interfaces [*type number*]	Lists a large set of informational messages about each interface, or about the one specifically listed interface.
show ip interface brief	Lists a single line of information about each interface, including the IP address, line and protocol status, and the method with which the address was configured (manual or Dynamic Host Configuration Protocol [DHCP]).
show protocols [*type number*]	Lists information about the listed interface (or all interfaces if the interface is omitted), including the IP address, mask, and line/protocol status.

This page intentionally left blank

CHAPTER 16

Configuring IPv4 Addresses and Static Routes

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

3.0 IP Connectivity

3.1 Interpret the components of routing table

3.1.a Routing protocol code

3.1.b Prefix

3.1.c Network mask

3.1.d Next hop

3.1.e Administrative distance

3.1.f Metric

3.1.g Gateway of last resort

3.2 Determine how a router makes a forwarding decision by default

3.2.a Longest match

3.2.b Administrative distance

3.3 Configure and verify IPv4 and IPv6 static routing

3.3.a Default route

3.3.b Network route

3.3.c Host route

3.3.d Floating static

Routers route IPv4 packets. That simple statement actually carries a lot of hidden meaning. For routers to route packets, routers follow a routing process. That routing process relies on information called IP routes. Each IP route lists a destination—an IP network, IP subnet, or some other group of IP addresses. Each route also lists instructions that tell the router where to forward packets sent to addresses in that IP network or subnet. For routers to do a good job of routing packets, routers need to have a detailed, accurate list of IP routes.

Routers use three methods to add IPv4 routes to their IPv4 routing tables. Routers first learn *connected routes*, which are routes for subnets attached to a router interface. Routers can

also use *static routes*, which are routes created through a configuration command (**ip route**) that tells the router what route to put in the IPv4 routing table. And routers can use a routing protocol, in which routers tell each other about all their known routes, so that all routers can learn and build routes to all networks and subnets.

This chapter examines IP routing in depth with the most straightforward routes that can be added to a router's routing table. The router starts with a detailed look at the IP packet routing (forwarding process)—a process that relies on each router having useful IP routes in their routing tables. The second section then examines connected routes, which are routes to subnets that exist on the interfaces connected to the local router. The third section then examines static routes, which are routes the network engineer configures directly. The chapter ends with a section that looks more specifically at the IP routing process in a router, how it matches packets to the routing table, and how to interpret all the details in the output of the **show ip route** command.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 16-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
IP Routing	1
Configuring Connected Routes	2
Configuring Static Routes	3–5
IP Forwarding with the Longest Prefix Match	6

1. Router R1 lists a route in its routing table. Which of the following answers list a fact from a route that the router uses when matching the packet's destination address? (Choose two answers.)
   - **a.** Mask
   - **b.** Next-hop router
   - **c.** Subnet ID
   - **d.** Outgoing interface

2. After configuring a working router interface with IP address/mask 10.1.1.100/26, which of the following routes would you expect to see in the output of the **show ip route** command? (Choose two answers.)
   a. A connected route for subnet 10.1.1.64 255.255.255.192
   b. A connected route for subnet 10.1.1.0 255.255.255.0
   c. A local route for host 10.1.1.100 255.255.255.192
   d. A local route for host 10.1.1.100 255.255.255.255
   e. A local route for host 10.1.1.64 255.255.255.255
3. An engineer configures a static IPv4 route on Router R1. Which of the following pieces of information should not be listed as a parameter in the configuration command that creates this static IPv4 route?
   a. The destination subnet's subnet ID
   b. The next-hop router's IP address
   c. The next-hop router's neighboring interface
   d. The subnet mask
4. Which of the following commands correctly configures a static route?
   a. **ip route 10.1.3.0 255.255.255.0 10.1.130.253**
   b. **ip route 10.1.3.0 serial 0**
   c. **ip route 10.1.3.0 /24 10.1.130.253**
   d. **ip route 10.1.3.0 /24 serial 0**
5. A network engineer configures the **ip route 10.1.1.0 255.255.255.0 s0/0/0** command on a router and then issues a **show ip route** command from enable mode. No routes for subnet 10.1.1.0/24 appear in the output. Which of the following could be true?
   a. The **ip route** command has incorrect syntax and was rejected in config mode.
   b. Interface s0/0/0 is down.
   c. The router has no up/up interfaces in Class A network 10.0.0.0.
   d. The **ip route** command is missing a next-hop router IP address.
6. A router lists the following partial output from the **show ip route** command. Out which interface will the router route packets destined to IP address 10.1.15.122?

```
     10.0.0.0/8 is variably subnetted, 8 subnets, 5 masks
O       10.1.15.100/32 [110/50] via 172.16.25.2, 00:00:04, GigabitEthernet0/0/0
O       10.1.15.64/26 [110/100] via 172.16.25.129, 00:00:09, GigabitEthernet0/1/0
O       10.1.14.0/23 [110/65] via 172.16.24.2, 00:00:04, GigabitEthernet0/2/0
O       10.1.15.96/27 [110/65] via 172.16.24.129, 00:00:09, GigabitEthernet0/3/0
O       0.0.0.0/0 [110/129] via 172.16.25.129, 00:00:09, GigabitEthernet0/0/0
```

   a. G0/0/0
   b. G0/1/0
   c. G0/2/0
   d. G0/3/0

Foundation Topics

IP Routing

IP routing—the process of forwarding IP packets—delivers packets across entire TCP/IP networks, from the device that originally builds the IP packet to the device that is supposed to receive the packet. In other words, IP routing delivers IP packets from the sending host to the destination host.

The complete end-to-end routing process relies on network layer logic on hosts and on routers. The sending host uses Layer 3 concepts to create an IP packet, forwarding the IP packet to the host's default gateway (default router). The process requires Layer 3 logic on the routers as well, by which the routers compare the destination address in the packet to their routing tables, to decide where to forward the IP packet next.

The routing process also relies on data-link and physical details at each link. IP routing relies on serial WAN links, Ethernet WAN links, Ethernet LANs, wireless LANs, and many other networks that implement data-link and physical layer standards. These lower-layer devices and protocols move the IP packets around the TCP/IP network by encapsulating and transmitting the packets inside data-link layer frames.

The previous two paragraphs summarize the key concepts about IP routing as introduced back in Chapter 3, "Fundamentals of WANs and IP Routing." Next, this section reviews IP routing, while taking the discussion another step or two deeper, taking advantage of the additional depth of knowledge discussed in all the earlier chapters in this book.

IPv4 Routing Process Reference

Because you already saw the basics back in Chapter 3, this section collects the routing process into steps for reference. The steps use many specific Ethernet LAN terms discussed in Parts II and III of this book and some IP addressing terms discussed in Part IV. The upcoming descriptions and example then discuss these summaries of routing logic to make sure that each step is clear.

The routing process starts with the host that creates the IP packet. First, the host asks the question: Is the destination IP address of this new packet in my local subnet? The host uses its own IP address/mask to determine the range of addresses in the local subnet. Based on its own opinion of the range of addresses in the local subnet, a LAN-based host acts as follows:

Key Topic

Step 1. If the destination is local, send directly:

- **A.** Find the destination host's MAC address. Use the already-known Address Resolution Protocol (ARP) table entry, or use ARP messages to learn the information.
- **B.** Encapsulate the IP packet in a data-link frame, with the destination data-link address of the destination host.

Step 2. If the destination is not local, send to the default gateway:

- **A.** Find the default gateway's MAC address. Use the already-known Address Resolution Protocol (ARP) table entry, or use ARP messages to learn the information.
- **B.** Encapsulate the IP packet in a data-link frame, with the destination data-link address of the default gateway.

Figure 16-1 summarizes these same concepts. In the figure, host A sends a local packet directly to host D. However, for packets to host B, on the other side of a router and therefore in a different subnet, host A sends the packet to its default router (R1). (As a reminder, the terms *default gateway* and *default router* are synonyms.)

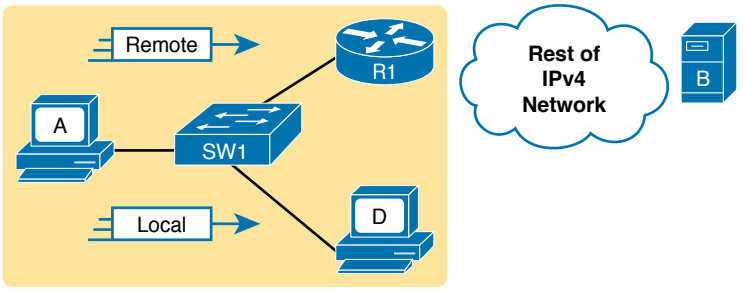


Figure 16-1 *Host Routing Logic Summary*

Routers have a little more routing work to do as compared with hosts. While the host logic began with an IP packet sitting in memory, a router has some work to do before getting to that point. With the following five-step summary of a router's routing logic, the router takes the first two steps just to receive the frame and extract the IP packet, before thinking about the packet's destination address at Step 3. The steps are as follows:

Key Topic

1. For each received data-link frame, choose whether or not to process the frame. Process it if
   - **A.** The frame has no errors (per the data-link trailer Frame Check Sequence [FCS] field).
   - **B.** The frame's destination data-link address is the router's address (or an appropriate multicast or broadcast address).
2. If choosing to process the frame at Step 1, de-encapsulate the packet from inside the data-link frame.
3. Make a routing decision. To do so, compare the packet's destination IP address to the routing table and find the route that matches the destination address. This route identifies the outgoing interface of the router and possibly the next-hop router.
4. Encapsulate the packet into a data-link frame appropriate for the outgoing interface. When forwarding out LAN interfaces, use ARP as needed to find the next device's MAC address.
5. Transmit the frame out the outgoing interface, as listed in the matched IP route.

This routing process summary lists many details, but sometimes you can think about the routing process in simpler terms. For example, leaving out some details, this paraphrase of the step list details the same big concepts:

> The router receives a frame, removes the packet from inside the frame, decides where to forward the packet, puts the packet into another frame, and sends the frame.

Answers to the "Do I Know This Already?" quiz:

1 A, C **2** A, D **3** C **4** A **5** B **6** D

To give you a little more perspective on these steps, Figure 16-2 breaks down the same five-step routing process as a diagram. The figure shows a packet arriving from the left, entering a router Ethernet interface, with an IP destination of host C. The figure shows the packet arriving, encapsulated inside an Ethernet frame (both header and trailer).

Key Topic

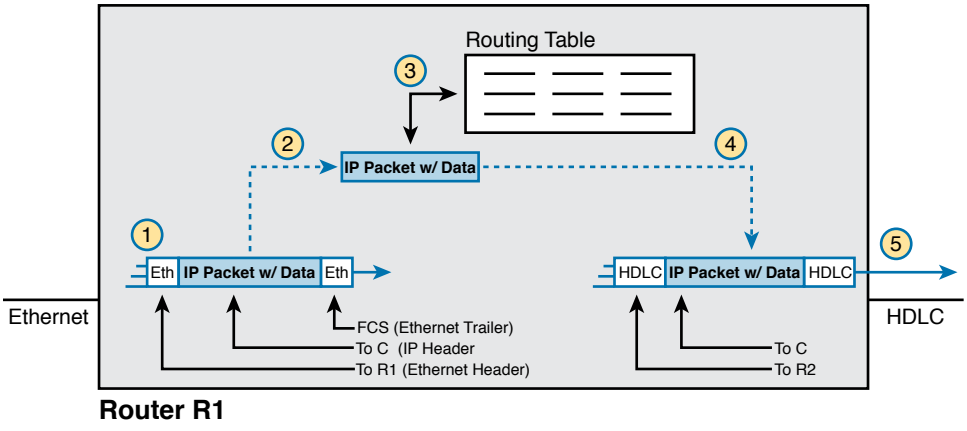


Figure 16-2 *Router Routing Logic Summary*

Router R1 processes the frame and packet as shown with the numbers in the figure, matching the same five-step process described just before the figure, as follows:

1. Router R1 notes that the received Ethernet frame passes the FCS check and that the destination Ethernet MAC address is R1's MAC address, so R1 processes the frame.
2. R1 de-encapsulates the IP packet from inside the Ethernet frame's header and trailer.
3. R1 compares the IP packet's destination IP address to R1's IP routing table.
4. R1 encapsulates the IP packet inside a new data-link frame, in this case, inside a High-Level Data Link Control (HDLC) header and trailer.
5. R1 transmits the IP packet, inside the new HDLC frame, out the serial link on the right.

NOTE This chapter uses several figures that show an IP packet encapsulated inside a data-link layer frame. These figures often show both the data-link header as well as the data-link trailer, with the IP packet in the middle. The IP packets all include the IP header, plus any encapsulated data.

An Example of IP Routing

The next several pages walk you through an example that discusses each routing step, in order, through multiple devices. The example uses a case in which host A (172.16.1.9) sends a packet to host B (172.16.2.9), with host routing logic and the five steps showing how R1 forwards the packet.

Figure 16-3 shows a typical IP addressing diagram for an IPv4 network with typical address abbreviations. The diagram can get a little too messy if it lists the full IP address for every router interface. When possible, these diagrams usually list the subnet and then the last octet or two of the individual IP addresses—just enough so that you know the IP address but with

less clutter. For example, host A uses IP address 172.16.1.9, taking from subnet 172.16.1.0/24 (in which all addresses begin 172.16.1) and the .9 beside the host A icon. As another example, R1 uses address 172.16.1.1 on its LAN interface, 172.16.4.1 on one serial interface, and 172.16.5.1 on an Ethernet WAN interface.

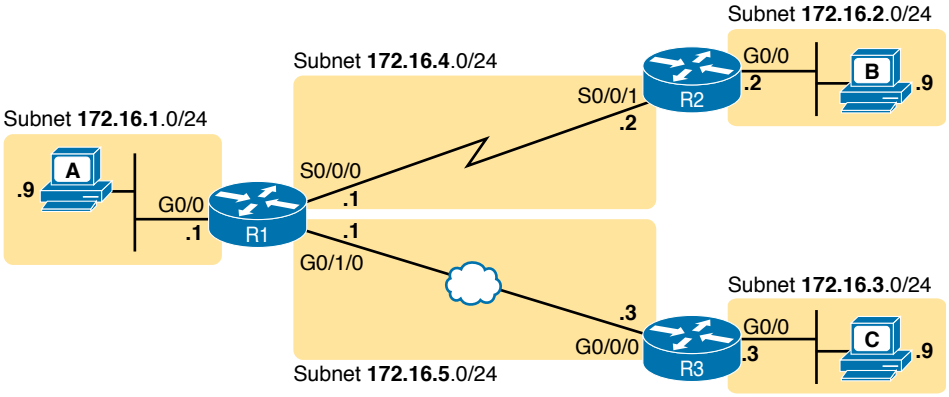


Figure 16-3 *IPv4 Network Used to Show Five-Step Routing Example*

Now on to the example, with host A (172.16.1.9) sending a packet to host B (172.16.2.9).

Host Forwards the IP Packet to the Default Router (Gateway)

In this example, host A uses some application that sends data to host B (172.16.2.9). After host A has the IP packet sitting in memory, host A's logic reduces to the following:

- My IP address/mask is 172.16.1.9/24, so my local subnet contains numbers 172.16.1.0–172.16.1.255 (including the subnet ID and subnet broadcast address).
- The destination address is 172.16.2.9, which is clearly not in my local subnet.
- Send the packet to my default gateway, which is set to 172.16.1.1.
- To send the packet, encapsulate it in an Ethernet frame. Make the destination MAC address be R1's G0/0 MAC address (host A's default gateway).

Figure 16-4 pulls these concepts together, showing the destination IP address and destination MAC address in the frame and packet sent by host A in this case. Note that the figure uses a common drawing convention in networking, showing an Ethernet as a few lines, hiding all the detail of the Layer 2 switches.

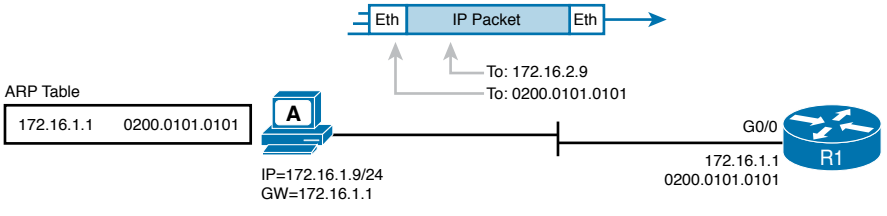


Figure 16-4 *Host A Sends Packet to Host B*

Routing Step 1: Decide Whether to Process the Incoming Frame

Routers receive many frames in an interface, particularly LAN interfaces. However, a router can and should ignore some of those frames. So, the first step in the routing process begins with a decision of whether a router should process the frame or silently discard (ignore) the frame.

First, the router does a simple but important check (Step 1A in the process summary) so that the router ignores all frames that had bit errors during transmission. The router uses the data-link trailer's FCS field to check the frame, and if errors occurred in transmission, the router discards the frame. (The router makes no attempt at error recovery; that is, the router does not ask the sender to retransmit the data.)

The router also checks the destination data-link address (Step 1B in the summary) to decide whether the frame is intended for the router. For example, frames sent to the router's unicast MAC address for that interface are clearly sent to that router. However, a router can actually receive a frame sent to some other unicast MAC address, and routers should ignore these frames.

For example, routers will receive some unicast frames sent to other devices in the VLAN just because of how LAN switches work. Think back to how LAN switches forward unknown unicast frames—frames for which the switch does not list the destination MAC address in the MAC address table. The LAN switch floods those frames. The result? Routers sometimes receive frames destined for some other device, with some other device's MAC address listed as the destination MAC address. Routers should ignore those frames.

In this example, host A sends a frame destined for R1's MAC address. So, after the frame is received, and after R1 confirms with the FCS that no errors occurred, R1 confirms that the frame is destined for R1's MAC address (0200.0101.0101 in this case). All checks have been passed, so R1 will process the frame, as shown in Figure 16-5. (Note that the large rectangle in the figure represents the internals of Router R1.)

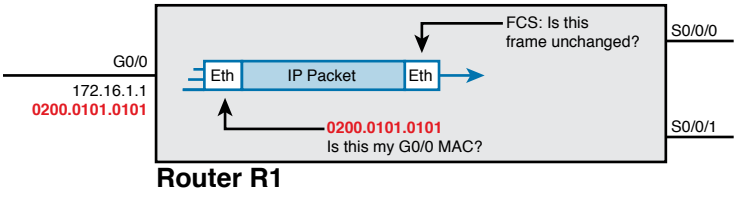


Figure 16-5 *Routing Step 1, on Router R1: Checking FCS and Destination MAC*

Routing Step 2: De-encapsulation of the IP Packet

After the router knows that it ought to process the received frame (per Step 1), the next step is relatively simple: de-encapsulating the packet. In router memory, the router no longer needs the original frame's data-link header and trailer, so the router removes and discards them, leaving the IP packet, as shown in Figure 16-6. Note that the destination IP address remains unchanged (172.16.2.9).

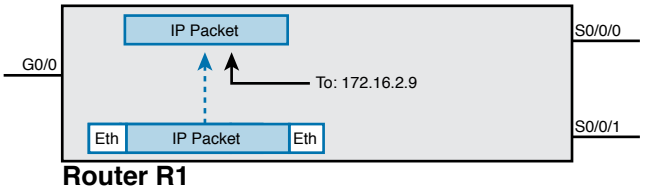


Figure 16-6 *Routing Step 2 on Router R1: De-encapsulating the Packet*

Routing Step 3: Choosing Where to Forward the Packet

While routing Step 2 required little thought, Step 3 requires the most thought of all the steps. At this point, the router needs to make a choice about where to forward the packet next. That process uses the router's IP routing table, with some matching logic to compare the packet's destination address with the table.

First, an IP routing table lists multiple routes. Each individual route contains several facts, which in turn can be grouped as shown in Figure 16-7. Part of each route is used to match the destination address of the packet, while the rest of the route lists forwarding instructions: where to send the packet next.

Key Topic

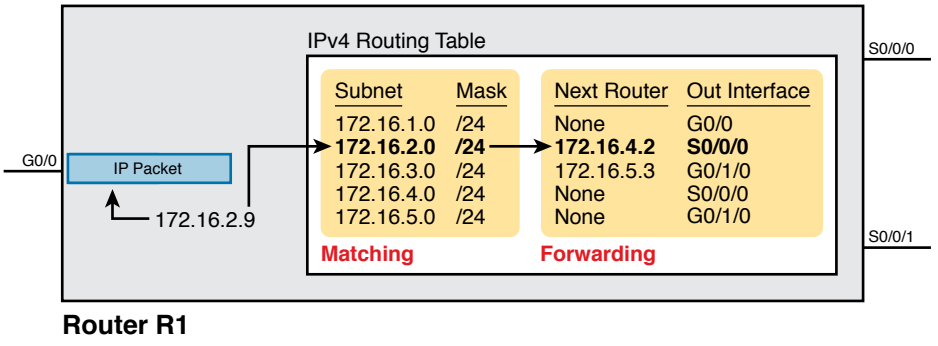


Figure 16-7 *Routing Step 3 on Router R1: Matching the Routing Table*

Focus on the entire routing table for a moment, and notice the fact that it lists five routes. Earlier, Figure 16-3 showed the entire example network, with five subnets, so R1 has a route for each of the five subnets.

Next, look at the part of the five routes that Router R1 will use to match packets. To fully define each subnet, each route lists both the subnet ID and the subnet mask. When matching the IP packet's destination with the routing table, the router looks at the packet's destination IP address (172.16.2.9) and compares it to the range of addresses defined by each subnet. Specifically, the router looks at the subnet and mask information; with a little math, the router can figure out in which of these subnets 172.16.2.9 resides (the route for subnet 172.16.2.0/24).

Finally, look to the right side of the figure, to the forwarding instructions for these five routes. After the router matches a specific route, the router uses the forwarding information in the route to tell the router where to send the packet next. In this case, the router matched the route for subnet 172.16.2.0/24, so R1 will forward the packet out its own interface S0/0/0, to Router R2 next, listed with its next-hop router IP address of 172.16.4.2.

NOTE Routes for remote subnets typically list both an outgoing interface and next-hop router IP address. Routes for subnets that connect directly to the router list only the outgoing interface because packets to these destinations do not need to be sent to another router.

Routing Step 4: Encapsulating the Packet in a New Frame

At this point, the router knows how it will forward the packet. However, routers cannot forward a packet without first wrapping a data-link header and trailer around it (encapsulation).

Encapsulating packets for serial links does not require a lot of thought, but the current CCNA 200-301 exam does not require a lot from us. Point-to-point serial WAN links use either HDLC (the default) or PPP as the data-link protocol. However, we can ignore any data-link logic, even ignoring data-link addressing, because serial links have only two devices on the link: the sender and the then-obvious receiver; the data-link addressing does not matter. In this example, R1 forwards the packet out S0/0/0, after encapsulating the packet inside an HDLC frame, as shown in Figure 16-8.

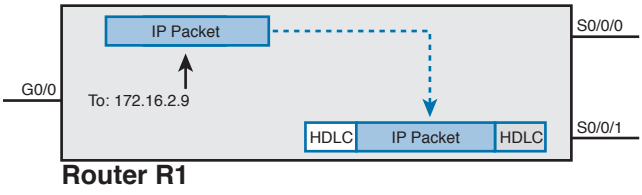


Figure 16-8 *Routing Step 4 on Router R1: Encapsulating the Packet*

Note that with some other types of data links, the router has a little more work to do at this routing step. For example, sometimes a router forwards packets out an Ethernet interface. To encapsulate the IP packet, the router would need to build an Ethernet header, and that Ethernet header's destination MAC address would need to list the correct value.

For example, consider a packet sent by that same PC A (172.16.1.9) in Figure 16-3 but with a destination of PC C (172.16.3.9). When R1 processes the packet, R1 matches a route that tells R1 to forward the packet out R1's G0/1/0 Ethernet interface to 172.16.5.3 (R3) next. R1 needs to put R3's MAC address in the header, and to do that, R1 uses its IP ARP table information, as shown in Figure 16-9. If R1 did not have an ARP table entry for 172.16.5.3, R1 would first have to use ARP to learn the matching MAC address.

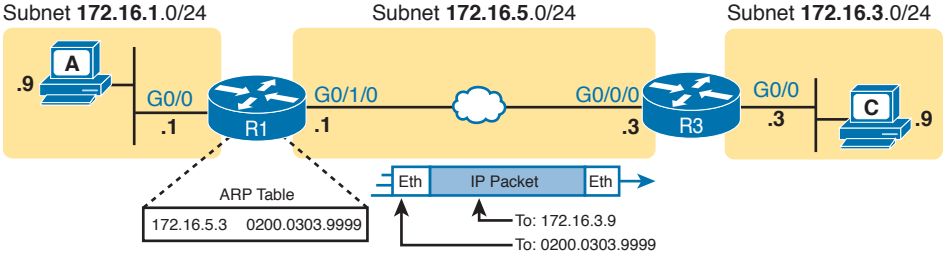


Figure 16-9 *Routing Step 4 on Router R1 with a LAN Outgoing Interface*

Routing Step 5: Transmitting the Frame

After the frame has been prepared, the router simply needs to transmit the frame. The router might have to wait, particularly if other frames are already waiting their turn to exit the interface.

Configuring IP Addresses and Connected Routes

Cisco routers enable IPv4 routing globally, by default. Then, to make the router be ready to route packets on a particular interface, the interface must be configured with an IP address and the interface must be configured such that it comes up, reaching a "line status up, line protocol up" state. Only at that point can routers route IP packets in and out a particular interface.

After a router can route IP packets out one or more interfaces, the router needs some routes. Routers can add routes to their routing tables through three methods:

Key Topic

Connected routes: Added because of the configuration of the **ip address** interface subcommand on the local router

Static routes: Added because of the configuration of the **ip route** global command on the local router

Routing protocols: Added as a function by configuration on all routers, resulting in a process by which routers dynamically tell each other about the network so that they all learn routes

This second of three sections discusses several variations on how to configure connected routes, while the next major section discusses static routes.

Connected Routes and the ip address Command

A Cisco router automatically adds a route to its routing table for the subnet connected to each interface, assuming that the following two facts are true:

Key Topic

- The interface is in a working state. In other words, the interface status in the **show interfaces** command lists a line status of up and a protocol status of up.
- The interface has an IP address assigned through the **ip address** interface subcommand.

The concept of connected routes is relatively basic. The router, of course, needs to know the subnet number connected to each of its interfaces, so the router can route packets to that subnet. The router does the math, taking the interface IP address and mask and calculating the subnet ID. However, the router only needs that route when the interface is up and working, so the router includes a connected route in the routing table only when the interface is working.

Example 16-1 shows the connected routes on Router R1 in Figure 16-10. The first part of the example shows the configuration of IP addresses on all three of R1's interfaces. The end of the example lists the output from the **show ip route** command, which lists these routes with a *c* as the route code, meaning *connected*.

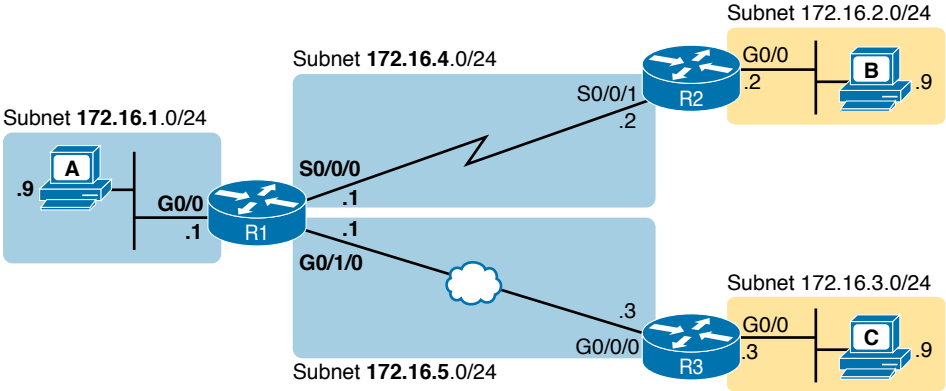


Figure 16-10 *Sample Network to Show Connected Routes*

Example 16-1 *Connected and Local Routes on Router R1*

```
! Excerpt from show running-config follows...
!
interface GigabitEthernet0/0
 ip address 172.16.1.1 255.255.255.0
!
interface Serial0/0/0
 ip address 172.16.4.1 255.255.255.0
!
interface GigabitEthernet0/1/0
 ip address 172.16.5.1 255.255.255.0

R1# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR

Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C        172.16.1.0/24 is directly connected, GigabitEthernet0/0
L        172.16.1.1/32 is directly connected, GigabitEthernet0/0
C        172.16.4.0/24 is directly connected, Serial0/0/0
L        172.16.4.1/32 is directly connected, Serial0/0/0
C        172.16.5.0/24 is directly connected, GigabitEthernet0/1/0
L        172.16.5.1/32 is directly connected, GigabitEthernet0/1/0
```

Take a moment to look closely at each of the three highlighted routes in the output of **show ip route**. Each lists a C in the first column, and each has text that says "directly connected"; both identify the route as connected to the router. The early part of each route lists the matching parameters (subnet ID and mask), as shown in the earlier example in Figure 16-7. The end of each of these routes lists the outgoing interface.

Note that the router also automatically produces a different kind of route, called a *local route*. The local routes define a route for the one specific IP address configured on the router interface. Each local route has a /32 prefix length, defining a *host route*, which defines a route just for that one IP address. For example, the last local route, for 172.16.5.1/32, defines a route that matches only the IP address of 172.16.5.1. Routers use these local routes that list their own local IP addresses to more efficiently forward packets sent to the router itself.

For the CCNA 200-301 exam, note that this example of the **show ip route** command reveals a few of the specific subitems within exam topic 3.1, with later examples revealing even more details. This section shows details related to the following terms from the exam topics:

- **Routing Protocol Code:** The legend at the top of the **show ip route** output (about nine lines) lists all the routing protocol codes (exam topic 3.1.a). This book references the codes for connected routes (C), local (L), static (S), and OSPF (O).
- **Prefix:** The word *prefix* (exam topic 3.1.b) is just another name for subnet ID.
- **Mask:** Each route lists a prefix (subnet ID) and network mask (exam topic 3.1.c) in prefix format, for example, /24.

The ARP Table on a Cisco Router

After a router has added these connected routes, the router can route IPv4 packets between those subnets. To do so, the router makes use of its IP ARP table.

The IPv4 ARP table lists the IPv4 address and matching MAC address of hosts connected to the same subnet as the router. When forwarding a packet to a host on the same subnet, the router encapsulates the packet, with a destination MAC address as found in the ARP table. If the router wants to forward a packet to an IP address on the same subnet as the router but does not find an ARP table entry for that IP address, the router will use ARP messages to learn that device's MAC address.

Example 16-2 shows R1's ARP table based on the previous example. The output lists R1's own IP address of 172.16.1.1, with an age of -, meaning that this entry does not time out. Dynamically learned ARP table entries have an upward counter, like the 35-minute value for the ARP table entry for IP address 172.16.1.9. By default, IOS will time out (remove) an ARP table entry after 240 minutes in which the entry is not used. (IOS resets the timer to 0 when an ARP table entry is used.) Note that to experiment in the lab, you might want to empty all dynamic entries (or a single entry for one IP address) using the **clear ip arp** [*ip-address*] EXEC command.

Example 16-2 *Displaying a Router's IP ARP Table*

```
R1# show ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  172.16.1.1             -   0200.2222.2222  ARPA   GigabitEthernet0/0
Internet  172.16.1.9            35   0200.3333.3333  ARPA   GigabitEthernet0/0
```

Thinking about how Router R1 forwards a packet to host A (172.16.1.9), over that final subnet, R1 does the following:

1. R1 looks in its ARP table for an entry for 172.16.1.9.
2. R1 encapsulates the IP packet in an Ethernet frame, adding destination 0200.3333.3333 to the Ethernet header (as taken from the ARP table).
3. R1 transmits the frame out interface G0/0.

Configuring Static Routes

All routers add connected routes, as discussed in the previous section. Then, most networks use dynamic routing protocols to cause each router to learn the rest of the routes in an internetwork. Networks use static routes—routes added to a routing table through direct configuration—much less often than dynamic routing. However, static routes can be useful at times, and they happen to be useful learning tools as well. This next major section in the chapter discusses static routes.

NOTE The CCNA 200-301 exam topic 3.2 breaks IPv4 (and IPv6) static routes into four subtopics: network routes, host routes, floating static routes, and default routes. This section explains all four types as noted in the upcoming headings.

Static Network Routes

IOS allows the definition of individual static routes using the **ip route** global configuration command. Every **ip route** command defines a destination that can be matched, usually with a subnet ID and mask. The command also lists the forwarding instructions, typically listing either the outgoing interface or the next-hop router's IP address. IOS then takes that information and adds that route to the IP routing table.

The static route is considered a *network route* when the destination listed in the **ip route** command defines a subnet, or an entire Class A, B, or C network. In contrast, a *default route* matches all destination IP addresses, while a *host route* matches a single IP address (that is, an address of one host.)

As an example of a network route, Figure 16-11 shows a subset of the figure used throughout this chapter so far, with some unrelated details removed. The figure shows only the details related to a static network route on R1, for destination subnet 172.16.2.0/24, which sits on the far right. To create that static network route on R1, R1 will configure the subnet ID and mask, and either R1's outgoing interface (S0/0/0) or R2 as the next-hop router IP address (172.16.4.2).

Key Topic

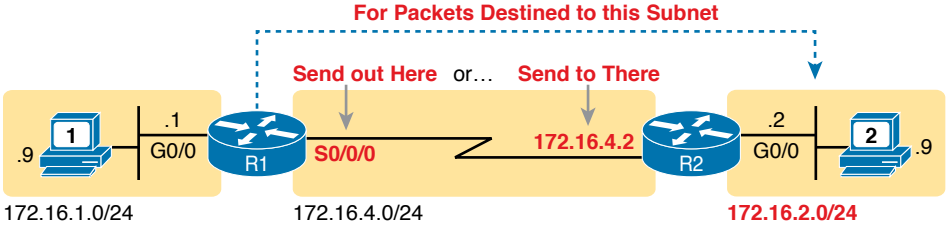


Figure 16-11 *Static Route Configuration Concept*

Example 16-3 shows the configuration of a couple of sample static routes. In particular, it shows routes on Router R1 in Figure 16-12, for the two subnets on the right side of the figure.

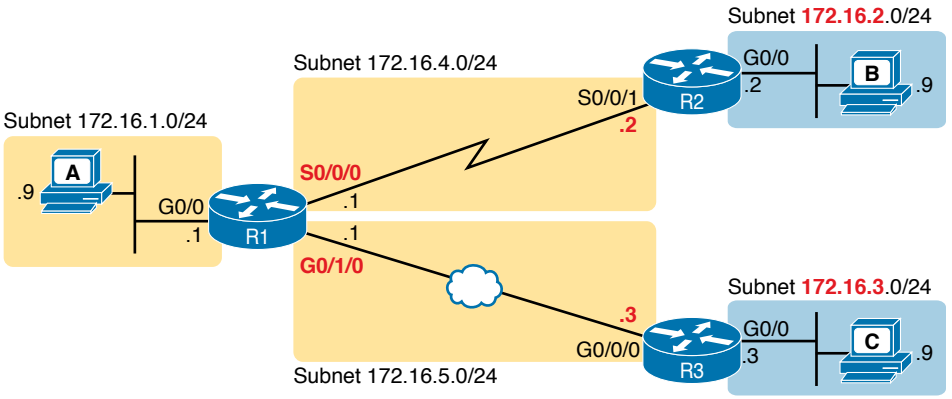


Figure 16-12 *Sample Network Used in Static Route Configuration Examples*

Example 16-3 *Static Routes Added to R1*

```
ip route 172.16.2.0 255.255.255.0 S0/0/0
ip route 172.16.3.0 255.255.255.0 172.16.5.3
```

The two example **ip route** commands show the two different styles of forwarding instructions. The first command shows subnet 172.16.2.0, mask 255.255.255.0, which sits on a LAN near Router R2. That same first command lists R1's S0/0/0 interface as the outgoing interface. This route basically states: To send packets to the subnet off Router R2, send them out my own local S0/0/0 interface (which happens to connect to R2).

The second route has the same kind of logic, except for using different forwarding instructions. Instead of referencing R1's outgoing interface, it instead lists the neighboring router's IP address on the WAN link as the next-hop router. This route basically says this: To send packets to the subnet off Router R3, send them to R3—specifically, R3's WAN IP address next.

The routes created by these two **ip route** commands actually look a little different in the IP routing table compared to each other. Both are static routes. However, the route that used the outgoing interface configuration is also noted as a connected route; this is just a quirk of the output of the **show ip route** command.

Example 16-4 lists these two routes using the **show ip route static** command. This command lists the details of static routes only, but it also lists a few statistics about all IPv4 routes. For example, the example shows two lines, for the two static routes configured in Example 16-4, but statistics state that this router has routes for eight subnets.

Example 16-4 *Static Routes Added to R1*

```
R1# show ip route static
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! lines omitted for brevity

Gateway of last resort is not set
```

```
      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
S        172.16.2.0/24 is directly connected, Serial0/0/0
S        172.16.3.0/24 [1/0] via 172.16.5.3
```

IOS adds and removes these static routes dynamically over time, based on whether the outgoing interface is working or not. For example, in this case, if R1's S0/0/0 interface fails, R1 removes the static route to 172.16.2.0/24 from the IPv4 routing table. Later, when the interface comes up again, IOS adds the route back to the routing table.

Note that most sites use a dynamic routing protocol to learn all the routes to remote subnets rather than using static routes. However, when not using a dynamic routing protocol, the engineer would need to configure static routes to each subnet on each router. For example, if the routers had only the configuration shown in the examples so far, PC A (from Figure 16-12) would not be able to receive packets back from PC B because Router R2 does not have a route for PC A's subnet. R2 would need static routes for other subnets, as would R3.

Finally, note that static routes that will send packets out an Ethernet interface—LAN or WAN—should use the next-hop IP address option on the **ip address** command, as shown in Example 16-4. Routers expect their Ethernet interfaces to be able to reach any number of other IP addresses in the connected subnet. Referencing the next-hop router identifies the specific device in the connected subnet, while referencing the local router's outgoing interface does not identify the specific neighboring router.

Static Host Routes

Earlier, this chapter defined a host route as a route to a single host address. To configure such a static route, the **ip route** command uses an IP address plus a mask of 255.255.255.255 so that the matching logic matches just that one address.

An engineer might use host routes to direct packets sent to one host over one path, with all other traffic to that host's subnet over some other path. For instance, you could define these two static routes for subnet 10.1.1.0/24 and host 10.1.1.9, with two different next-hop addresses, as follows:

```
ip route 10.1.1.0 255.255.255.0 10.2.2.2
ip route 10.1.1.9 255.255.255.255 10.9.9.9
```

Note that these two routes overlap: a packet sent to 10.1.1.9 that arrives at the router would match both routes. When that happens, routers use the most specific route (that is, the route with the longest prefix length). So, a packet sent to 10.1.1.9 would be forwarded to next-hop router 10.9.9.9, and packets sent to other destinations in subnet 10.1.1.0/24 would be sent to next-hop router 10.2.2.2.

Note that the section "IP Forwarding with the Longest Prefix Match" later in this chapter gets into this topic in more detail.

Floating Static Routes

Next, consider the case in which a static route competes with other static routes or routes learned by a routing protocol. That is, the **ip route** command defines a route to a subnet, but the router also knows of other static or dynamically learned routes to reach that same

subnet. In these cases, the router must first decide which routing source has the better *administrative distance*, with lower being better, and then use the route learned from the better source.

To see how that works, consider the example illustrated in Figure 16-13, which shows a different design than in the previous examples, this time with a branch office with two WAN links: one very fast Gigabit Ethernet link and one rather slow (but cheap) T1. In this design, the network uses Open Shortest Path First Version 2 (OSPFv2) over the primary link, learning a route for subnet 172.16.2.0/24. R1 also defines a static route over the backup link to that exact same subnet, so R1 must choose whether to use the static route or the OSPF-learned route.

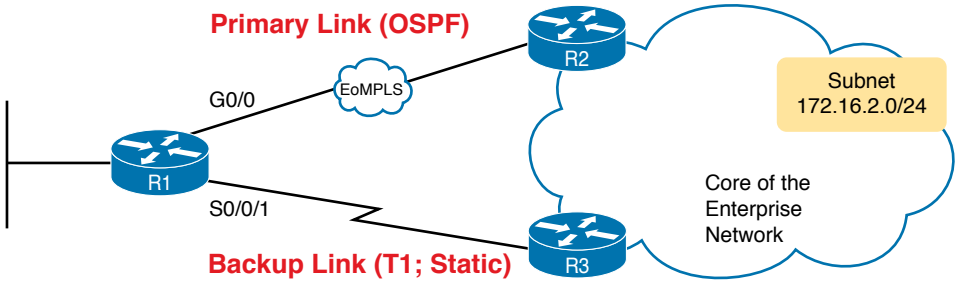


Figure 16-13 *Using a Floating Static Route to Key Subnet 172.16.2.0/24*

By default, IOS considers static routes better than OSPF-learned routes. By default, IOS gives static routes an administrative distance of 1 and OSPF routes an administrative distance of 110. Using these defaults in Figure 16-13, R1 would use the T1 to reach subnet 172.16.2.0/24 in this case, which is not the intended design. Instead, the engineer prefers to use the OSPF-learned routes over the much-faster primary link and use the static route over the backup link only as needed when the primary link fails.

To instead prefer the OSPF routes, the configuration would need to change the administrative distance settings and use what many networkers call a floating static route. A *floating static* route floats or moves into and out of the IP routing table depending on whether the better (lower) administrative distance route learned by the routing protocol happens to exist currently. Basically, the router ignores the static route during times when the better routing protocol route is known.

To implement a floating static route, you need to use a parameter on the **ip route** command that sets the administrative distance for just that route, making the value larger than the default administrative distance of the routing protocol. For example, the **ip route 172.16.2.0 255.255.255.0 172.16.5.3 130** command on R1 would do exactly that—setting the static route's administrative distance to 130. As long as the primary link stays up, and OSPF on R1 learns a route for 172.16.2.0/24, with a default administrative distance of 110, R1 ignores the static route.

Finally, note that while the **show ip route** command lists the administrative distance of most routes, as the first of two numbers inside two brackets, the **show ip route** *subnet* command plainly lists the administrative distance. Example 16-5 shows a sample, matching this most recent example.

Example 16-5 *Displaying the Administrative Distance of the Static Route*

```
R1# show ip route static
! Legend omitted for brevity
       172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
S         172.16.2.0/24 is directly connected, Serial0/0/1

R1# show ip route 172.16.2.0
Routing entry for 172.16.2.0/24
  Known via "static", distance 130, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0/1
      Route metric is 0, traffic share count is 1
```


Static Default Routes

When a router tries to route a packet, the router might not match the packet's destination IP address with any route. When that happens, the router normally just discards the packet.

Routers can be configured so that they use either a statically configured or dynamically learned default route. The *default route* matches all packets, so that if a packet does not match any other more specific route in the routing table, the router can at least forward the packet based on the default route.

One classic example in which companies might use static default routes in their enterprise TCP/IP networks is when the company has many remote sites, each with a single, relatively slow WAN connection. Each remote site has only one possible physical route to use to send packets to the rest of the network. So, rather than use a routing protocol, which sends messages over the WAN and uses precious WAN bandwidth, each remote router might use a default route that sends all traffic to the central site, as shown in Figure 16-14.

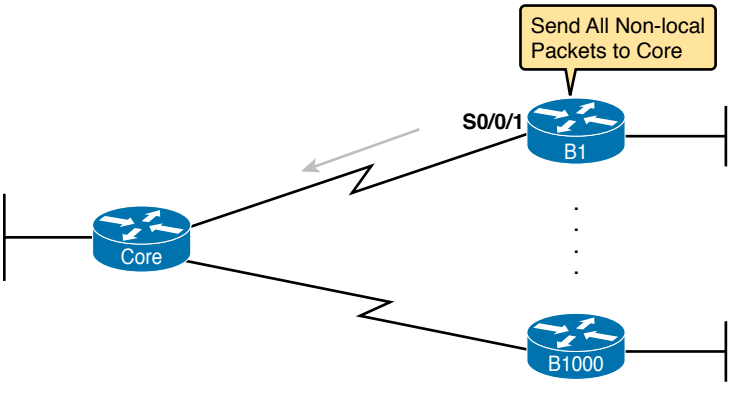


Figure 16-14 *Example Use of Static Default Routes at 1000 Low-Speed Remote Sites*

IOS allows the configuration of a static default route by using special values for the subnet and mask fields in the **ip route** command: 0.0.0.0 and 0.0.0.0. For example, the command **ip route 0.0.0.0 0.0.0.0 S0/0/1** creates a static default route on Router B1—a route that matches all IP packets—and sends those packets out interface S0/0/1.

Example 16-6 shows an example of a static default route, using Router R2 from Figure 16-12. Earlier, that figure, along with Example 16-5, showed R1 with static routes to the two subnets on the right side of the figure. Example 16-6 completes the configuration of static IP routes by configuring R2, on the right side of Figure 16-13, with a static default route to route packets back to the routers on the left side of the figure.

Example 16-6 *Adding a Static Default Route on R2 (Figure 16-13)*

```
R2# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)# ip route 0.0.0.0 0.0.0.0 s0/0/1
R2(config)# ^Z
R2# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is 0.0.0.0 to network 0.0.0.0

S* 0.0.0.0/0 is directly connected, Serial0/0/1
      172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        172.16.2.0/24 is directly connected, GigabitEthernet0/0
L        172.16.2.2/32 is directly connected, GigabitEthernet0/0
C        172.16.4.0/24 is directly connected, Serial0/0/1
L        172.16.4.2/32 is directly connected, Serial0/0/1
```

The output of the **show ip route** command lists a few new and interesting facts. First, it lists the route with a code of S, meaning static, but also with a *, meaning it is a *candidate default route*. A router can learn about more than one default route, and the router then has to choose which one to use; the * means that it is at least a candidate to become the default route. Just above, the "Gateway of Last Resort" refers to the chosen default route, which in this case is the just-configured static route with outgoing interface S0/0/1.

Troubleshooting Static Routes

These final few pages about IPv4 static routes examine some issues that can occur with static routes, both reviewing some reasons mentioned over the last few pages, while adding more detail. This topic breaks static route troubleshooting into three perspectives:

- The route is in the routing table but is incorrect.
- The route is not in the routing table.
- The route is in the routing table and is correct, but the packets do not arrive at the destination host.

Troubleshooting Incorrect Static Routes That Appear in the IP Routing Table

This first troubleshooting item can be obvious, but it is worth pausing to think about. A static route is only as good as the input typed into the **ip route** command. IOS checks the syntax, and as mentioned earlier, makes a few other checks that this section reviews in the next heading. But once those checks are passed, IOS puts the route into the IP routing table, even if the route had poorly chosen parameters.

For instance, the route might use a subnet and mask that implies a different range of addresses than the addresses in the destination subnet. Or, for a router sitting in the middle of a diagram, the next-hop address might be a router to the left, while the destination subnet is to the right. Or the next-hop address could be an IP address in a connected subnet, but it might be a typo and be an address of a PC or even a currently unused IP address.

When you see an exam question that has static routes, and you see them in the output of **show ip route**, remember to check on these items:

Key Topic

- Is there a subnetting math error in the subnet ID and mask?
- Is the next-hop IP address correct and referencing an IP address on a neighboring router?
- Does the next-hop IP address identify the correct router?
- Is the outgoing interface correct, and referencing an interface on the local router (that is, the same router where the static route is configured)?

The Static Route Does Not Appear in the IP Routing Table

After configuring an **ip route** command, IOS might or might not add the route to the IP routing table. IOS also considers the following before adding the route to its routing table:

Key Topic

- For **ip route** commands that list an outgoing interface, that interface must be in an up/up state.
- For **ip route** commands that list a next-hop IP address, the local router must have a route to reach that next-hop address.

For example, earlier in Example 16-3, R1's command **ip route 172.16.3.0 255.255.255.0 172.16.5.3** defines a static route. Before adding the route to the IP routing table, R1 looks for an existing IP route to reach 172.16.5.3. In that case, R1 will find a connected route for subnet 172.16.5.0/24 as long as its Ethernet WAN link is up. As a result, R1 adds the static route to subnet 172.16.3.0/24. Later, if R1's G0/1/0 were to fail, R1 would remove its connected route to 172.16.5.0/24 from the IP routing table—an action that would also then cause R1 to remove its static route to 172.16.3.0/24.

You can configure a static route so that IOS ignores these basic checks, always putting the IP route in the routing table. To do so, just use the **permanent** keyword on the **ip route** command. For example, by adding the **permanent** keyword to the end of the two commands as demonstrated in Example 16-7, R1 would now add these routes, regardless of whether the two WAN links were up.

Example 16-7 *Permanently Adding Static Routes to the IP Routing Table (Router R1)*

```
ip route 172.16.2.0 255.255.255.0 S0/0/0 permanent
ip route 172.16.3.0 255.255.255.0 172.16.5.3 permanent
```

Note that although the **permanent** keyword lets the router keep the route in the routing table without checking the outgoing interface or route to the next-hop address, it does not magically fix a broken route. For example, if the outgoing interface fails, the route will remain in the routing table, but the router cannot forward packets because the outgoing interface is down.

The Correct Static Route Appears but Works Poorly

This last section is a place to make two points—one mainstream and one point to review a bit of trivia.

First, on the mainstream point, the static route can be perfect, but the packets from one host to the next still might not arrive because of other problems. An incorrect static route is just one of many items to check when you're troubleshooting problems like "host A cannot connect to server B." The root cause may be the static route, or it may be something else. Chapter 18, "Troubleshooting IPv4 Routing," goes into some depth about troubleshooting these types of problems.

On the more specific point, be wary of any **ip route** command with the **permanent** keyword. IOS puts these routes in the routing table with no checks for accuracy. You should check whether the outgoing interface is down and/or whether the router has a route to reach the next-hop address.

IP Forwarding with the Longest Prefix Match

A router's IP routing process requires that the router compare the destination IP address of each packet with the existing contents of that router's IP routing table. Often, only one route matches a particular destination address. When only one route matches the packet's destination, the action is obvious: forward the packet based on the details listed in that route.

In some cases, a particular destination address matches more than one of the router's routes. For instance, one route might list subnet 10.1.0.0/16, another 10.1.1.0/25, and another 10.1.1.1/32. All would match packets sent to IP address 10.1.1.1. Many legitimate router features can cause these multiple routes to appear in a router's routing table, including

- Static routes
- Route autosummarization
- Manual route summarization

This fourth of four major sections of this chapter explains how a router makes its routing decisions when a packet matches multiple routes. When more than one route matches a packet's destination address, the router uses the "best" route, defined as follows:

When a particular destination IP address matches more than one route in a router's IPv4 routing table, the router uses the most specific route—in other words, the route with the longest prefix length mask.

Using show ip route to Find the Best Route

We humans have a couple of ways to figure out what choice a router makes for choosing the best route. One way uses the **show ip route** command, plus some subnetting math, to decide

the route the router will choose. To let you see how to use this option, Example 16-8 shows a series of overlapping routes, all created with OSPF, so the output lists only OSPF-learned routes.

Example 16-8 **show ip route** *Command with Overlapping Routes*

```
R1# show ip route ospf
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is 172.16.25.129 to network 0.0.0.0

     172.16.0.0/16 is variably subnetted, 9 subnets, 5 masks
O       172.16.1.1/32 [110/50] via 172.16.25.2, 00:00:04, GigabitEthernet0/0/0
O       172.16.1.0/24 [110/100] via 172.16.25.129, 00:00:09, GigabitEthernet0/1/0
O       172.16.0.0/22 [110/65] via 172.16.24.2, 00:00:04, GigabitEthernet0/2/0
O       172.16.0.0/16 [110/65] via 172.16.24.129, 00:00:09, GigabitEthernet0/3/0
O       0.0.0.0/0 [110/129] via 172.16.25.129, 00:00:09, GigabitEthernet0/0/0
```

To predict which of its routes a router will match, two pieces of information are required: the destination IP address of the packet and the contents of the router's routing table. The subnet ID and mask listed for a route define the range of addresses matched by that route. With a little subnetting math, a network engineer can find the range of addresses matched by each route. For instance, Table 16-2 lists the five subnets listed in Example 16-8 and the address ranges implied by each.

Table 16-2 Analysis of Address Ranges for the Subnets in Example 16-8

Subnet/Prefix	Address Range
172.16.1.1/32	172.16.1.1 (just this one address)
172.16.1.0/24	172.16.1.0 – 172.16.1.255
172.16.0.0/22	172.16.0.0 – 172.16.3.255
172.16.0.0/16	172.16.0.0 – 172.16.255.255
0.0.0.0/0	0.0.0.0 – 255.255.255.255 (all addresses)

NOTE The route listed as 0.0.0.0/0 is the default route.

As you can see from these ranges, several of the routes' address ranges overlap. When matching more than one route, the route with the longer prefix length is used. That is, a route with /16 is better than a route with /10; a route with a /25 prefix is better than a route with a /20 prefix; and so on.

For example, a packet sent to 172.16.1.1 actually matches all five routes listed in the routing table in Example 16-8. The various prefix lengths range from /0 to /32. The longest prefix (largest /P value, meaning the best and most specific route) is /32. So, a packet sent to 172.16.1.1 uses the route to 172.16.1.1/32, and not the other routes.

The following list gives some examples of destination IP addresses. For each address, the list describes the routes from Table 16-2 that the router would match, and which specific route the router would use.

172.16.1.1: Matches all five routes; the longest prefix is /32, the route to 172.16.1.1/32.

172.16.1.2: Matches the last four routes; the longest prefix is /24, the route to 172.16.1.0/24.

172.16.2.3: Matches the last three routes; the longest prefix is /22, the route to 172.16.0.0/22.

172.16.4.3: Matches the last two routes; the longest prefix is /16, the route to 172.16.0.0/16.

Using show ip route *address* to Find the Best Route

A second way to identify the route a router will use, one that does not require any subnetting math, is the **show ip route** *address* command. The last parameter on this command is the IP address of an assumed IP packet. The router replies by listing the route it would use to route a packet sent to that address.

For example, Example 16-9 lists the output of the **show ip route 172.16.4.3** command on the same router used in Example 16-8. The first line of (highlighted) output lists the matched route: the route to 172.16.0.0/16. The rest of the output lists the details of that particular route, like the outgoing interface of GigabitEthernet0/3/0 and the next-hop router of 172.16.24.129.

Example 16-9 **show ip route** *Command with Overlapping Routes*

```
R1# show ip route 172.16.4.3
Routing entry for 172.16.0.0/16
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 10.2.2.5 on GigabitEthernet0/3/0, 14:22:06 ago
  Routing Descriptor Blocks:
  * 172.16.24.129, from 172.16.24.129, 14:22:05 ago, via GigabitEthernet0/3/0
      Route metric is 65, traffic share count is 1
```

Certainly, if you have an option, just using a command to check what the router actually chooses is a much quicker option than doing the subnetting math.

Interpreting the IP Routing Table

The **show ip route** command plays a huge role in verifying and troubleshooting IP routing and addressing. This final topic of the chapter pulls the concepts together in one place for easier reference and study.

Figure 16-15 shows the output of a sample **show ip route** command. The figure numbers various parts of the command output for easier reference, with Table 16-3 describing the output noted by each number.

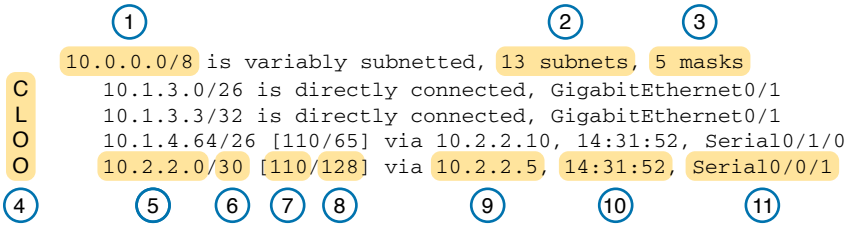


Figure 16-15 **show ip route** *Command Output Reference*

Key Topic

Table 16-3 Descriptions of the **show ip route** Command Output

Item	Idea	Value in the Figure	Description
1	Classful network	10.0.0.0/8	The routing table is organized by classful network. This line is the heading line for classful network 10.0.0.0; it lists the default mask for Class A networks (/8).
2	Number of subnets	13 subnets	The number of routes for subnets of the classful network known to this router, from all sources, including local routes—the /32 routes that match each router interface IP address.
3	Number of masks	5 masks	The number of different masks used in all routes known to this router inside this classful network.
4	Legend code	C, L, O	A short code that identifies the source of the routing information. *O* is for OSPF, *D* for EIGRP, *C* for Connected, *S* for static, and *L* for local. (See Example 16-8 for a sample of the legend.)
5	Prefix (Subnet ID)	10.2.2.0	The subnet number of this particular route.
6	Prefix length (Mask)	/30	The prefix mask used with this subnet.
7	Administrative distance	110	If a router learns routes for the listed subnet from more than one source of routing information, the router uses the source with the lowest administrative distance (AD).
8	Metric	128	The metric for this route.
9	Next-hop router	10.2.2.5	For packets matching this route, the IP address of the next router to which the packet should be forwarded.
10	Timer	14:31:52	For OSPF and EIGRP routes, this is the time since the route was first learned.
11	Outgoing interface	Serial0/0/1	For packets matching this route, the interface out which the packet should be forwarded.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 16-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 16-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Do labs		Blog

Review All the Key Topics

Key Topic

Table 16-5 Key Topics for Chapter 16

Key Topic Element	Description	Page Number
List	Steps taken by a host when forwarding IP packets	369
List	Steps taken by a router when forwarding IP packets	370
Figure 16-2	Diagram of five routing steps taken by a router	371
Figure 16-7	Breakdown of IP routing table with matching and forwarding details	374
List	Three common sources from which routers build IP routes	376
List	Rules regarding when a router creates a connected route	376
Figure 16-11	Static route configuration concept	379
List	Troubleshooting checklist for routes that do appear in the IP routing table	385
List	Troubleshooting checklist for static routes that do not appear in the IP routing table	385
Paragraph	A description of how a router makes a longest prefix decision to match the routing table	386
Table 16-3	List of items found in a Cisco router IP routing table	389

Key Terms You Should Know

ARP table, routing table, next-hop router, outgoing interface, connected route, static route, default route, host route, floating static route, network route, administrative distance

Command References

Tables 16-6 and 16-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 16-6 Chapter 16 Configuration Command Reference

Command	Description
ip address *ip-address mask*	Interface subcommand that assigns the interface's IP address
interface *type number.subint*	Global command to create a subinterface and to enter configuration mode for that subinterface
[**no**] **ip routing**	Global command that enables (**ip routing**) or disables (**no ip routing**) the routing of IPv4 packets on a router or Layer 3 switch
ip route *prefix mask* {*ip-address* | *interface-type interface-number*} [*distance*] [**permanent**]	Global configuration command that creates a static route

Table 16-7 Chapter 16 EXEC Command Reference

Command	Description
show ip route	Lists the router's entire routing table
show ip route [connected | static | ospf]	Lists a subset of the IP routing table
show ip route *ip-address*	Lists detailed information about the route that a router matches for the listed IP address
show arp, show ip arp	Lists the router's IPv4 ARP table
clear ip arp [ip-address]	Removes all dynamically learned ARP table entries, or if the command lists an IP address, removes the entry for that IP address only

CHAPTER 17

IP Routing in the LAN

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

2.0 Network Access

2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)

The preceding two chapters showed how to configure an IP address and mask on a router interface, making the router ready to route packets to/from the subnet implied by that address/mask combination. While true and useful, all the examples so far ignored the LAN switches and the possibility of VLANs. In fact, the examples so far show the simplest possible cases: the attached switches as Layer 2 switches, using only one VLAN, with the router configured with one **ip address** command on its physical interface. This chapter takes a detailed look at how to configure routers so that they route packets to/from the subnets that exist on each and every VLAN.

Because Layer 2 switches do not forward Layer 2 frames between VLANs, a network must use routers to route IP packets between subnets to allow those devices in different VLANs/subnets to communicate. To review, Ethernet defines the concept of a VLAN, while IP defines the concept of an IP subnet, so a VLAN is not equivalent to a subnet. However, the set of devices in one VLAN are typically also in one subnet. By the same reasoning, devices in two different VLANs are normally in two different subnets. For two devices in different VLANs to communicate with each other, routers must connect to the subnets that exist on each VLAN, and then the routers forward IP packets between the devices in those subnets.

This chapter discusses the configuration and verification steps related to three methods of routing between VLANs with three major sections:

- **VLAN Routing with Router 802.1Q Trunks:** The first section discusses how to configure a router to use VLAN trunking as connected to a Layer 2 switch. The router does the routing, with the switch creating the VLANs. The link between the router and switch use trunking so that the router has an interface connected to each VLAN/subnet. This feature is known as routing over a VLAN trunk and also known as router-on-a-stick (ROAS).
- **VLAN Routing with Layer 3 Switch SVIs:** The second section discusses using a LAN switch that supports both Layer 2 switching and Layer 3 routing (called a Layer 3 switch or multilayer switch). To route, the Layer 3 switch configuration uses interfaces called switched virtual interfaces (SVI), which are also called VLAN interfaces.
- **VLAN Routing with Layer 3 Switch Routed Ports:** The third major section of the chapter discusses an alternative to SVIs called routed ports, in which the physical switch ports are made to act like interfaces on a router. This third section also introduces the concept of an EtherChannel as used as a routed port in a feature called Layer 3 EtherChannel.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 17-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
VLAN Routing with Router 802.1Q Trunks	1, 2
VLAN Routing with Layer 3 Switch SVIs	3, 4
VLAN Routing with Layer 3 Switch Routed Ports	5, 6

1. Router 1 has a Fast Ethernet interface 0/0 with IP address 10.1.1.1. The interface is connected to a switch. This connection is then migrated to use 802.1Q trunking. Which of the following commands could be part of a valid configuration for Router 1's Fa0/0 interface? (Choose two answers.)
   a. **interface fastethernet 0/0.4**
   b. **dot1q enable**
   c. **dot1q enable 4**
   d. **trunking enable**
   e. **trunking enable 4**
   f. **encapsulation dot1q 4**
2. Router R1 has a router-on-a-stick (ROAS) configuration with two subinterfaces of interface G0/1: G0/1.1 and G0/1.2. Physical interface G0/1 is currently in a down/down state. The network engineer then configures a **shutdown** command when in interface configuration mode for G0/1.1 and a **no shutdown** command when in interface configuration mode for G0/1.2. Which answers are correct about the interface state for the subinterfaces? (Choose two answers.)
   a. G0/1.1 will be in a down/down state.
   b. G0/1.2 will be in a down/down state.
   c. G0/1.1 will be in an administratively down state.
   d. G0/1.2 will be in an up/up state.

3. A Layer 3 switch has been configured to route IP packets between VLANs 1, 2, and 3 using SVIs, which connect to subnets 172.20.1.0/25, 172.20.2.0/25, and 172.20.3.0/25, respectively. The engineer issues a **show ip route connected** command on the Layer 3 switch, listing the connected routes. Which of the following answers lists a piece of information that should be in at least one of the routes?
   a. Interface Gigabit Ethernet 0/0.3
   b. Next-hop router 172.20.2.1
   c. Interface VLAN 2
   d. Mask 255.255.255.0

4. An engineer has successfully configured a Layer 3 switch with SVIs for VLANs 2 and 3. Hosts in the subnets using VLANs 2 and 3 can ping each other with the Layer 3 switch routing the packets. The next week, the network engineer receives a call that those same users can no longer ping each other. If the problem is with the Layer 3 switching function, which of the following could have caused the problem? (Choose two answers.)
   a. Six (or more) out of 10 working VLAN 2 access ports failing due to physical problems
   b. A **shutdown** command issued from interface VLAN 4 configuration mode
   c. VTP on the switch removing VLAN 3 from the switch's VLAN list
   d. A **shutdown** command issued from VLAN 2 configuration mode

5. A LAN design uses a Layer 3 EtherChannel between two switches SW1 and SW2, with port-channel interface 1 used on both switches. SW1 uses ports G0/1, G0/2, and G0/3 in the channel. Which of the following are true about SW1's configuration to make the channel be able to route IPv4 packets correctly? (Choose two answers.)
   a. The **ip address** command must be on the port-channel 1 interface.
   b. The **ip address** command must be on interface G0/1 (lowest numbered port).
   c. The port-channel 1 interface must be configured with the **no switchport** command.
   d. Interface G0/1 must be configured with the **routedport** command.

6. A LAN design uses a Layer 3 EtherChannel between two switches SW1 and SW2, with port-channel interface 1 used on both switches. SW1 uses ports G0/1 and G0/2 in the channel. However, only interface G0/1 is bundled into the channel and working. Think about the configuration settings on port G0/2 that could have existed before adding G0/2 to the EtherChannel. Which answers identify a setting that could prevent IOS from adding G0/2 to the Layer 3 EtherChannel? (Choose two answers.)
   a. A different STP cost (**spanning-tree cost** *value*)
   b. A different speed (**speed** *value*)
   c. A default setting for switchport (**switchport**)
   d. A different access VLAN (**switchport access vlan** *vlan-id*)

Foundation Topics

VLAN Routing with Router 802.1Q Trunks

Almost all enterprise networks use VLANs. To route IP packets in and out of those VLANs, some devices (either routers or Layer 3 switches) need to have an IP address in each subnet and have a connected route to each of those subnets. Then the IP addresses on those routers or Layer 3 switches can serve as the default gateways in those subnets.

This chapter breaks down the LAN routing options into four categories:

- Use a router, with one router LAN interface and cable connected to the switch for each and every VLAN (typically not used)
- Use a router, with a VLAN trunk connecting to a LAN switch (known as router-on-a-stick, or ROAS)
- Use a Layer 3 switch with switched virtual interfaces (SVI)
- Use a Layer 3 switch with routed interfaces (which may or may not be Layer 3 EtherChannels)

Of the items in the list, the first option works, but to be practical, it requires far too many interfaces. It is mentioned here only to make the list complete.

As for the other three options, this chapter discusses each in turn as the main focus of one of the three major sections in this chapter. Each feature is used in real networks today, with the choice to use one or the other driven by the design and needs for a particular part of the network. Figure 17-1 shows cases in which these options could be used.

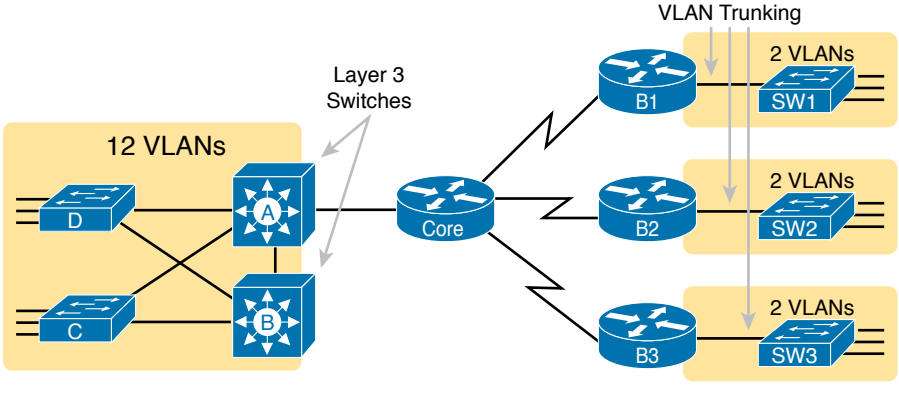


Figure 17-1 *Layer 3 Switching at the Central Site*

Figure 17-1 shows two switches, labeled A and B, which could act as Layer 3 switches—both with SVIs and routed interfaces. The figure shows a central site campus LAN on the left, with 12 VLANs. Switches A and B act as Layer 3 switches, combining the functions of a router and a switch, routing between all 12 subnets/VLANs, as well as routing to/from the Core router. Those Layer 3 switches could use SVIs, routed interfaces, or both.

Figure 17-1 also shows a classic case for using a router with a VLAN trunk. Sites like the remote sites on the right side of the figure may have a WAN-connected router and a LAN

switch. These sites might use ROAS to take advantage of the router's ability to route over an 802.1Q trunk.

Note that Figure 17-1 just shows an example. The engineer could use Layer 3 switching at each site or routers with VLAN trunking at each site.

Configuring ROAS

This next topic discusses how routers route packets to subnets associated with VLANs connected to a router 802.1Q trunk. That long description can be a bit of a chore to repeat each time someone wants to discuss this feature, so over time, the networking world has instead settled on a shorter and more interesting name for this feature: router-on-a-stick (ROAS).

ROAS uses router VLAN trunking configuration to give the router a logical router interface connected to each VLAN. Because the router then has an interface connected to each VLAN, the router can also be configured with an IP address in the subnet that exists on each VLAN.

Routers use subinterfaces as the means to have an interface connected to a VLAN. The router needs to have an IP address/mask associated with each VLAN on the trunk. However, the router has only one physical interface for the link connected to the trunk. Cisco solves this problem by creating multiple virtual router interfaces, one associated with each VLAN on that trunk (at least for each VLAN that you want the trunk to support). Cisco calls these virtual interfaces *subinterfaces*. The configuration can then include an **ip address** command for each subinterface.

Figure 17-2 shows the concept with Router B1, one of the branch routers from Figure 17-1. Because this router needs to route between only two VLANs, the figure also shows two subinterfaces, named G0/0.10 and G0/0.20, which create a new place in the configuration where the per-VLAN configuration settings can be made. The router treats frames tagged with VLAN 10 as if they came in or out of G0/0.10 and frames tagged with VLAN 20 as if they came in or out G0/0.20.

Key Topic

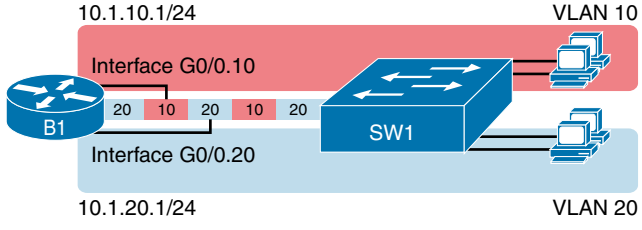


Figure 17-2 *Subinterfaces on Router B1*

In addition, note that most Cisco routers do not attempt to negotiate trunking, so both the router and switch need to manually configure trunking. This chapter discusses the router side of that trunking configuration; the matching switch interface would need to be configured with the **switchport mode trunk** command.

Answers to the "Do I Know This Already?" quiz:

1 A, F **2** B, C **3** C **4** C, D **5** A, C **6** B, C

Example 17-1 shows a full example of the 802.1Q trunking configuration required on Router B1 in Figure 17-2. More generally, these steps detail how to configure 802.1Q trunking on a router:

Config Checklist

Step 1. Use the **interface** *type number.subint* command in global configuration mode to create a unique subinterface for each VLAN that needs to be routed.

Step 2. Use the **encapsulation dot1q** *vlan_id* command in subinterface configuration mode to enable 802.1Q and associate one specific VLAN with the subinterface.

Step 3. Use the **ip address** *address mask* command in subinterface configuration mode to configure IP settings (address and mask).

Example 17-1 *Router Configuration for the 802.1Q Encapsulation Shown in Figure 17-2*

```
B1# show running-config
! Only pertinent lines shown
interface gigabitethernet 0/0
! No IP address up here! No encapsulation up here!
!
interface gigabitethernet 0/0.10
 encapsulation dot1q 10
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

First, look at the subinterface numbers. The subinterface number begins with the period, like .10 and .20 in this case. These numbers can be any number from 1 up through a very large number (over 4 billion). The number just needs to be unique among all subinterfaces associated with this one physical interface. In fact, the subinterface number does not even have to match the associated VLAN ID. (The **encapsulation** command, and not the subinterface number, defines the VLAN ID associated with the subinterface.)

NOTE Although not required, most sites do choose to make the subinterface number match the VLAN ID, as shown in Example 17-1, just to avoid confusion.

Each subinterface configuration lists two subcommands. One command (**encapsulation**) enables trunking and defines the VLAN whose frames are considered to be coming in and out of the subinterface. The **ip address** command works the same way it does on any other interface. Note that if the physical Ethernet interface reaches an up/up state, the subinterface should as well, which would then let the router add the connected routes shown at the bottom of the example.

Now that the router has a working interface, with IPv4 addresses configured, the router can route IPv4 packets on these subinterfaces. That is, the router treats these subinterfaces like

any physical interface in terms of adding connected routes, matching those routes, and forwarding packets to/from those connected subnets.

The configuration and use of the native VLAN on the trunk require a little extra thought. The native VLAN can be configured on a subinterface, or on the physical interface, or ignored as in Example 17-1. Each 802.1Q trunk has one native VLAN, and if the router needs to route packets for a subnet that exists in the native VLAN, then the router needs some configuration to support that subnet. The two options to define a router interface for the native VLAN are

- Configure the **ip address** command on the physical interface, but without an **encapsulation** command; the router considers this physical interface to be using the native VLAN.
- Configure the **ip address** command on a subinterface and use the **encapsulation dot1q** *vlan-id* **native** subcommand to tell the router both the VLAN ID and the fact that it is the native VLAN.

Example 17-2 shows both native VLAN configuration options with a small change to the same configuration in Example 17-1. In this case, VLAN 10 becomes the native VLAN. The top part of the example shows the option to configure the router physical interface to use native VLAN 10. The second half of the example shows how to configure that same native VLAN on a subinterface. In both cases, the switch configuration also needs to be changed to make VLAN 10 the native VLAN.

Example 17-2 *Router Configuration Using Native VLAN 10 on Router B1*

```
! First option: put the native VLAN IP address on the physical interface
interface gigabitethernet 0/0
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

```
! Second option: like Example 17-1, but add the native keyword
interface gigabitethernet 0/0.10
 encapsulation dot1q 10 native
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

Verifying ROAS

Beyond using the **show running-config** command, ROAS configuration on a router can be best verified with two commands: **show ip route** [**connected**] and **show vlans**. As with any router interface, as long as the interface is in an up/up state and has an IPv4 address configured, IOS will put a connected (and local) route in the IPv4 routing table. So, a first and obvious check would be to see if all the expected connected routes exist. Example 17-3 lists the connected routes per the configuration shown in Example 17-1.

Example 17-3 *Connected Routes Based on Example 17-1 Configuration*

```
B1# show ip route connected
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! Legend omitted for brevity

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.10.0/24 is directly connected, GigabitEthernet0/0.10
L        10.1.10.1/32 is directly connected, GigabitEthernet0/0.10
C        10.1.20.0/24 is directly connected, GigabitEthernet0/0.20
L        10.1.20.1/32 is directly connected, GigabitEthernet0/0.20
```


As for interface and subinterface state, note that the ROAS subinterface state does depend to some degree on the physical interface state. In particular, the subinterface state cannot be better than the state of the matching physical interface. For instance, on Router B1 in the examples so far, physical interface G0/0 is in an up/up state, and the subinterfaces are in an up/up state. But if you unplugged the cable from that port, the physical port would fail to a down/down state, and the subinterfaces would also fail to a down/down state. Example 17-4 shows another example, with the physical interface being shut down, with the subinterfaces then automatically changed to an administratively down state as a result.

Example 17-4 *Subinterface State Tied to Physical Interface State*

```
B1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
B1(config)# interface g0/0
B1(config-if)# shutdown
B1(config-if)# ^Z
B1# show ip interface brief | include 0/0
GigabitEthernet0/0         unassigned      YES manual administratively down down
GigabitEthernet0/0.10      10.1.10.1       YES manual administratively down down
GigabitEthernet0/0.20      10.1.20.1       YES manual administratively down down
```

Additionally, the subinterface state can also be enabled and disabled independently from the physical interface, using the **no shutdown** and **shutdown** commands in subinterface configuration mode.

Another useful ROAS verification command, **show vlans**, spells out which router trunk interfaces use which VLANs, which VLAN is the native VLAN, plus some packet statistics. The fact that the packet counters are increasing can be useful when verifying whether traffic is happening or not. Example 17-5 shows a sample, based on the Router B1 configuration in Example 17-2 (bottom half), in which native VLAN 10 is configured on subinterface G0/0.10. Note that the output identifies VLAN 1 associated with the physical interface, VLAN 10 as the native VLAN associated with G0/0.10, and VLAN 20 associated with G0/0.20. It also lists the IP addresses assigned to each interface/subinterface.

Example 17-5 *Sample* **show vlans** *Command to Match Sample Router Trunking Configuration*

```
R1# show vlans
Virtual LAN ID: 1 (IEEE 802.1Q Encapsulation)

   vLAN Trunk Interface: GigabitEthernet0/0

   Protocols Configured:  Address:        Received:    Transmitted:
        Other                                    0              83

   69 packets, 20914 bytes input
   147 packets, 11841 bytes output

Virtual LAN ID:   10 (IEEE 802.1Q Encapsulation)

    vLAN Trunk Interface:   GigabitEthernet0/0.10

 This is configured as native Vlan for the following interface(s) :
GigabitEthernet0/0      Native-vlan Tx-type: Untagged

      Protocols Configured:   Address:     Received:    Transmitted:
              IP              10.1.10.1           2              3
           Other                                  0              1

   3 packets, 722 bytes input
   4 packets, 264 bytes output

Virtual LAN ID:   20 (IEEE 802.1Q Encapsulation)

    vLAN Trunk Interface:   GigabitEthernet0/0.20

    Protocols Configured:  Address:        Received:  Transmitted:
            IP             10.1.20.1              0            134
         Other                                    0              1

    0 packets, 0 bytes input
    135 packets, 10498 bytes output
```

Troubleshooting ROAS

The biggest challenge when troubleshooting ROAS has to do with the fact that if you misconfigure only the router or misconfigure only the switch, the other device on the trunk has no way to know that the other side is misconfigured. That is, if you check the **show ip route** and **show vlans** commands on a router, and the output looks like it matches the intended configuration, and the connected routes for the correct subinterfaces show up, routing may still fail because of problems on the attached switch. So, troubleshooting ROAS often begins with checking the configuration on both the router and switch because there is no status output on either device that tells you where the problem might be.

First, to check ROAS on the router, you need to start with the intended configuration and ask questions about the configuration:

Key Topic

1. Is each non-native VLAN configured on the router with an **encapsulation dot1q** *vlan-id* command on a subinterface?
2. Do those same VLANs exist on the trunk on the neighboring switch (**show interfaces trunk**), and are they in the allowed list, not VTP pruned, and not STP blocked?
3. Does each router ROAS subinterface have an IP address/mask configured per the planned configuration?
4. If using the native VLAN, is it configured correctly on the router either on a subinterface (with an **encapsulation dot1q** *vlan-id* **native** command) or implied on the physical interface?
5. Is the same native VLAN configured on the neighboring switch's trunk in comparison to the native VLAN configured on the router?
6. Are the router physical or ROAS subinterfaces configured with a **shutdown** command?

For some of these steps, you need to be ready to investigate possible VLAN trunking issues on the LAN switch. The reason is that on many Cisco routers, router interfaces do not negotiate trunking. As a result, ROAS relies on static trunk configuration on both the router and switch. If the switch has any problems with VLANs or the VLAN trunking configuration on its side of the trunk, the router has no way to realize that the problem exists.

For example, imagine you configured ROAS on a router just like in Example 17-1 or Example 17-2. However, the switch on the other end of the link had no matching configuration. For instance, maybe the switch did not even define VLANs 10 and 20. Maybe the switch did not configure trunking on the port connected to the router. Even with blatant misconfiguration or missing configuration on the switch, the router still shows up/up ROAS interfaces and subinterfaces, IP routes in the output of **show ip route**, and meaningful configuration information in the output of the **show vlans** command.

VLAN Routing with Layer 3 Switch SVIs

Using a router with ROAS to route packets makes sense in some cases, particularly at small remote sites. In sites with a larger LAN, network designers choose to use Layer 3 switches for most inter-VLAN routing.

A Layer 3 switch (also called a multilayer switch) is one device, but it executes logic at two layers: Layer 2 LAN switching and Layer 3 IP routing. The Layer 2 switch function forwards frames inside each VLAN, but it will not forward frames between VLANs. The Layer 3 forwarding (routing) logic forwards IP packets between VLANs.

Layer 3 switches typically support two configuration options to enable IPv4 routing inside the switch, specifically to enable IPv4 on switch interfaces. This section explains one option, an option that uses switched virtual interfaces (SVI). The final major section of the chapter deals with the other option for configuring IPv4 addresses on Layer 3 switches: routed interfaces.

Configuring Routing Using Switch SVIs

The configuration of a Layer 3 switch mostly looks like the Layer 2 switching configuration shown back in Parts II and III of this book, with a small bit of configuration added for

the Layer 3 functions. The Layer 3 switching function needs a virtual interface connected to each VLAN internal to the switch. These *VLAN interfaces* act like router interfaces, with an IP address and mask. The Layer 3 switch has an IP routing table, with connected routes off each of these VLAN interfaces. (These interfaces are also referred to as *switched virtual interfaces* [SVI].)

To show the concept of Layer 3 switching with SVIs, the following example uses the same branch office with two VLANs shown in the earlier examples, but now the design will use Layer 3 switching in the LAN switch. Figure 17-3 shows the design changes and configuration concept for the Layer 3 switch function with a router icon inside the switch, to emphasize that the switch routes the packets.

Key Topic

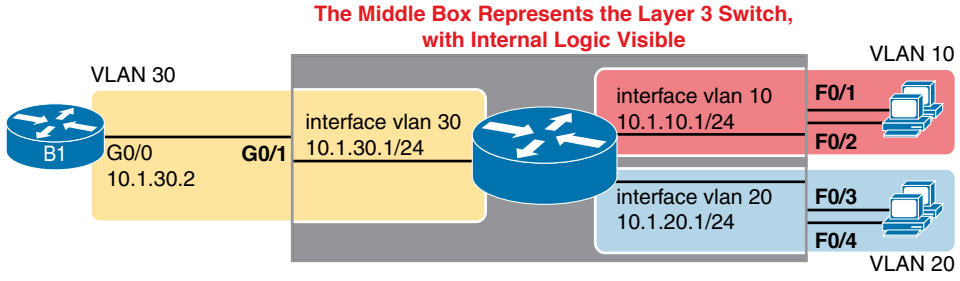


Figure 17-3 *Routing on VLAN Interfaces in a Layer 3 Switch*

Note that the figure represents the internals of the Layer 3 switch within the box in the middle of the figure. The branch still has two user VLANs (10 and 20), so the Layer 3 switch needs one VLAN interface for each VLAN. The figure shows a router icon inside the gray box to represent the Layer 3 switching function, with two VLAN interfaces on the right side of that icon. In addition, the traffic still needs to get to router B1 (a physical router) to access the WAN, so the switch uses a third VLAN (VLAN 30 in this case) for the link to Router B1. The physical link between the Layer 3 switch and router B1 would not be a trunk, but instead be an access link.

The following steps show how to configure Layer 3 switching using SVIs. Note that on some switches, like the 2960 and 2960-XR switches used for the examples in this book, the ability to route IPv4 packets must be enabled first, with a **reload** of the switch required to enable the feature. The steps that occur after the reload would apply to all models of Cisco switches that are capable of doing Layer 3 switching.

Config Checklist

Step 1. Enable IP routing on the switch, as needed:

A. Use the **sdm prefer lanbase-routing** command (or similar) in global configuration mode to change the switch forwarding ASIC settings to make space for IPv4 routes at the next reload of the switch.

B. Use the **reload** EXEC command in enable mode to reload (reboot) the switch to pick up the new **sdm prefer** command setting.

C. Once reloaded, use the **ip routing** command in global configuration mode to enable the IPv4 routing function in IOS software and to enable key commands like **show ip route**.

Step 2. Configure each SVI interface, one per VLAN for which routing should be done by this Layer 3 switch:

A. Use the **interface vlan** *vlan_id* command in global configuration mode to create a VLAN interface and to give the switch's routing logic a Layer 3 interface connected into the VLAN of the same number.

B. Use the **ip address** *address mask* command in VLAN interface configuration mode to configure an IP address and mask on the VLAN interface, enabling IPv4 routing on that VLAN interface.

C. (As needed) Use the **no shutdown** command in interface configuration mode to enable the VLAN interface (if it is currently in a shutdown state).

Example 17-6 shows the configuration to match Figure 17-3. In this case, switch SW1 has already used the **sdm prefer** global command to change to a setting that supports IPv4 routing, and the switch has been reloaded. The example shows the related configuration on all three VLAN interfaces.

Example 17-6 *VLAN Interface Configuration for Layer 3 Switching*

```
ip routing
!
interface vlan 10
 ip address 10.1.10.1 255.255.255.0
!
interface vlan 20
 ip address 10.1.20.1 255.255.255.0
!
interface vlan 30
 ip address 10.1.30.1 255.255.255.0
```

Verifying Routing with SVIs

With the VLAN configuration shown in the previous section, the switch is ready to route packets between the VLANs as shown in Figure 17-3. To support the routing of packets, the switch adds connected IP routes as shown in Example 17-7; note that each route is listed as being connected to a different VLAN interface.

Example 17-7 *Connected Routes on a Layer 3 Switch*

```
SW1# show ip route
! legend omitted for brevity

       10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C         10.1.10.0/24 is directly connected, Vlan10
L         10.1.10.1/32 is directly connected, Vlan10
C         10.1.20.0/24 is directly connected, Vlan20
L         10.1.20.1/32 is directly connected, Vlan20
C         10.1.30.0/24 is directly connected, Vlan30
L         10.1.30.1/32 is directly connected, Vlan30
```

The switch would also need additional routes to the rest of the network (not shown in the figures in this chapter). The Layer 3 switch could use static routes or a routing protocol, depending on the capabilities of the switch. For instance, if you then enabled OSPF on the Layer 3 switch, the configuration and verification would work the same as it does on a router, as discussed in Chapter 20, "Implementing OSPF." The routes that IOS adds to the Layer 3 switch's IP routing table would list the VLAN interfaces as outgoing interfaces.

NOTE Some models of Cisco enterprise switches, based on model, IOS version, and IOS feature set, support different capabilities for IP routing and routing protocols, so for real networks, check the capabilities of the switch model by browsing at Cisco.com. In particular, check the Cisco Feature Navigator (CFN) tool at http://www.cisco.com/go/cfn.

Troubleshooting Routing with SVIs

There are two big topics to investigate when troubleshooting routing over LANs with SVIs. First, you have to make sure the switch has been enabled to support IP routing. Second, the VLAN associated with each VLAN interface must be known and active on the local switch; otherwise, the VLAN interfaces do not come up.

First, about enabling IP routing, note that some models of Cisco switches default to enable Layer 3 switching, and some do not. So, to make sure your switch supports Layer 3 routing, look to those first few configuration commands listed in the configuration checklist found in the earlier section "Configuring Routing Using Switch SVIs." Those commands are **sdm prefer** (followed by a **reload**) and then **ip routing** (after the **reload**).

The **sdm prefer** command changes how the switch forwarding chips allocate memory for different forwarding tables, and changes to those tables require a reload of the switch. By default, many access switches that support Layer 3 switching still have an SDM default that does not allocate space for an IP routing table. Once changed and reloaded, the **ip routing** command then enables IPv4 routing in IOS software. Both are necessary before some Cisco switches will act as a Layer 3 switch.

Example 17-8 shows some symptoms on a router for which Layer 3 switching had not yet been enabled by the **sdm prefer** command. As you can see, both the **show ip route** EXEC command and the **ip routing** config command are rejected because they do not exist to IOS until the **sdm prefer** command has been used (followed by a **reload** of the switch).

Example 17-8 *Evidence That a Switch Has Not Yet Enabled IPv4 Routing*

```
SW1# show ip route
             ^
% Invalid input detected at '^' marker.

SW3# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW3(config)# ip routing
               ^
% Invalid input detected at '^' marker.
```

The second big area to investigate when troubleshooting SVIs relates to the SVI state, a state that ties to the state of the associated VLANs. Each VLAN interface has a matching VLAN of the same number, and the VLAN interface's state is tied to the state of the VLAN in certain ways. In particular, for a VLAN interface to be in an up/up state:

Key Topic

Step 1. The VLAN must be defined on the local switch (either explicitly or learned with VTP).

Step 2. The switch must have at least one up/up interface using the VLAN, either/both:

A. An up/up access interface assigned to that VLAN

B. A trunk interface for which the VLAN is in the allowed list, is STP forwarding, and is not VTP pruned

Step 3. The VLAN (not the VLAN interface) must be administratively enabled (that is, not **shutdown**).

Step 4. The VLAN interface (not the VLAN) must be administratively enabled (that is, not **shutdown**).

When working through the steps in the list, keep in mind that the VLAN and the VLAN interface are related but separate ideas, and the configuration items are separate in the CLI. The VLAN interface is a switch's Layer 3 interface connected to the VLAN. If you want to route packets for the subnets on VLANs 11, 12, and 13, the matching VLAN interfaces must be numbered 11, 12, and 13. And both the VLANs and the VLAN interfaces can be disabled and enabled with the **shutdown** and **no shutdown** commands (as mentioned in Steps 3 and 4 in the previous list), so you have to check for both.

Example 17-9 shows three scenarios, each of which leads to one of the VLAN interfaces in the previous configuration example (Figure 17-3, Example 17-6) to fail. At the beginning of the example, all three VLAN interfaces are up/up. VLANs 10, 20, and 30 each have at least one access interface up and working. The example works through three scenarios:

- **Scenario 1:** The last access interface in VLAN 10 is shut down (F0/1), so IOS shuts down the VLAN 10 interface.
- **Scenario 2:** VLAN 20 (not VLAN interface 20, but VLAN 20) is deleted, which results in IOS then bringing down (not shutting down) the VLAN 20 interface.
- **Scenario 3:** VLAN 30 (not VLAN interface 30, but VLAN 30) is shut down, which results in IOS then bringing down (not shutting down) the VLAN 30 interface.

Example 17-9 *Three Examples That Cause VLAN Interfaces to Fail*

```
SW1# show interfaces status
! Only ports related to the example are shown
Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        connected    10         a-full  a-100 10/100BaseTX
Fa0/2                        notconnect   10           auto   auto 10/100BaseTX
Fa0/3                        connected    20         a-full  a-100 10/100BaseTX
Fa0/4                        connected    20         a-full  a-100 10/100BaseTX
Gi0/1                        connected    30         a-full a-1000 10/100/1000BaseTX
```

```
SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.

! Case 1: Interface F0/1, the last up/up access interface in VLAN 10, is shutdown
SW1(config)# interface fastEthernet 0/1
SW1(config-if)# shutdown
SW1(config-if)#
*Apr 2 19:54:08.784: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan10, changed
state to down
SW1(config-if)#
*Apr 2 19:54:10.772: %LINK-5-CHANGED: Interface FastEthernet0/1, changed state to
administratively down
*Apr 2 19:54:11.779: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to down

! Case 2: VLAN 20 is deleted
SW1(config)# no vlan 20
SW1(config)#
*Apr 2 19:54:39.688: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan20, changed
state to down

! Case 3: VLAN 30, the VLAN from the switch to the router, is shutdown
SW1(config)# vlan 30
SW1(config-vlan)# shutdown
SW1(config-vlan)# exit
SW1(config)#
*Apr 2 19:55:25.204: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan30, changed
state to down

! Final status of all three VLAN interfaces are below
SW1# show ip interface brief | include Vlan
Vlan1                  unassigned      YES manual administratively down down
Vlan10                 10.1.10.1       YES manual up                    down
Vlan20                 10.1.20.1       YES manual up                    down
Vlan30                 10.1.30.1       YES manual up                    down
```

Note that the example ends with the three VLAN interfaces in an up/down state per the **show ip interface brief** command.

VLAN Routing with Layer 3 Switch Routed Ports

When Layer 3 switches use SVIs, the physical interfaces on the switches act like they always have: as Layer 2 interfaces. That is, the physical interfaces receive Ethernet frames. The switch learns the source MAC address of the frame, and the switch forwards the frame based on the destination MAC address. To perform routing, any Ethernet frames destined for any of the SVI interface MAC addresses trigger the processing of the Layer 2 switching logic, resulting in normal routing actions like stripping data-link headers, making a routing decision, and so on.

Alternately, the Layer 3 switch configuration can make a physical port act like a router interface instead of a switch interface. To do so, the switch configuration makes that port a routed port. On a *routed* port, the switch does not perform Layer 2 switching logic on that frame. Instead, frames arriving in a routed port trigger the Layer 3 routing logic, including

1. Stripping off the incoming frame's Ethernet data-link header/trailer
2. Making a Layer 3 forwarding decision by comparing the destination IP address to the IP routing table
3. Adding a new Ethernet data-link header/trailer to the packet
4. Forwarding the packet, encapsulated in a new frame

This third major section of the chapter examines routed interfaces as configured on Cisco Layer 3 switches, but with a particular goal in mind: to also discuss Layer 3 EtherChannels. The exam topics do not mention routed interfaces specifically, but the exam topics do mention L3 EtherChannels, meaning Layer 3 EtherChannels.

You might recall that Chapter 10, "RSTP and EtherChannel Configuration," discussed Layer 2 EtherChannels. Like Layer 2 EtherChannels, Layer 3 EtherChannels also treat multiple links as one link. Unlike Layer 2 EtherChannels, however, Layer 3 EtherChannels treat the channel as a *routed* port instead of *switched* port. So this section first looks at routed ports on Cisco Layer 3 switches and then discusses Layer 3 EtherChannels.

Implementing Routed Interfaces on Switches

When a Layer 3 switch needs a Layer 3 interface connected to a subnet, and only one physical interface connects to that subnet, the network engineer can choose to use a routed port instead of an SVI. Conversely, when the Layer 3 switch needs a Layer 3 interface connected to a subnet, and many physical interfaces on the switch connect to that subnet, an SVI needs to be used. (SVIs forward traffic internally into the VLAN, so that then the Layer 2 logic can forward the frame out any of the ports in the VLAN. Routed ports cannot.)

To see why, consider the design in Figure 17-4, which repeats the same design from Figure 17-3 (used in the SVI examples). In that design, the gray rectangle on the right represents the switch and its internals. On the right of the switch, at least two access ports sit in both VLAN 10 and VLAN 20. However, that figure shows a single link from the switch to Router B1. The switch could configure the port as an access port in a separate VLAN, as shown with VLAN 30 in Examples 17-6 and 17-7. However, with only one switch port needed, the switch could configure that link as a routed port, as shown in the figure.

Key Topic

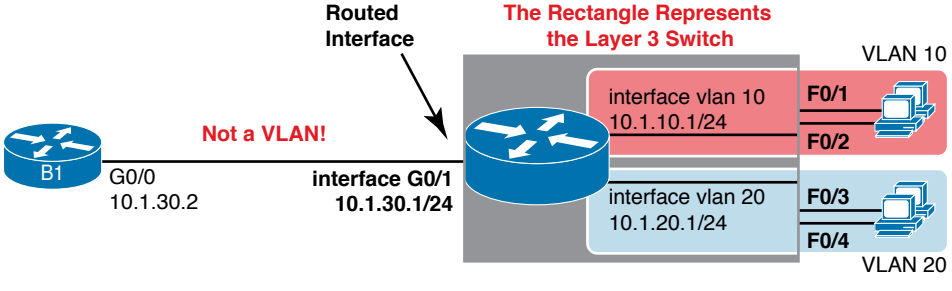


Figure 17-4 *Routing on a Routed Interface on a Switch*

Enabling a switch interface to be a routed interface instead of a switched interface is simple: just use the **no switchport** subcommand on the physical interface. Cisco switches capable of being a Layer 3 switch use a default of the **switchport** command to each switch physical interface. Think about the word *switchport* for a moment. With that term, Cisco tells the switch to treat the port like it is a port on a switch—that is, a Layer 2 port on a switch. To make the port stop acting like a switch port and instead act like a router port, use the **no switchport** command on the interface.

Once the port is acting as a routed port, think of it like a router interface. That is, configure the IP address on the physical port, as implied in Figure 17-4. Example 17-10 shows a completed configuration for the interfaces configured on the switch in Figure 17-4. Note that the design uses the exact same IP subnets as the example that showed SVI configuration in Example 17-6, but now, the port connected to subnet 10.1.30.0 has been converted to a routed port. All you have to do is add the **no switchport** command to the physical interface and configure the IP address on the physical interface.

Example 17-10 *Configuring Interface G0/1 on Switch SW1 as a Routed Port*

```
ip routing
!
interface vlan 10
 ip address 10.1.10.1 255.255.255.0
!
interface vlan 20
 ip address 10.1.20.1 255.255.255.0
!
interface gigabitethernet 0/1
 no switchport
 ip address 10.1.30.1 255.255.255.0
```

Once configured, the routed interface will show up differently in command output in the switch. In particular, for an interface configured as a routed port with an IP address, like interface GigabitEthernet0/1 in the previous example:

Key Topic

show interfaces: Similar to the same command on a router, the output will display the IP address of the interface. (Conversely, for switch ports, this command does not list an IP address.)

show interfaces status: Under the "VLAN" heading, instead of listing the access VLAN or the word *trunk*, the output lists the word *routed*, meaning that it is a routed port.

show ip route: Lists the routed port as an outgoing interface in routes.

show interfaces *type number* **switchport:** If a routed port, the output is short and confirms that the port is not a switch port. (If the port is a Layer 2 port, this command lists many configuration and status details.)

Example 17-11 shows samples of all four of these commands as taken from the switch as configured in Example 17-10.

Example 17-11 *Verification Commands for Routed Ports on Switches*

```
SW1# show interfaces g0/1
GigabitEthernet0/1 is up, line protocol is up (connected)
 Hardware is Gigabit Ethernet, address is bcc4.938b.e541 (bia bcc4.938b.e541)
 Internet address is 10.1.30.1/24
! lines omitted for brevity

SW1# show interfaces status
! Only ports related to the example are shown; the command lists physical only
Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        connected    10         a-full  a-100 10/100BaseTX
Fa0/2                        notconnect   10           auto   auto 10/100BaseTX
Fa0/3                        connected    20         a-full  a-100 10/100BaseTX
Fa0/4                        connected    20         a-full  a-100 10/100BaseTX
Gi0/1                        connected    routed     a-full a-1000 10/100/1000BaseTX

SW1# show ip route
! legend omitted for brevity

       10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C         10.1.10.0/24 is directly connected, Vlan10
L         10.1.10.1/32 is directly connected, Vlan10
C         10.1.20.0/24 is directly connected, Vlan20
L         10.1.20.1/32 is directly connected, Vlan20
C         10.1.30.0/24 is directly connected, GigabitEthernet0/1
L         10.1.30.1/32 is directly connected, GigabitEthernet0/1

SW1# show interfaces g0/1 switchport
Name: Gi0/1
Switchport: Disabled
```

So, with two options—SVI and routed ports—where should you use each?

> For any topologies with a point-to-point link between two devices that do routing, a routed interface works well.

Figure 17-5 shows an example of where to use SVIs and where to use routed ports in a typical core/distribution/access design. In this design, the core (Core1, Core2) and distribution (D11 through D14) switches perform Layer 3 switching. All the ports that are links directly between the Layer 3 switches can be routed interfaces. For VLANs for which many interfaces (access and trunk) connect to the VLAN, SVIs make sense because the SVIs can send and receive traffic out multiple ports on the same switch. In this design, all the ports on Core1 and Core2 will be routed ports, while the four distribution switches will use some routed ports and some SVIs.

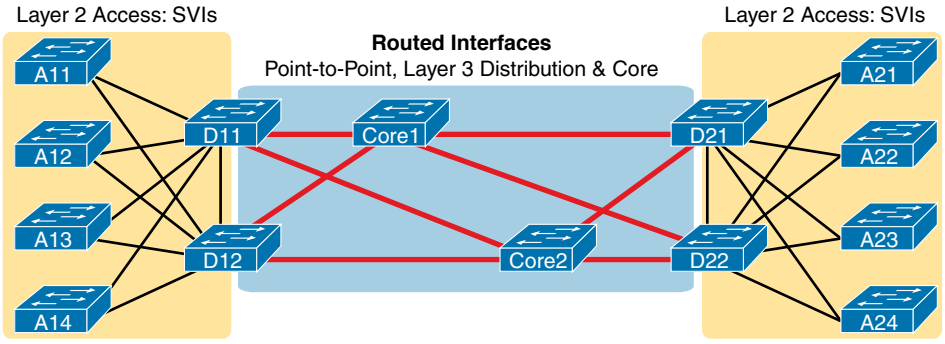


Figure 17-5 *Using Routed Interfaces for Core and Distribution Layer 3 Links*

Implementing Layer 3 EtherChannels

So far, this section has stated that routed interfaces can be used with a single point-to-point link between pairs of Layer 3 switches, or between a Layer 3 switch and a router. However, in most designs, the network engineers use at least two links between each pair of distribution and core switches, as shown in Figure 17-6.

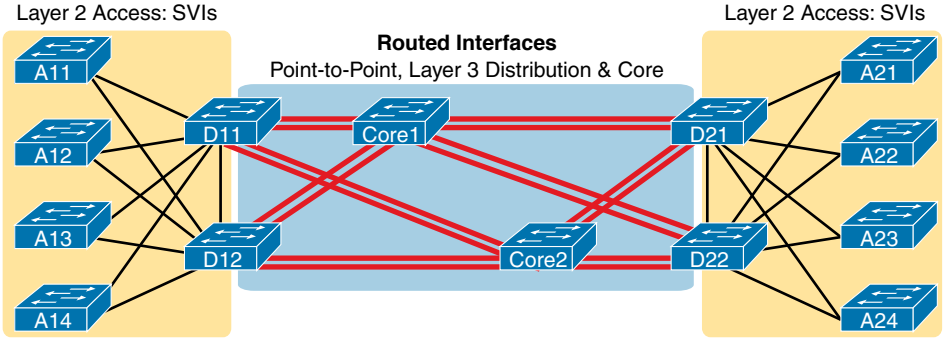


Figure 17-6 *Two Links Between Each Distribution and Core Switch*

While each individual port in the distribution and core could be treated as a separate routed port, it is better to combine each pair of parallel links into a Layer 3 EtherChannel. Without using EtherChannel, you can still make each port on each switch in the center of the figure be a routed port. It works. However, once you enable a routing protocol but don't use EtherChannels, each Layer 3 switch will now learn two IP routes with the same neighboring switch as the next hop—one route over one link, another route over the other link.

Using a Layer 3 EtherChannel makes more sense with multiple parallel links between two switches. By doing so, each pair of links acts as one Layer 3 link. So, each pair of switches has one routing protocol neighbor relationship with the neighbor, and not two. Each switch learns one route per destination per pair of links, and not two. IOS then balances the traffic, often with better balancing than the balancing that occurs with the use of multiple IP routes to the same subnet. Overall, the Layer 3 EtherChannel approach works much better than leaving each link as a separate routed port and using Layer 3 balancing.

Compared to what you have already learned, configuring a Layer 3 EtherChannel takes only a little more work. Chapter 10 already showed you how to configure an EtherChannel. This chapter has already shown how to make a port a Layer 3 routed port. Next, you have to combine the two ideas by combining both the EtherChannel and routed port configuration. The following checklist shows the steps, assuming a static definition.

Config Checklist

Step 1. Configure the physical interfaces as follows, in interface configuration mode:

A. Add the **channel-group** *number* **mode on** command to add it to the channel. Use the same number for all physical interfaces on the same switch, but the number used (the channel-group number) can differ on the two neighboring switches.

B. Add the **no switchport** command to make each physical port a routed port.

Step 2. Configure the PortChannel interface:

A. Use the **interface port-channel** *number* command to move to port-channel configuration mode for the same channel number configured on the physical interfaces.

B. Add the **no switchport** command to make sure that the port-channel interface acts as a routed port. (IOS may have already added this command.)

C. Use the **ip address** *address mask* command to configure the address and mask.

NOTE Cisco uses the term *EtherChannel* in concepts discussed in this section and then uses the term *PortChannel*, with command keyword **port-channel**, when verifying and configuring EtherChannels. For the purposes of understanding the technology, you may treat these terms as synonyms. However, it helps to pay close attention to the use of the terms *PortChannel* and *EtherChannel* as you work through the examples in this section because IOS uses both.

Example 17-12 shows an example of the configuration for a Layer 3 EtherChannel for switch SW1 in Figure 17-7. The EtherChannel defines port-channel interface 12 and uses subnet 10.1.12.0/24.

Key Topic

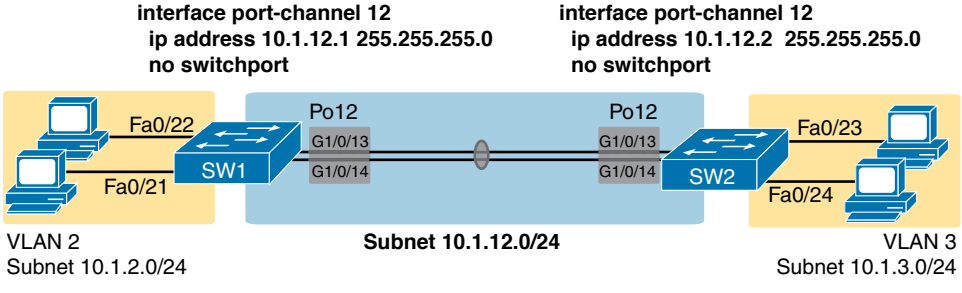


Figure 17-7 *Design Used in EtherChannel Configuration Examples*

Example 17-12 *Layer 3 EtherChannel Configuration on Switch SW1*

```
interface GigabitEthernet1/0/13
 no switchport
 no ip address
 channel-group 12 mode on
!
interface GigabitEthernet1/0/14
 no switchport
 no ip address
 channel-group 12 mode on
!
interface Port-channel12
 no switchport
 ip address 10.1.12.1 255.255.255.0
```

Of particular importance, note that although the physical interfaces and PortChannel interface are all routed ports, the IP address should be placed on the PortChannel interface only. In fact, when the **no switchport** command is configured on an interface, IOS adds the **no ip address** command to the interface. Then configure the IP address on the PortChannel interface only.

Once configured, the PortChannel interface appears in several commands, as shown in Example 17-13. The commands that list IP addresses and routes refer to the PortChannel interface. Also, note that the **show interfaces status** command lists the fact that the physical ports and the port-channel 12 interface are all routed ports.

Example 17-13 *Verification Commands Listing Interface Port-Channel 12 from Switch SW1*

```
SW1# show interfaces port-channel 12
Port-channel12 is up, line protocol is up (connected)
  Hardware is EtherChannel, address is bcc4.938b.e543 (bia bcc4.938b.e543)
  Internet address is 10.1.12.1/24
! lines omitted for brevity

SW1# show interfaces status
! Only ports related to the example are shown.
Port      Name               Status       Vlan       Duplex  Speed Type
Gi1/0/13                     connected    routed     a-full a-1000 10/100/1000BaseTX
Gi1/0/14                     connected    routed     a-full a-1000 10/100/1000BaseTX
Po12                         connected    routed     a-full a-1000

SW1# show ip route
! legend omitted for brevity
       10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C         10.1.2.0/24 is directly connected, Vlan2
L         10.1.2.1/32 is directly connected, Vlan2
C         10.1.12.0/24 is directly connected, Port-channel12
L         10.1.12.1/32 is directly connected, Port-channel12
```

For a final bit of verification, you can examine the EtherChannel directly with the **show etherchannel summary** command as listed in Example 17-14. Note in particular that it lists a flag legend for characters that identify key operational states, such as whether a port is bundled (included) in the PortChannel (P) and whether it is acting as a routed (R) or switched (S) port.

Example 17-14 *Verifying the EtherChannel*

```
SW1# show etherchannel 12 summary
Flags: D - down        P - bundled in port-channel
       I - stand-alone s - suspended
       H - Hot-standby (LACP only)
       R - Layer3      S - Layer2
       U - in use      f - failed to allocate aggregator

       M - not in use, minimum links not met
       u - unsuitable for bundling
       w - waiting to be aggregated
       d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
12     Po12(RU)          -        Gi1/0/13(P) Gi1/0/14(P)
```

Troubleshooting Layer 3 EtherChannels

When you are troubleshooting a Layer 3 EtherChannel, there are two main areas to consider. First, you need to look at the configuration of the **channel-group** command, which enables an interface for an EtherChannel. Second, you should check a list of settings that must match on the interfaces for a Layer 3 EtherChannel to work correctly.

As for the **channel-group** interface subcommand, this command can enable EtherChannel statically or dynamically. If dynamic, this command's keywords imply either Port Aggregation Protocol (PaGP) or Link Aggregation Control Protocol (LACP) as the protocol to negotiate between the neighboring switches whether they put the link into the EtherChannel.

If all this sounds vaguely familiar, it is the exact same configuration covered way back in the Chapter 10 section "Configuring Dynamic EtherChannels." The configuration of the **channel-group** subcommand is exactly the same, with the same requirements, whether configuring Layer 2 or Layer 3 EtherChannels. So, it might be a good time to review those EtherChannel configuration details from Chapter 10. However, regardless of when you review and master those commands, note that the configuration of the EtherChannel (with the **channel-group** subcommand) is the same, whether Layer 2 or Layer 3.

Additionally, you must do more than just configure the **channel-group** command correctly for all the physical ports to be bundled into the EtherChannel. Layer 2 EtherChannels have a longer list of requirements, but Layer 3 EtherChannels also require a few consistency checks between the ports before they can be added to the EtherChannel. The following is the list of requirements for Layer 3 EtherChannels:

Key Topic

no switchport: The PortChannel interface must be configured with the **no switchport** command, and so must the physical interfaces. If a physical interface is not also configured with the **no switchport** command, it will not become operational in the EtherChannel.

Speed: The physical ports in the channel must use the same speed.

duplex: The physical ports in the channel must use the same duplex.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 17-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 17-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review config checklists		Book, website
Review command tables		Book
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 17-3 Key Topics for Chapter 17

Key Topic Element	Description	Page Number
Figure 17-2	Concept of VLAN subinterfaces on a router	396
List	Two alternative methods to configure the native VLAN in a ROAS configuration	398
List	Troubleshooting suggestions for ROAS configuration	401
Figure 17-3	Layer 3 switching with SVIs concept and configuration	402

Key Topic Element	Description	Page Number
List	Troubleshooting suggestions for correct operation of a Layer 3 switch that uses SVIs	405
Figure 17-4	Layer 3 switching with routed ports concept and configuration	407
List	**show** commands that list Layer 3 routed ports in their output	408
Figure 17-7	Layer 3 EtherChannel concept and configuration	411
List	List of configuration settings that must be consistent before IOS will bundle a link with an existing Layer 3 EtherChannel	414

Key Terms You Should Know

router-on-a-stick (ROAS), switched virtual interface (SVI), VLAN interface, Layer 3 EtherChannel (L3 EtherChannel), routed port, Layer 3 switch, multilayer switch, subinterfaces

Command References

Tables 17-4 and 17-5 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 17-4 Chapter 17 Configuration Command Reference

Command	Description
interface *type number.subint*	Router global command to create a subinterface and to enter configuration mode for that subinterface
encapsulation dot1q *vlan-id* [**native**]	Router subinterface subcommand that tells the router to use 802.1Q trunking, for a particular VLAN, and with the **native** keyword, to not encapsulate in a trunking header
[**no**] **ip routing**	Global command that enables (**ip routing**) or disables (**no ip routing**) the routing of IPv4 packets on a router or Layer 3 switch
interface vlan *vlan-id*	A switch global command on a Layer 3 switch to create a VLAN interface and to enter configuration mode for that VLAN interface
sdm prefer lanbase-routing	Command on some Cisco switches that reallocates forwarding chip memory to allow for an IPv4 routing table
[**no**] **switchport**	Layer 3 switch subcommand that makes the port act as a Layer 2 port (**switchport**) or Layer 3 routed port (**no switchport**)

Command	Description
interface port-channel *channel-number*	A switch command to enter PortChannel configuration mode and also to create the PortChannel if not already created
channel-group *channel-number* **mode** {**auto** | **desirable** | **active** | **passive** | **on**}	Interface subcommand that enables EtherChannel on the interface

Table 17-5 Chapter 17 EXEC Command Reference

Command	Description
show ip route	Lists the router's entire routing table
show ip route [**connected**]	Lists a subset of the IP routing table
show vlans	Lists VLAN configuration and statistics for VLAN trunks configured on routers
show interfaces [**interface** *type number*]	Lists detailed status and statistical information, including IP address and mask, about all interfaces (or the listed interface only)
show interfaces [**interface** *type number*] **status**	Among other facts, for switch ports, lists the access VLAN or the fact that the interface is a trunk; or, for routed ports, lists "routed"
show interfaces *interface-id* **switchport**	For switch ports, lists information about any interface regarding administrative settings and operational state; for routed ports, the output simply confirms the port is a routed (not switched) port
show interfaces vlan *number*	Lists the interface status, the switch's IPv4 address and mask, and much more
show etherchannel [*channel-group-number*] **summary**	Lists information about the state of EtherChannels on this switch, including whether the channel is a Layer 2 or Layer 3 EtherChannel

This page intentionally left blank

CHAPTER 18

Troubleshooting IPv4 Routing

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.6 Configure and verify IPv4 addressing and subnetting

3.0 IP Connectivity

3.3 Configure and verify IPv4 and IPv6 static routing

3.3.a Default route

3.3.b Network route

3.3.c Host route

3.3.d Floating static

The first three chapters in this part of the book took you from a starting point of understanding IP addressing and subnetting to the details of implementing IP addressing, routing between connected subnets, and configuring static routes. All those steps include the idea of configuring a command and seeing a route show up in the IP routing table on that same router.

This chapter turns our attention to routing from end-to-end across an entire enterprise network. How do you troubleshoot an IPv4 network? How do you verify correct operation, identify root causes, and fix those for various IP routing features? How do you do that in the presence of an IP addressing and subnetting plan, requiring you to apply all that subnetting math from Part IV of this book and the basic address/mask and static route configuration from the other chapters here in Part V? This chapter answers some of those questions.

In particular, this chapter focuses on two tools and how to use them: ping and traceroute. Both tools test the IPv4 data plane; that is, the ability of each networking device to route or forward IPv4 packets. This chapter devotes a major section each to **ping** and **traceroute**. The chapter then ends with a short discussion of two other router tools that can also be useful for troubleshooting: Telnet and Secure Shell (SSH).

"Do I Know This Already?" Quiz

I put DIKTA quizzes in most of the chapters as a tool to help you decide how to approach reading a chapter. However, this chapter does not have a DIKTA quiz because I think you should read it regardless of your prior knowledge. As with all chapters in this book, this chapter introduces new concepts, but it also acts as a tool to review and deepen your understanding of IP routing. I hope you enjoy the perspectives on using ping and traceroute in this chapter.

Foundation Topics

Problem Isolation Using the ping Command

Someone sends you an email or text, or a phone message, asking you to look into a user's network problem. You Secure Shell (SSH) to a router and issue a **ping** command that works. What does that result rule out as a possible reason for the problem? What does it rule in as still being a possible root cause?

Then you issue another **ping** to another address, and this time the ping fails. Again, what does the failure of that **ping** command tell you? What parts of IPv4 routing may still be a problem, and what parts do you now know are not a problem?

The **ping** command gives us one of the most common network troubleshooting tools. When the **ping** command succeeds, it confirms many individual parts of how IP routing works, ruling out some possible causes of the current problem. When a **ping** command fails, it often helps narrow down where in the internetwork the root cause of the problem may be happening, further isolating the problem.

This section begins with a brief explanation of how ping works. It then moves on to some suggestions and analysis of how to use the **ping** command to isolate problems by removing some items from consideration.

Ping Command Basics

The **ping** command tests connectivity by sending packets to an IP address, expecting the device at that address to send packets back. The command sends packets that mean "if you receive this packet, and it is addressed to you, send a reply back." Each time the **ping** command sends one of these packets and receives the message sent back by the other host, the **ping** command knows a packet made it from the source host to the destination and back.

More formally, the **ping** command uses the Internet Control Message Protocol (ICMP), specifically the ICMP echo request and ICMP echo reply messages. ICMP defines many other messages as well, but these two messages were made specifically for connectivity testing by commands like ping. As a protocol, ICMP does not rely on TCP or UDP, and it does not use any application layer protocol. It functions as part of Layer 3, as a control protocol to assist IP by helping manage the IP network functions.

Figure 18-1 shows the ICMP messages, with IP headers, in an example. In this case, the user at host A opens a command prompt and issues the **ping 172.16.2.101** command, testing connectivity to host B. The command sends one echo request and waits (Step 1); host B receives the messages and sends back an echo reply (Step 2).

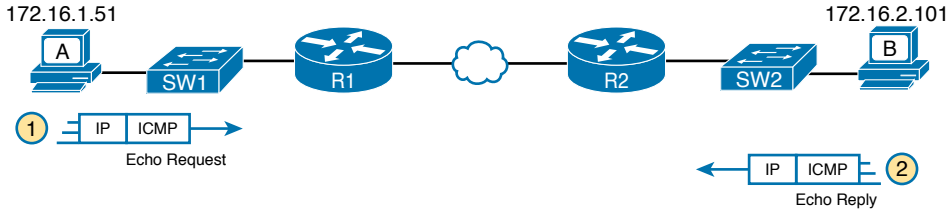


Figure 18-1 *Concept Behind* **ping 172.16.2.101** *on Host A*

The **ping** command is supported on many different devices and many common operating systems. The command has many options: the name or IP address of the destination, how many times the command should send an echo request, how long the command should wait (timeout) for an echo reply, how big to make the packets, and many other options. Example 18-1 shows a sample from host A, with the same command that matches the concept in Figure 18-1: a **ping 172.16.2.101** command on host A.

Example 18-1 *Sample Output from Host A's* **ping 172.16.2.101** *Command*

```
Wendell-Odoms-iMac:~ wendellodom$ ping 172.16.2.101
PING 172.16.2.101 (172.16.2.101): 56 data bytes
64 bytes from 172.16.2.101: icmp_seq=0 ttl=64 time=1.112 ms
64 bytes from 172.16.2.101: icmp_seq=1 ttl=64 time=0.673 ms
64 bytes from 172.16.2.101: icmp_seq=2 ttl=64 time=0.631 ms
64 bytes from 172.16.2.101: icmp_seq=3 ttl=64 time=0.674 ms
64 bytes from 172.16.2.101: icmp_seq=4 ttl=64 time=0.642 ms
64 bytes from 172.16.2.101: icmp_seq=5 ttl=64 time=0.656 ms
^C
--- 172.16.2.101 ping statistics ---
6 packets transmitted, 6 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.631/0.731/1.112/0.171 ms
```

Strategies and Results When Testing with the ping Command

Often, the person handling initial calls from users about problems (often called a customer support rep, or CSR) cannot issue **ping** commands from the user's device. In some cases, talking users through typing the right commands and making the right clicks on their machines can be a problem. Or, the user just might not be available. As an alternative, using different **ping** commands from different routers can help isolate the problem.

The problem with using **ping** commands from routers, instead of from the host that has the problem, is that no single router **ping** command can exactly replicate a **ping** command done from the user's device. However, each different **ping** command can help isolate a problem further. The rest of this section of **ping** commands discusses troubleshooting IPv4 routing by using various **ping** commands from the command-line interface (CLI) of a router.

Testing Longer Routes from Near the Source of the Problem

Most problems begin with some idea like "host X cannot communicate with host Y." A great first troubleshooting step is to issue a **ping** command from X for host Y's IP address. However, assuming the engineer does not have access to host X, the engineer can instead issue the **ping** from the router nearest X, typically the router acting as host X's default gateway.

For instance, in Figure 18-1, imagine that the user of host A had called IT support with a problem related to sending packets to host B. A **ping 172.16.2.101** command on host A would be a great first troubleshooting step, but the CSR cannot access host A or get in touch with the user of host A. So, the CSR telnets to Router R1 and pings host B from there, as shown in Example 18-2.

Example 18-2 *Router R1 Pings Host B (Two Commands)*

```
R1# ping 172.16.2.101
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms
R1# ping 172.16.2.101
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```


First, take a moment to review the output of the first IOS **ping** command. By default, the Cisco IOS **ping** command sends five echo messages, with a timeout of 2 seconds. If the command does not receive an echo reply within 2 seconds, the command considers that message to be a failure, and the command lists a period. If a successful reply is received within 2 seconds, the command displays an exclamation point. So, in this first command, the first echo reply timed out, whereas the other four received a matching echo reply within 2 seconds.

As a quick aside, the example shows a common and normal behavior with **ping** commands: the first **ping** command shows one failure to start, but then the rest of the messages work. This usually happens because some device in the end-to-end route is missing an ARP table entry.

Now think about troubleshooting and what a working **ping** command tells us about the current behavior of this internetwork. First, focus on the big picture for a moment:

- R1 can send ICMP echo request messages to host B (172.16.2.101).
- R1 sends these messages from its outgoing interface's IP address (by default), 172.16.4.1 in this case.
- Host B can send ICMP echo reply messages to R1's 172.16.4.1 IP address (hosts send echo reply messages to the IP address from which the echo request was received).

Figure 18-2 shows the packet flow.

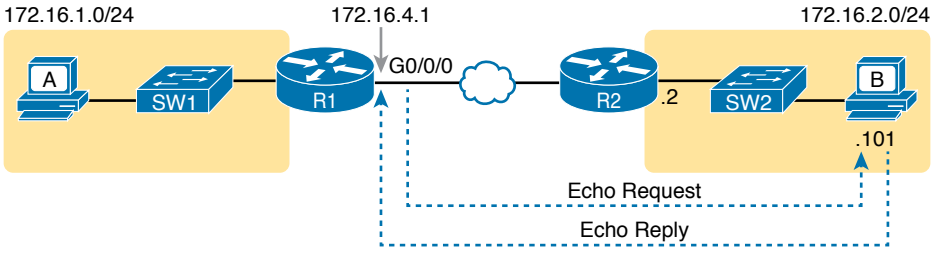


Figure 18-2 *Standard* **ping 172.6.2.101** *Command Using the Source Interface IP Address*

Next, think about IPv4 routing. In the forward direction, R1 must have a route that matches host B's address (172.16.2.101); this route will be either a static route or one learned with a routing protocol. R2 also needs a route for host B's address, in this case a connected route to B's subnet (172.16.2.0/24), as shown in the top arrow lines in Figure 18-3.

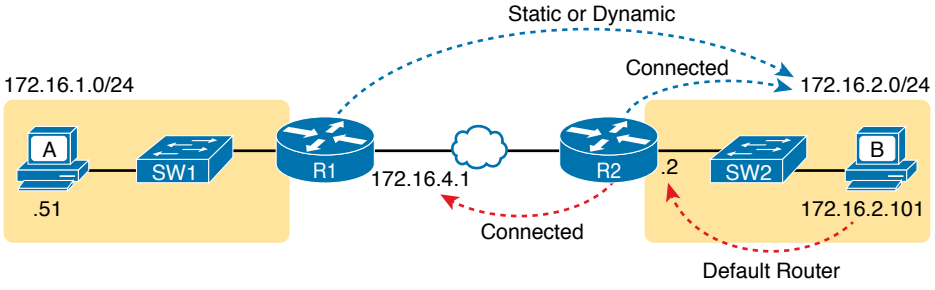


Figure 18-3 *Layer 3 Routes Needed for R1's Ping 172.16.2.101 to Work*

The arrow lines on the bottom of Figure 18-3 show the routes needed to forward the ICMP echo reply message back to Router R1's 172.16.4.1 interface. First, host B must have a valid default router setting because 172.16.4.1 sits in a different subnet than host B. R2 must also have a route that matches destination 172.16.4.1 (in this case, likely to be a connected route).

The working **ping** commands in Example 18-2 also require the data-link and physical layer details to be working. The WAN link must be working: The router interfaces must be up/up, which typically indicates that the link can pass data. On the LAN, R2's LAN interface must be in an up/up state. In addition, everything discussed about Ethernet LANs must be working because the **ping** confirmed that the packets went all the way from R1 to host B and back. In particular

- The switch interfaces in use are in a connected (up/up) state.
- Port security (discussed in the *CCNA 200-301 Official Cert Guide, Volume 2*) does not filter frames sent by R2 or host B.
- STP has placed the right ports into a forwarding state.

The **ping 172.16.2.101** command in Example 18-2 also confirms that IP access control lists (ACL) did not filter the ICMP messages. One ACL contains a set of matching rules and actions: some matched packets are filtered (discarded), while others can continue on their path as normal. ACLs can examine packets as they enter or exit a router interface, so Figure 18-4 shows the various locations on routers R1 and R2 where an ACL could have filtered (discarded) the ICMP messages. (Note that an outbound ACL on router R1 would not filter packets created on R1, so there is no rightward-facing arrow over R1.)

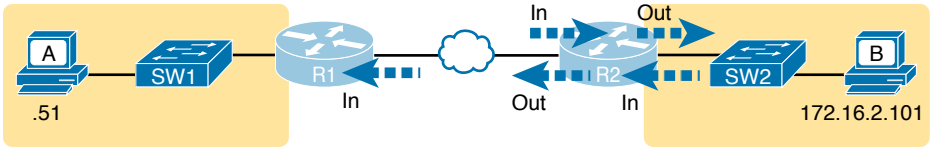


Figure 18-4 *Locations Where IP ACLs Could Have Filtered the Ping Messages*

Finally, the working **ping 172.16.2.101** command on R1 can also be used to reasonably predict that ARP worked and that switch SW2 learned MAC addresses for its MAC address table. R2 and host B need to know each other's MAC addresses so that they can encapsulate the IP packet inside an Ethernet frame, which means both must have a matching ARP table entry. The switch learns the MAC address used by R2 and by host B when it sends the ARP messages or when it sends the frames that hold the IP packets. Figure 18-5 shows the type of information expected in those tables.

Key Topic

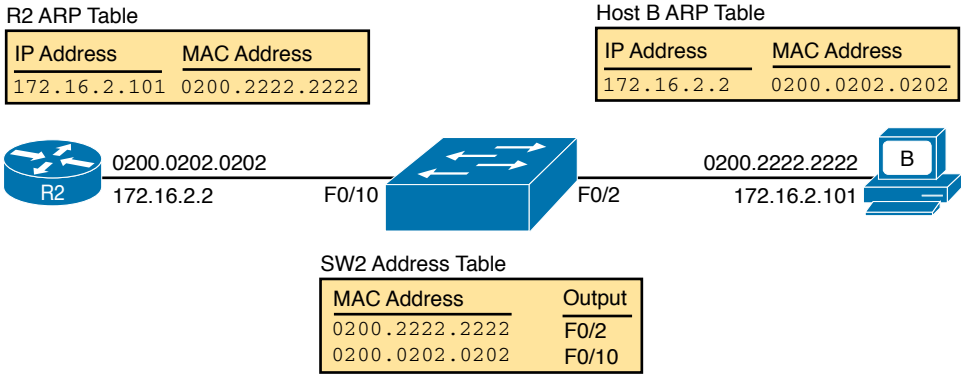


Figure 18-5 *Router and Host ARP Tables, with the Switch MAC Address Table*

As you can see from the last few pages, a strategy of using a **ping** command from near the source of the problem can rule out a lot of possible root causes of any problems between two hosts—assuming the **ping** command succeeds. However, this **ping** command does not act exactly like the same **ping** command on the actual host. To overcome some of what is missing in the **ping** command from a nearby router, the next several examples show some strategies for testing other parts of the path between the two hosts that might have a current problem.

Using Extended Ping to Test the Reverse Route

Pinging from the default router, as discussed in the past few pages, misses an opportunity to test IP routes more fully. In particular, it does not test the reverse route back toward the original host.

For instance, referring to the internetwork in Figure 18-2 again, note that the reverse routes do not point to an address in host A's subnet. When R1 processes the **ping 172.16.2.101** command, R1 has to pick a source IP address to use for the echo request, and routers choose the *IP address of the outgoing interface*. The echo request from R1 to host B flows with source IP address 172.16.4.1 (R1's G0/0/0 IP address). The echo reply flows back to that same address (172.16.4.1).

A standard ping often does not test the reverse route that you need to test. In this case, the standard **ping 172.16.2.101** command on R1 does not test whether the routers can route back to subnet 172.16.1.0/24, instead testing their routes for subnet172.16.4.0. A better ping test would test the route back to host A's subnet; an extended ping from R1 can cause that test to happen. Extended ping allows R1's **ping** command to use R1's LAN IP address from within subnet 172.16.1.0/24. Then, the echo reply messages would flow to host A's subnet, as shown in Figure 18-6.

Key Topic

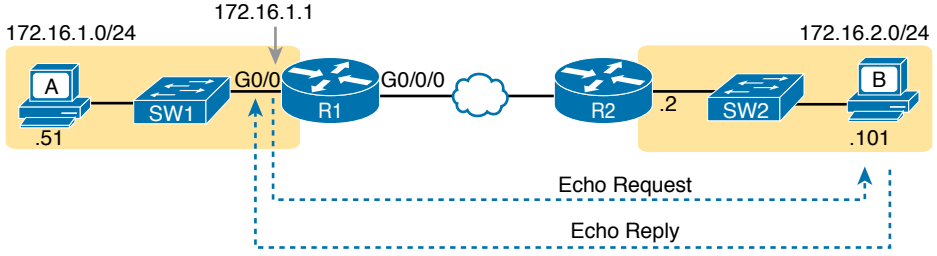


Figure 18-6 *Extended Ping Command Tests the Route to 172.16.1.51 (Host A)*

The extended **ping** command does allow the user to type all the parameters on a potentially long command, but it also allows users to simply issue the **ping** command, press **Enter**, with IOS then asking the user to answer questions to complete the command, as shown in Example 18-3. The example shows the **ping** command on R1 that matches the logic in Figure 18-6. This same command could have been issued from the command line as **ping 172.16.2.101 source 172.16.1.1.**

Example 18-3 *Testing the Reverse Route Using the Extended Ping*

```
R1# ping
Protocol [ip]:
Target IP address: 172.16.2.101
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 172.16.1.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
Packet sent with a source address of 172.16.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

This particular extended **ping** command tests the same routes for the echo request going to the right, but it forces a better test of routes pointing back to the left for the ICMP echo reply. For that direction, R2 needs a route that matches address 172.16.1.1, which is likely to be a route for subnet 172.16.1.0/24—the same subnet in which host A resides.

From a troubleshooting perspective, using both standard and extended **ping** commands can be useful. However, neither can exactly mimic a **ping** command created on the host itself because the routers cannot send packets with the host's IP address. For instance, the extended **ping** in Example 18-3 uses source IP address 172.16.1.1, which is not host A's IP address. As a result, neither the standard or extended **ping** commands in these two examples so far in this chapter can test for some kinds of problems, such as the following:

- IP ACLs that discard packets based on host A's IP address but allow packets that match the router's IP address
- LAN switch port security that filters A's frames (based on A's MAC address)
- IP routes on routers that happen to match host A's 172.16.1.51 address, with different routes that match R1's 172.16.1.1 address
- Problems with host A's default router setting

NOTE IP ACLs and LAN switch port security are covered in *CCNA 200-301 Official Cert Guide, Volume 2*. For now, know that IP ACLs can filter packets on routers, focusing on the Layer 3 and 4 headers. Port security can be enabled on Layer 2 switches to filter based on source MAC addresses.

Testing LAN Neighbors with Standard Ping

Testing using a **ping** of another device on the LAN can quickly confirm whether the LAN can pass packets and frames. Specifically, a working **ping** rules out many possible root causes of a problem. For instance, Figure 18-7 shows the ICMP messages that occur if R1 issues the command **ping 172.16.1.51**, pinging host A, which sits on the same VLAN as R1.

Key Topic

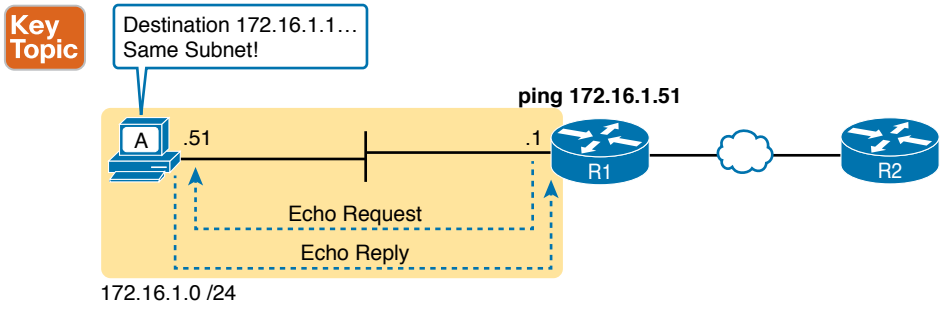


Figure 18-7 *Standard* **ping** *Command Confirms That the LAN Works*

If the ping works, it confirms the following, which rules out some potential issues:

- The host with address 172.16.1.51 replied.
- The LAN can pass unicast frames from R1 to host 172.16.1.51 and vice versa.

- You can reasonably assume that the switches learned the MAC addresses of the router and the host, adding those to the MAC address tables.
- Host A and Router R1 completed the ARP process and list each other in their respective Address Resolution Protocol (ARP) tables.

The failure of a ping, even with two devices on the same subnet, can point to a variety of problems, like those mentioned in this list. For instance, if the **ping 172.16.1.51** on R1 fails (Figure 18-7), that result points to this list of potential root causes:

Key Topic

- **IP addressing problem:** Host A could be statically configured with the wrong IP address.
- **DHCP problems:** If you are using Dynamic Host Configuration Protocol (DHCP), many problems could exist. Chapter 7, "Implementing DHCP" in *CCNA 200-301 Official Cert Guide, Volume 2*, discusses those possibilities in some depth.
- **VLAN trunking problems:** The router could be configured for 802.1Q trunking, when the switch is not (or vice versa).
- **LAN problems:** A wide variety of issues could exist with the Layer 2 switches, preventing any frames from flowing between host A and the router.

So, whether the ping works or fails, simply pinging a LAN host from a router can help further isolate the problem.

Testing LAN Neighbors with Extended Ping

A standard ping of a LAN host from a router does not test that host's default router setting. However, an extended ping can test the host's default router setting. Both tests can be useful, especially for problem isolation, because

Key Topic

- If a standard ping of a local LAN host works…
- But an extended ping of the same LAN host fails…
- The problem likely relates somehow to the host's default router setting.

First, to understand why the standard and extended ping results have different effects, consider first the standard **ping 172.16.1.51** command on R1, as shown previously in Figure 18-7. As a standard **ping** command, R1 used its LAN interface IP address (172.16.1.1) as the source of the ICMP Echo. So, when the host (A) sent back its ICMP echo reply, host A considered the destination of 172.16.1.1 as being on the same subnet. Host A's ICMP echo reply message, sent back to 172.16.1.1, would work even if host A did not have a default router setting at all!

In comparison, Figure 18-8 shows the difference when using an extended ping on Router R1. An extended ping from local Router R1, using R1's S0/0/0 IP address of 172.16.4.1 as the source of the ICMP echo request, means that host A's ICMP echo reply will flow to an address in another subnet, which makes host A use its default router setting.

The comparison between the previous two figures shows one of the most classic mistakes when troubleshooting networks. Sometimes, the temptation is to connect to a router and ping the host on the attached LAN, and it works. So, the engineer moves on, thinking that the network layer issues between the router and host work fine, when the problem still exists with the host's default router setting.

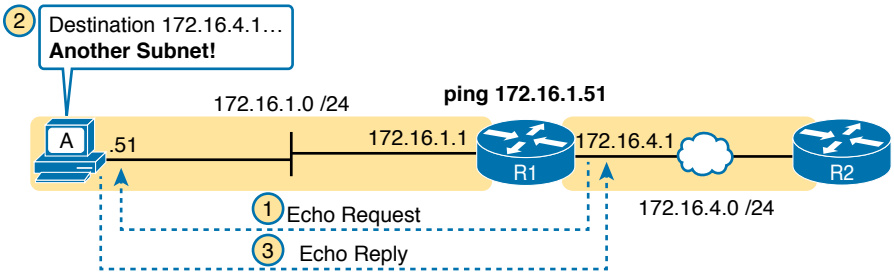


Figure 18-8 *Extended* **ping** *Command Does Test Host A's Default Router Setting*

Testing WAN Neighbors with Standard Ping

As with a standard ping test across a LAN, a standard ping test between routers over a serial or Ethernet WAN link tests whether the link can pass IPv4 packets. With a properly designed IPv4 addressing plan, two routers on the same serial or Ethernet WAN link should have IP addresses in the same subnet. A ping from one router to the IP address of the other router confirms that an IP packet can be sent over the link and back, as shown in the **ping 172.16.4.2** command on R1 in Figure 18-9.

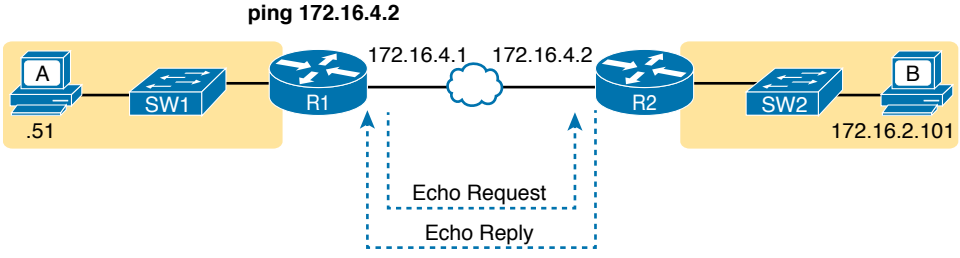


Figure 18-9 *Pinging Across a WAN Link*

A successful ping of the IP address on the other end of an Ethernet WAN link that sits between two routers confirms several specific facts, such as the following:

- Both routers' WAN interfaces are in an up/up state.
- The Layer 1 and 2 features of the link work.
- The routers believe that the neighboring router's IP address is in the same subnet.
- Inbound ACLs on both routers do not filter the incoming packets, respectively.
- The remote router is configured with the expected IP address (172.16.4.2 in this case).

Testing by pinging the other neighboring router does not test many other features. However, although the test is limited in scope, it does let you rule out WAN links as having a Layer 1 or 2 problem, and it rules out some basic Layer 3 addressing problems.

Using Ping with Names and with IP Addresses

All the ping examples so far in this chapter show a ping of an IP address. However, the **ping** command can use hostnames, and pinging a hostname allows the network engineer to further test whether the Domain Name System (DNS) process works.

First, most every TCP/IP application today uses hostnames rather than IP addresses to identify the other device. No one opens a web browser and types in 72.163.4.185. Instead, they type in a web address, like www.cisco.com, which includes the hostname www.cisco.com. Then, before a host can send data to a specific IP address, the host must first ask a DNS server to resolve that hostname into the matching IP address.

For example, in the small internetwork used for several examples in this chapter, a **ping B** command on host A tests A's DNS settings, as shown in Figure 18-10. When host A sees the use of a hostname (B), it first looks in its local DNS name cache to find out whether it has already resolved the name B. If not, host A first asks the DNS to supply (resolve) the name into its matching IP address (Step 1 in the figure). Only then does host A send a packet to 172.16.2.101, host B's IP address (Step 2).

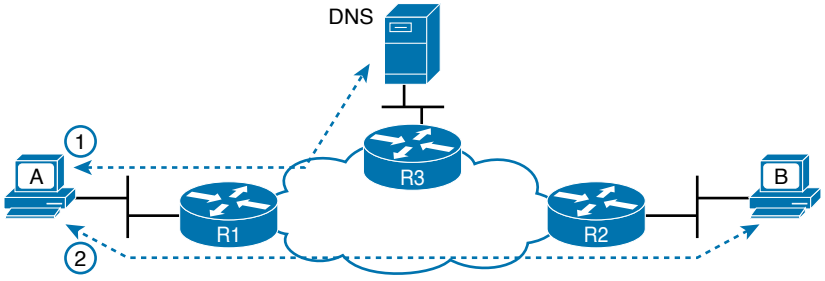


Figure 18-10 *DNS Name Resolution by Host A*

When troubleshooting, testing from the host by pinging using a hostname can be very helpful. The command, of course, tests the host's own DNS client settings. For instance, a classic comparison is to first ping the destination host using the hostname, which requires a DNS request. Then, repeat the same test, but use the destination host's IP address instead of its name, which does not require the DNS request. If the ping of the hostname fails but the ping of the IP address works, the problem usually has something to do with DNS.

Problem Isolation Using the traceroute Command

Like **ping**, the **traceroute** command helps network engineers isolate problems. Here is a comparison of the two:

Key Topic

- Both send messages in the network to test connectivity.
- Both rely on other devices to send back a reply.
- Both have wide support on many different operating systems.
- Both can use a hostname or an IP address to identify the destination.
- On routers, both have a standard and extended version, allowing better testing of the reverse route.

The biggest differences relate to the more detailed results in the output of the **traceroute** command and the extra time and effort it takes **traceroute** to build that output. This second major section examines how **traceroute** works; plus it provides some suggestions on how to use this more detailed information to more quickly isolate IP routing problems.

traceroute Basics

Imagine some network engineer or CSR starts to troubleshoot some problem. The engineer pings from the user's host, pings from a nearby router, and after a few commands, convinces herself that the host can indeed send and receive IP packets. The problem might not be solved yet, but the problem does not appear to be a network problem.

Now imagine the next problem comes along, and this time the **ping** command fails. It appears that some problem does exist in the IP network. Where is the problem? Where should the engineer look more closely? Although the **ping** command can prove helpful in isolating the source of the problem, the **traceroute** command may be a better option. The **traceroute** command systematically helps pinpoint routing problems by showing how far a packet goes through an IP network before being discarded.

The **traceroute** command identifies the routers in the path from source host to destination host. Specifically, it lists the next-hop IP address of each router that would be in each of the individual routes. For instance, a **traceroute 172.16.2.101** command on host A in Figure 18-11 would identify an IP address on Router R1, another on Router R2, and then host B, as shown in the figure. Example 18-4, which follows, lists the output of the command, taken from host A.

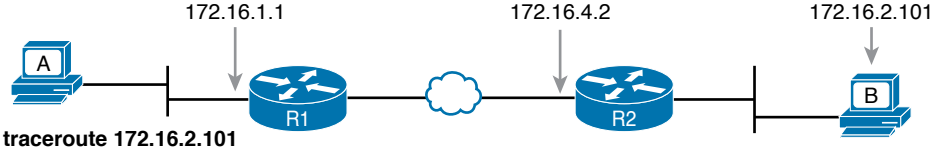


Figure 18-11 *IP Addresses Identified by a Successful* **traceroute 172.16.2.101** *Command*

Example 18-4 *Output from* **traceroute 172.16.2.101** *on Host A*

```
Wendell-Odoms-iMac:~ wendellodom$ traceroute 172.16.2.101
traceroute to 172.16.2.101, 64 hops max, 52 byte packets
  1 172.16.1.1 (172.16.1.1) 0.870 ms 0.520 ms 0.496 ms
  2 172.16.4.2 (172.16.4.2) 8.263 ms 7.518 ms 9.319 ms
  3 172.16.2.101 (172.16.2.101) 16.770 ms 9.819 ms 9.830 ms
```

How the traceroute Command Works

The **traceroute** command gathers information by generating packets that trigger error messages from routers; these messages identify the routers, letting the **traceroute** command list the routers' IP addresses in the output of the command. That error message is the ICMP Time-to-Live Exceeded (TTL Exceeded) message, originally meant to notify hosts when a packet had been looping around a network.

Ignoring traceroute for a moment and instead focusing on IP routing, IPv4 routers defeat routing loops in part by discarding looping IP packets. To do so, the IPv4 header holds a field called Time To Live (TTL). The original host that creates the packet sets an initial TTL value. Then each router that forwards the packet decrements the TTL value by 1. When a router decrements the TTL to 0, the router perceives the packet is looping, and the router discards the packet. The router also notifies the host that sent the discarded packet by sending an ICMP TTL Exceeded message.

Now back to traceroute. Traceroute sends messages with low TTL values to make the routers send back a TTL Exceeded message. Specifically, a **traceroute** command begins by sending several packets (usually three), each with the header TTL field equal to 1. When that packet arrives at the next router—host A's default Router R1 in the example of Figure 18-12—the router decrements TTL to 0 and discards the packet. The router then sends host A the TTL Exceeded message, which identifies the router's IP address to the **traceroute** command.

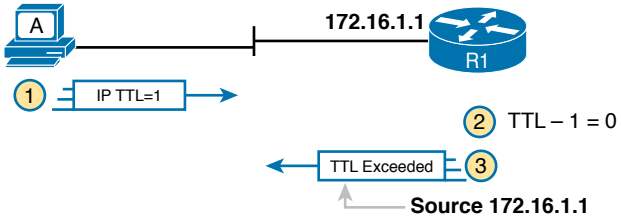


Figure 18-12 *How* **traceroute** *Identifies the First Router in the Route*

The **traceroute** command sends several TTL=1 packets, checking them to see whether the TTL Exceeded messages flow from the same router, based on the source IP address of the TTL Exceeded message. Assuming the messages come from the same router, the **traceroute** command lists that IP address as the next line of output on the command.

To find all the routers in the path, and finally confirm that packets flow all the way to the destination host, the **traceroute** command sends a small set of packets with TTL=1, then a small set with TTL=2, then 3, 4, and so on, until the destination host replies. Figure 18-13 shows the packet from the second set with TTL=2. In this case, one router (R1) actually forwards the packet, while another router (R2) happens to decrement the TTL to 0, causing a TTL Exceeded message to be sent back to host A.

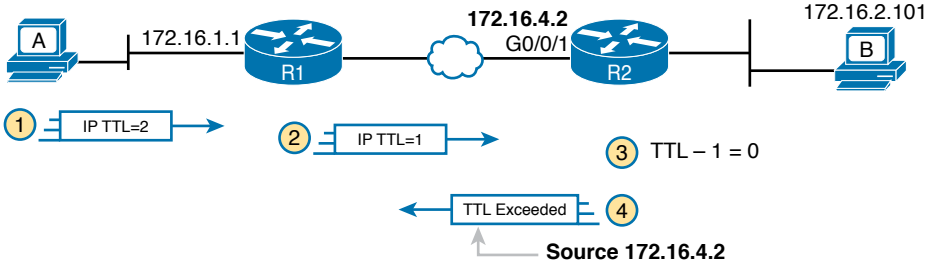


Figure 18-13 *TTL=2 Message Sent by* **traceroute**

The figure shows these four steps:

1. The traceroute command sends a packet from the second set with TTL=2.
2. Router R1 processes the packet and decrements TTL to 1. R1 forwards the packet.
3. Router R2 processes the packet and decrements TTL to 0. R2 discards the packet.
4. R2 notifies the sending host of the discarded packet by sending a TTL Exceeded ICMP message. The source IP address of that message is 172.16.4.2.

Finally, the choice of source IP address to use on the time-exceeded message returned by routers has a big impact on the output of the **traceroute** command. Most routers use simpler

logic that also makes command output like **traceroute** more consistent and meaningful. That logic: choose the TTL Exceeded message's source IP address based on the source interface of the original message that was discarded due to TTL. In the example in Figure 18-13, the original message at Step 2 arrived on R2's G0/0/1 interface, so at Step 3, R2 uses G0/0/1's IP address as the source IP address of the TTL Exceeded message, and as the interface out which to send the message.

Standard and Extended traceroute

The standard and extended options for the **traceroute** command give you many of the same options as the **ping** command. For instance, Example 18-5 lists the output of a standard **traceroute** command on Router R1. Like the standard **ping** command, a standard **traceroute** command chooses an IP address based on the outgoing interface for the packet sent by the command. So, in this example, the packets sent by R1 come from source IP address 172.16.4.1, R1's G0/0/0 IP address.

Example 18-5 *Standard* **traceroute** *Command on R1*

```
R1# traceroute 172.16.2.101
Type escape sequence to abort.
Tracing the route to 172.16.2.101
VRF info: (vrf in name/id, vrf out name/id)
   1 172.16.4.2 0 msec 0 msec 0 msec
   2 172.16.2.101 0 msec 0 msec *
```

The extended **traceroute** command, as shown in Example 18-6, follows the same basic command structure as the extended **ping** command. The user can type all the parameters on one command line, but it is much easier to just type **traceroute**, press **Enter**, and let IOS prompt for all the parameters, including the source IP address of the packets (172.16.1.1 in this example).

Example 18-6 *Extended* **traceroute** *Command on R1*

```
R1# traceroute
Protocol [ip]:
Target IP address: 172.16.2.101
Source address: 172.16.1.1
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Type escape sequence to abort.
Tracing the route to 172.16.2.101
VRF info: (vrf in name/id, vrf out name/id)
   1 172.16.4.2 0 msec 0 msec 0 msec
   2 172.16.2.101 0 msec 0 msec *
```

Both the **ping** and **traceroute** commands exist on most operating systems, including Cisco IOS. However, some operating systems use a slightly different syntax for **traceroute**. For example, most Windows operating systems support **tracert** and **pathping**, and not **traceroute**. Linux and OS X support the **traceroute** command.

NOTE Host OS **traceroute** commands usually create ICMP echo requests. The Cisco IOS **traceroute** command instead creates IP packets with a UDP header. This bit of information may seem trivial at this point. However, note that an ACL may actually filter the traffic from a host's **traceroute** messages but not the router **traceroute** command, or vice versa.

Telnet and SSH

The **ping** and **traceroute** commands do give networkers two great tools to begin isolating the cause of an IP routing problem. However, these two commands tell us nothing about the operation state inside the various network devices. Once you begin to get an idea of the kinds of problems and the possible locations of the problems using **ping** and **traceroute**, the next step is to look at the status of various router and switch features. One way to do that is to use Telnet or Secure Shell (SSH) to log in to the devices.

Common Reasons to Use the IOS Telnet and SSH Client

Normally, a network engineer would log in to the remote device using a Telnet or SSH client on a PC, tablet, or any other user device. In fact, often, the same software package does both Telnet and SSH. However, in some cases, you may want to take advantage of the Telnet and SSH client built in to IOS on the routers and switches to Telnet/SSH from one Cisco device to the next.

To understand why, consider the example shown in Figure 18-14. The figure shows arrowed lines to three separate IP addresses on three separate Cisco routers. PC1 has attempted to Telnet to each address from a different tab in PC1's Telnet/SSH client. However, R2 happens to have an error in its routing protocol configuration, so R1, R2, and R3 fail to learn any routes from each other. As a result, PC1's Telnet attempt to both 10.1.2.2 (R2) and 10.1.3.3 (R3) fails.

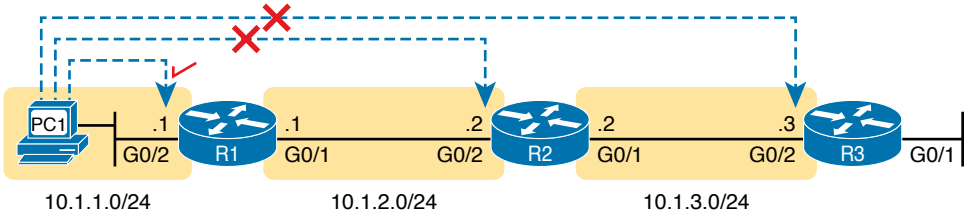


Figure 18-14 *Telnet Works from PC1 to R1 but Not to R2 or R3*

In some cases, like this one, a Telnet or SSH login from the network engineer's device can fail, while you could still find a way to log in using the **telnet** and **ssh** commands to use the Telnet and SSH clients on the routers or switches. With this particular scenario, all the individual data links work; the problem is with the routing protocol exchanging routes. PC1 can

ping R1's 10.1.1.1 IP address, R1 can ping R2's 10.1.2.2 address, and R2 can ping R3's 10.1.3.3 address. Because each link works, and each router can send and receive packets with its neighbor on the shared data link, you could Telnet/SSH to each successive device.

Figure 18-15 shows the idea. On the left, PC1 begins with either a Telnet/SSH or a console connection into Router R1, as shown on the left. Then the user issues the **telnet 10.1.2.2** command from R1 to Telnet to R2. Once logged in to R2, the user can issue commands on R2. Then from R2, the user could issue the **telnet 10.1.3.3** command to Telnet to R3, from which the user could issue commands on R3.

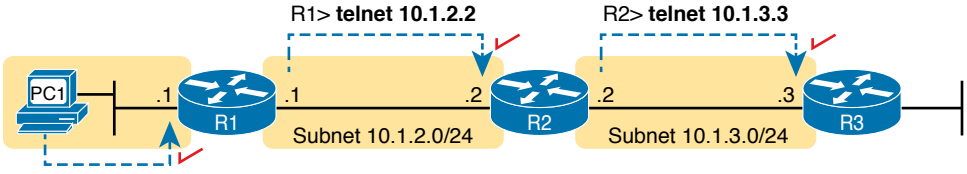


Figure 18-15 *Successive Telnet Connections: PC1 to R1, R1 to R2, and R2 to R3*

The Telnet connections shown in Figure 18-15 work because each Telnet in this case uses source and destination addresses in the same subnet. For example, R1's **telnet 10.1.2.2** command uses 10.1.2.2 as the destination, of course. R1 uses the outgoing interface IP address used to send packets to 10.1.2.2, 10.1.2.1 in this case. Because each of these **telnet** commands connects to an IP address in a connected subnet, the routing protocol could be completely misconfigured, and you could still Telnet/SSH to each successive device to troubleshoot and fix the problem.

Network engineers also use the IOS Telnet and SSH client just for preference. For instance, if you need to log in to several Cisco devices, you could open several windows and tabs on your PC, and log in from your PC (assuming the network was not having problems). Or, you could log in from your PC to some nearby Cisco router or switch, and from there Telnet or SSH to other Cisco devices.

IOS Telnet and SSH Examples

Using the IOS Telnet client via the **telnet** *host* command is pretty simple. Just use the IP address or hostname to identify the host to which you want to connect, and press **Enter**. Example 18-7 shows an example based on Figure 18-15, with R1 using Telnet to connect to 10.1.2.2 (R2).

Example 18-7 *Telnet from R1 to R2 to View Interface Status on R2*

```
R1# telnet 10.1.2.2
Trying 10.1.2.2 ... Open

User Access Verification

Username: wendell
Password:
R2>
R2> show ip interface brief
```

```
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  administratively down down
GigabitEthernet0/1         10.1.3.2        YES manual up                    up
GigabitEthernet0/2         10.1.2.2        YES manual up                    up
GigabitEthernet0/3         unassigned      YES unset  administratively down down
```

Take the time to pay close attention to the command prompts. The example begins with the user logged in to Router R1, with the R1# command prompt. After issuing the **telnet 10.1.2.2** command, R2 asks the user for both a username and password because Router R2 uses local username authentication, which requires those credentials. The **show ip interfaces brief** command at the end of the output shows Router R2's interfaces and IP addresses again per Example 18-7 and Figure 18-15.

The **ssh -l** *username host* command in Example 18-8 follows the same basic ideas as the **telnet** *host* command, but with an SSH client. The **-l** flag means that the next parameter is the login username. In this case, the user begins logged in to Router R1 and then uses the **ssh -l wendell 10.1.2.2** command to SSH to Router R2. R2 expects a username/password of wendell/odom, with wendell supplied in the command and odom supplied when R2 prompts the user.

Example 18-8 *SSH Client from R1 to R2 to View Interface Status on R2*

```
R1# ssh -l wendell 10.1.2.2

Password:

R2>
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  administratively down down
GigabitEthernet0/1         10.1.3.2        YES manual up                    up
GigabitEthernet0/2         10.1.2.2        YES manual up                    up
GigabitEthernet0/3         unassigned      YES unset  administratively down down
```

When you have finished using the other router, you can log out from your Telnet or SSH connection using the **exit** or **quit** command.

Finally, note that IOS supports a mechanism to use hotkeys to move between multiple Telnet or SSH sessions from the CLI. Basically, starting at one router, you could telnet or SSH to a router, do some commands, and instead of using the exit command to end your connection, you could keep the connection open while still moving back to the command prompt of the original router. For instance, if starting at Router R1, you could Telnet to R2, R3, and R4, suspending but not exiting those Telnet connections. Then you could easily move between the sessions to issue new commands with a few keystrokes.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 18-1 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 18-1 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Watch video		Website

Review All the Key Topics

Key Topic

Table 18-2 Key Topics for Chapter 18

Key Topic Element	Description	Page Number
Figure 18-5	ARP tables on Layer 3 hosts, with MAC address tables on Layer 2 switch	423
Figure 18-6	How extended ping in IOS performs a better test of the reverse route	424
Figure 18-7	Why a standard ping over a LAN does not exercise a host's default router logic	425
List	Network layer problems that could cause a ping to fail between a router and host on the same LAN subnet	426
List	Testing a host's default router setting using extended ping	426
List	Comparisons between the **ping** and **traceroute** commands	428

Key Terms You Should Know

ping, traceroute, ICMP echo request, ICMP echo reply, extended ping, forward route, reverse route, DNS

Part V Review

Keep track of your part review progress with the checklist in Table P5-1. Details on each task follow the table.

Table P5-1 Part V Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Review Videos		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book, using the PTP software.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson ICND1 or CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book, as found at http://blog.certskills.com. Then navigate to the Hands-on Config labs.

Other: If using other lab tools, here are a few suggestions: Make sure to experiment heavily with IPv4 addressing, static routing, and Layer 3 switching. In each case, test all your routes using **ping** and **traceroute**.

Watch Videos

Chapters 15, 17, and 18 each list a video to be found on the companion website, on topics ranging from how to use the router CLI, how to configure ROAS, and how to troubleshoot using Extended **ping**.

Part IV began the story in this book about IP Version 4 (IPv4) addressing. Part V continued that story with how to implement addressing in Cisco routers, along with a variety of methods to route packets between local interfaces. But those topics delayed the discussion of one of the most important topics in TCP/IP, namely IP routing protocols.

Routers use IP routing protocols to learn about the subnets in an internetwork, choose the current best routes to reach each subnet, and to add those routes to each router's IP routing table. Cisco chose to include one and only one IP routing protocol in the CCNA 200-301 exam: the Open Shortest Path First (OSPF) routing protocol. This entire part focuses on OSPF as an example of how routing protocols work.

Part VI

OSPF

CHAPTER 19

Understanding OSPF Concepts

This chapter covers the following exam topics:

3.0 IP Connectivity

3.2 Determine how a router makes a forwarding decision by default

3.2.b Administrative distance

3.2.c Routing protocol metric

3.4 Configure and verify single area OSPFv2

3.4.a Neighbor adjacencies

3.4.b Point-to-point

3.4.c Broadcast (DR/BR selection)

3.4.d Router ID

This chapter takes a long look at Open Shortest Path First Version 2 (OSPFv2) concepts. OSPF runs on each router, sending and receiving OSPF messages with neighboring (nearby) routers. These messages give OSPF the means to exchange data about the network and to learn and add IP Version 4 (IPv4) routes to the IPv4 routing table on each router.

Most enterprises over the last 25 years have used either OSPF or the Enhanced Interior Gateway Routing Protocol (EIGRP) for their primary IPv4 routing protocol. For perspective, both OSPF and EIGRP have been part of CCNA throughout most of its 20+ year history. For the CCNA 200-301 exam blueprint, Cisco has included OSPFv2 as the only IPv4 routing protocol. (Note that Cisco does include EIGRP in the CCNP Enterprise certification.)

This chapter breaks the content into three major sections. The first section sets the context about routing protocols in general, defining interior and exterior routing protocols and basic routing protocol features and terms. The second major section presents the nuts and bolts of how OSPFv2 works, using OSPF neighbor relationships, database exchange, and then route calculation. The third section wraps up the discussion by looking at OSPF areas and LSAs.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 19-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Comparing Dynamic Routing Protocol Features	1–3
OSPF Concepts and Operation	4, 5
OSPF Areas and LSAs	6

1. Which of the following routing protocols is considered to use link-state logic?
   a. RIPv1
   b. RIPv2
   c. EIGRP
   d. OSPF
2. Which of the following routing protocols use a metric that is, by default, at least partially affected by link bandwidth? (Choose two answers.)
   a. RIPv1
   b. RIPv2
   c. EIGRP
   d. OSPF
3. Which of the following interior routing protocols support VLSM? (Choose three answers.)
   a. RIPv1
   b. RIPv2
   c. EIGRP
   d. OSPF
4. Two routers using OSPFv2 have become neighbors and exchanged all LSAs. As a result, Router R1 now lists some OSPF-learned routes in its routing table. Which of the following best describes how R1 uses those recently learned LSAs to choose which IP routes to add to its IP routing table?
   a. Each LSA lists a route to be copied to the routing table.
   b. Some LSAs list a route that can be copied to the routing table.
   c. Run some SPF math against the LSAs to calculate the routes.
   d. R1 does not use the LSAs at all when choosing what routes to add.

5. Which of the following OSPF neighbor states is expected when the exchange of topology information is complete between two OSPF neighbors?
   a. 2-way
   b. Full
   c. Up/up
   d. Final

6. A company has a small/medium-sized network with 15 routers and 40 subnets and uses OSPFv2. Which of the following is considered an advantage of using a single-area design as opposed to a multiarea design?
   a. It reduces the processing overhead on most routers.
   b. Status changes to one link may not require SPF to run on all other routers.
   c. It allows for simpler planning and operations.
   d. It allows for route summarization, reducing the size of IP routing tables.

Foundation Topics

Comparing Dynamic Routing Protocol Features

Routers add IP routes to their routing tables using three methods: connected routes, static routes, and routes learned by using dynamic routing protocols. Before we get too far into the discussion, however, it is important to define a few related terms and clear up any misconceptions about the terms *routing protocol*, *routed protocol*, and *routable protocol*. The concepts behind these terms are not that difficult, but because the terms are so similar, and because many documents pay poor attention to when each of these terms is used, they can be a bit confusing. These terms are generally defined as follows:

- **Routing protocol:** A set of messages, rules, and algorithms used by routers for the overall purpose of learning routes. This process includes the exchange and analysis of routing information. Each router chooses the best route to each subnet (path selection) and finally places those best routes in its IP routing table. Examples include RIP, EIGRP, OSPF, and BGP.
- **Routed protocol and routable protocol:** Both terms refer to a protocol that defines a packet structure and logical addressing, allowing routers to forward or route the packets. Routers forward packets defined by routed and routable protocols. Examples include IP Version 4 (IPv4) and IP Version 6 (IPv6).

NOTE The term *path selection* sometimes refers to part of the job of a routing protocol, in which the routing protocol chooses the best route.

Even though routing protocols (such as OSPF) are different from routed protocols (such as IP), they do work together very closely. The routing process forwards IP packets, but if a router does not have any routes in its IP routing table that match a packet's destination address, the router discards the packet. Routers need routing protocols so that the routers

can learn all the possible routes and add them to the routing table so that the routing process can forward (route) routable protocols such as IP.

Routing Protocol Functions

Cisco IOS software supports several IP routing protocols, performing the same general functions:

Key Topic

1. Learn routing information about IP subnets from neighboring routers.
2. Advertise routing information about IP subnets to neighboring routers.
3. If more than one possible route exists to reach one subnet, pick the best route based on a metric.
4. If the network topology changes—for example, a link fails—react by advertising that some routes have failed and pick a new currently best route. (This process is called convergence.)

NOTE A neighboring router connects to the same link as another router, such as the same WAN link or the same Ethernet LAN.

Figure 19-1 shows an example of three of the four functions in the list. Router R1, in the lower left of the figure, must make a decision about the best route to reach the subnet connected off router R2, on the bottom right of the figure. Following the steps in the figure:

Step 1. R2 advertises a route to the lower right subnet—172.16.3.0/24—to both router R1 and R3.

Step 2. After R3 learns about the route to 172.16.3.0/24 from R2, R3 advertises that route to R1.

Step 3. R1 must make a decision about the two routes it learned about for reaching subnet 172.16.3.0/24—one with metric 1 from R2 and one with metric 2 from R3. R1 chooses the lower metric route through R2 (function 3).

The other routing protocol function, *convergence*, occurs when the topology changes—that is, when either a router or link fails or comes back up again. When something changes, the best routes available in the network can change. Convergence simply refers to the process by which all the routers collectively realize something has changed, advertise the information about the changes to all the other routers, and all the routers then choose the currently best routes for each subnet. The ability to converge quickly, without causing loops, is one of the most important considerations when choosing which IP routing protocol to use.

In Figure 19-1, convergence might occur if the link between R1 and R2 failed. In that case, R1 should stop using its old route for subnet 172.16.3.0/24 (directly through R2) and begin sending packets to R3.

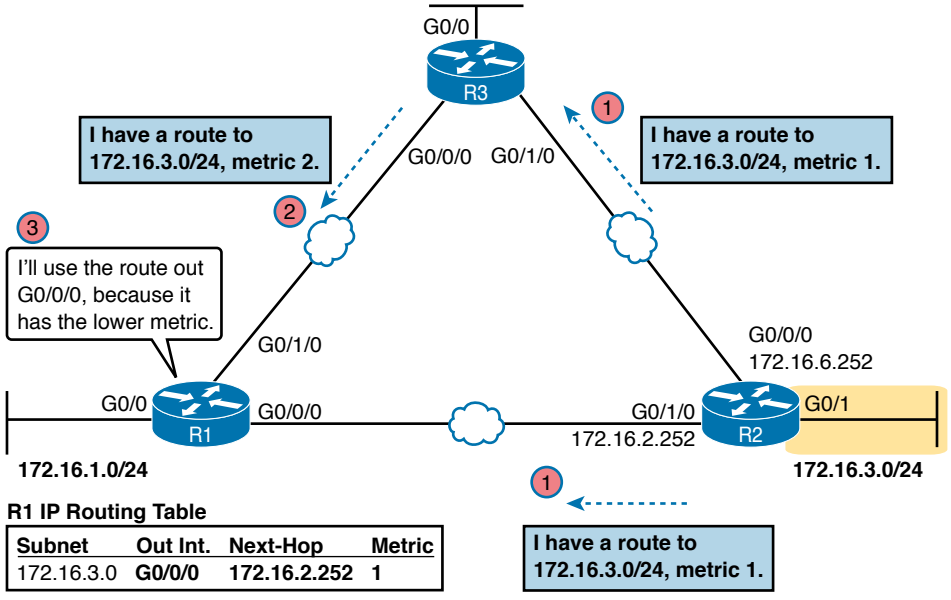


Figure 19-1 *Three of the Four Basic Functions of Routing Protocols*

Interior and Exterior Routing Protocols

IP routing protocols fall into one of two major categories: *interior gateway protocols* (IGP) or *exterior gateway protocols* (EGP). The definitions of each are as follows:

Key Topic

- **IGP:** A routing protocol that was designed and intended for use inside a single autonomous system (AS)
- **EGP:** A routing protocol that was designed and intended for use between different autonomous systems

NOTE The terms *IGP* and *EGP* include the word *gateway* because routers used to be called gateways.

These definitions use another new term: *autonomous system* (AS). An AS is a network under the administrative control of a single organization. For example, a network created and paid for by a single company is probably a single AS, and a network created by a single school system is probably a single AS. Other examples include large divisions of a state or national government, where different government agencies might be able to build their own networks. Each ISP is also typically a single different AS.

Some routing protocols work best inside a single AS by design, so these routing protocols are called IGPs. Conversely, routing protocols designed to exchange routes between routers

---

Answers to the "Do I Know This Already?" quiz:

1 D **2** C, D **3** B, C, D **4** C **5** B **6** C

in different autonomous systems are called EGPs. Today, Border Gateway Protocol (BGP) is the only EGP used.

Each AS can be assigned a number called (unsurprisingly) an *AS number* (ASN). Like public IP addresses, the Internet Assigned Numbers Authority (IANA, www.iana.org) controls the worldwide rights to assigning ASNs. It delegates that authority to other organizations around the world, typically to the same organizations that assign public IP addresses. For example, in North America, the American Registry for Internet Numbers (ARIN, www.arin.net) assigns public IP address ranges and ASNs.

Figure 19-2 shows a small view of the worldwide Internet. The figure shows two enterprises and three ISPs using IGPs (OSPF and EIGRP) inside their own networks and with BGP being used between the ASNs.

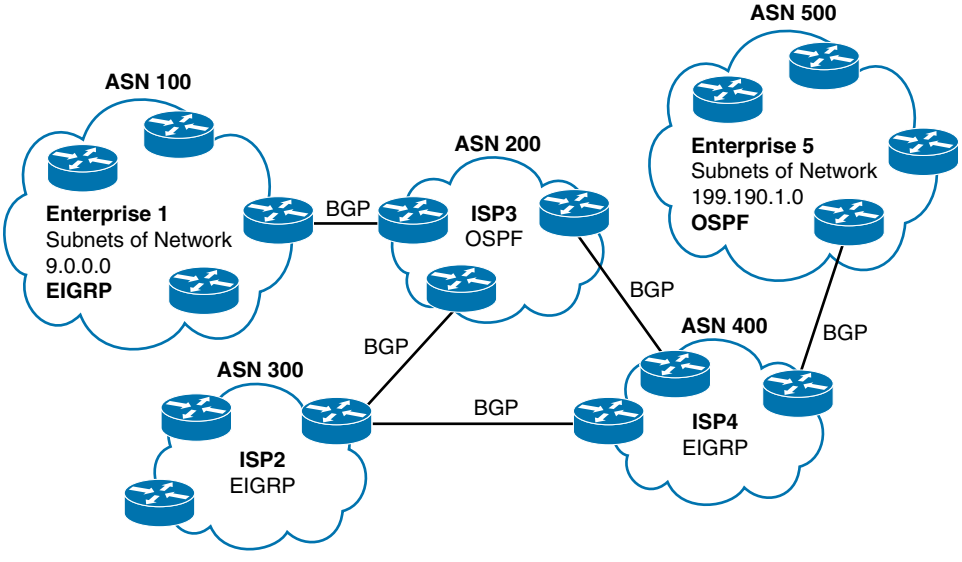


Figure 19-2 *Comparing Locations for Using IGPs and EGPs*

Comparing IGPs

Organizations have several options when choosing an IGP for their enterprise network, but most companies today use either OSPF or EIGRP. This book discusses OSPFv2, with the CCNP Enterprise certification adding EIGRP. Before getting into detail on these two protocols, the next section first discusses some of the main goals of every IGP, comparing OSPF, EIGRP, plus a few other IPv4 routing protocols.

IGP Routing Protocol Algorithms

A routing protocol's underlying algorithm determines how the routing protocol does its job. The term *routing protocol algorithm* simply refers to the logic and processes used by different routing protocols to solve the problem of learning all routes, choosing the best

route to each subnet, and converging in reaction to changes in the internetwork. Three main branches of routing protocol algorithms exist for IGP routing protocols:

Key Topic

- Distance vector (sometimes called Bellman-Ford after its creators)
- Advanced distance vector (sometimes called "balanced hybrid")
- Link-state

Historically speaking, distance vector protocols were invented first, mainly in the early 1980s. Routing Information Protocol (RIP) was the first popularly used IP distance vector protocol, with the Cisco-proprietary Interior Gateway Routing Protocol (IGRP) being introduced a little later.

By the early 1990s, distance vector protocols' somewhat slow convergence and potential for routing loops drove the development of new alternative routing protocols that used new algorithms. Link-state protocols—in particular, Open Shortest Path First (OSPF) and Integrated Intermediate System to Intermediate System (IS-IS)—solved the main issues. They also came with a price: they required extra CPU and memory on routers, with more planning required from the network engineers.

NOTE All references to OSPF in this chapter refer to OSPFv2 unless otherwise stated.

Around the same time as the introduction of OSPF, Cisco created a proprietary routing protocol called Enhanced Interior Gateway Routing Protocol (EIGRP), which used some features of the earlier IGRP protocol. EIGRP solved the same problems as did link-state routing protocols, but EIGRP required less planning when implementing the network. As time went on, EIGRP was classified as a unique type of routing protocol. However, it used more distance vector features than link-state, so it is more commonly classified as an advanced distance vector protocol.

Metrics

Routing protocols choose the best route to reach a subnet by choosing the route with the lowest metric. For example, RIP uses a counter of the number of routers (hops) between a router and the destination subnet, as shown in the example of Figure 19-1. OSPF totals the cost associated with each interface in the end-to-end route, with the cost based on link bandwidth. Table 19-2 lists the most common IP routing protocols and some details about the metric in each case.

Key Topic

Table 19-2 IP IGP Metrics

IGP	Metric	Description
RIPv2	Hop count	The number of routers (hops) between a router and the destination subnet
OSPF	Cost	The sum of all interface cost settings for all links in a route, with the cost defaulting to be based on interface bandwidth
EIGRP	Calculation based on bandwidth and delay	Calculated based on the route's slowest link and the cumulative delay associated with each interface in the route

A brief comparison of the metric used by the older RIP versus the metric used by OSPF shows some insight into why OSPF and EIGRP surpassed RIP. Figure 19-3 shows an example

in which Router B has two possible routes to subnet 10.1.1.0 on the left side of the network: a shorter route over a very slow serial link at 1544 Kbps, or a longer route over two Gigabit Ethernet WAN links.

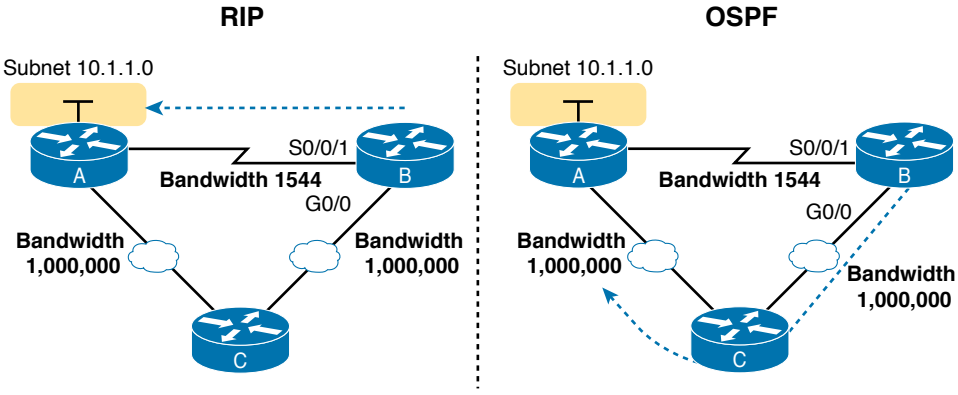


Figure 19-3 *RIP and OSPF Metrics Compared*

The left side of the figure shows the results of RIP in this network. Using hop count, Router B learns of a one-hop route directly to Router A through B's S0/0/1 interface. B also learns of a two-hop route through Router C, through B's G0/0 interface. Router B chooses the lower hop count route, which happens to go over the slow-speed serial link.

The right side of the figure shows the better choice made by OSPF based on its better metric. To cause OSPF to make the right choice, the engineer could use default settings based on the correct interface bandwidth to match the actual link speeds, thereby allowing OSPF to choose the faster route. (The **bandwidth** interface subcommand does not change the actual physical speed of the interface. It just tells IOS what speed to assume the interface is using.)

Other IGP Comparisons

Routing protocols can be compared based on many features, some of which matter to the current CCNA exam, whereas some do not. Table 19-3 introduces a few more points and lists the comparison points mentioned in this book for easier study, with a few supporting comments following the table.

Table 19-3 Interior IP Routing Protocols Compared

Feature	RIPv2	EIGRP	OSPF
Classless/sends mask in updates/supports VLSM	Yes	Yes	Yes
Algorithm (DV, advanced DV, LS)	DV	Advanced DV	LS
Supports manual summarization	Yes	Yes	Yes
Cisco-proprietary	No	Yes[1]	No
Routing updates are sent to a multicast IP address	Yes	Yes	Yes
Convergence	Slow	Fast	Fast

[1] Although Cisco created EIGRP and has kept it as a proprietary protocol for many years, Cisco chose to publish EIGRP as an informational RFC in 2013. This allows other vendors to implement EIGRP, while Cisco retains the rights to the protocol.

Regarding the top row of the table, routing protocols can be considered to be a classless routing protocol or a classful routing protocol. Classless routing protocols support variable-length subnet masks (VLSM) as well as manual route summarization by sending routing protocol messages that include the subnet masks in the message. The older RIPv1 and IGRP routing protocols—both classful routing protocols—do not.

Also, note that the older routing protocols (RIPv1, IGRP) sent routing protocol messages as IP broadcast addresses, while the newer routing protocols in the table all use IP multicast destination addresses. The use of multicasts makes the protocol more efficient and causes less overhead and fewer issues with the devices in the subnet that are not running the routing protocol.

Administrative Distance

Many companies and organizations use a single routing protocol. However, in some cases, a company needs to use multiple routing protocols. For example, if two companies connect their networks so that they can exchange information, they need to exchange some routing information. If one company uses OSPF and the other uses EIGRP on at least one router, both OSPF and EIGRP must be used. Then that router can take routes learned by OSPF and advertise them into EIGRP, and vice versa, through a process called *route redistribution*.

Depending on the network topology, the two routing protocols might learn routes to the same subnets. When a single routing protocol learns multiple routes to the same subnet, the metric tells it which route is best. However, when two different routing protocols learn routes to the same subnet, because each routing protocol's metric is based on different information, IOS cannot compare the metrics. For example, OSPF might learn a route to subnet 10.1.1.0 with metric 101, and EIGRP might learn a route to 10.1.1.0 with metric 2,195,416, but the EIGRP-learned route might be the better route—or it might not. There is simply no basis for comparison between the two metrics.

When IOS must choose between routes learned using different routing protocols, IOS uses a concept called *administrative distance*. Administrative distance is a number that denotes how believable an entire routing protocol is on a single router. The lower the number, the better, or more believable, the routing protocol. For example, RIP has a default administrative distance of 120, OSPF uses a default of 110, and EIGRP defaults to 90. When using OSPF and EIGRP, the router will believe the EIGRP route instead of the OSPF route (at least by default). The administrative distance values are configured on a single router and are not exchanged with other routers. Table 19-4 lists the various sources of routing information, along with the default administrative distances.

Table 19-4 Default Administrative Distances

Route Type	Administrative Distance
Connected	0
Static	1
BGP (external routes [eBGP])	20
EIGRP (internal routes)	90
IGRP	100
OSPF	110

Route Type	Administrative Distance
IS-IS	115
RIP	120
EIGRP (external routes)	170
BGP (internal routes [iBGP])	200
DHCP default route	254
Unusable	255

NOTE The **show ip route** command lists each route's administrative distance as the first of the two numbers inside the brackets. The second number in brackets is the metric.

The table shows the default administrative distance values, but IOS can be configured to change the administrative distance of a particular routing protocol, a particular route, or even a static route. For example, the command **ip route 10.1.3.0 255.255.255.0 10.1.130.253** defines a static route with a default administrative distance of 1, but the command **ip route 10.1.3.0 255.255.255.0 10.1.130.253 210** defines the same static route with an administrative distance of 210. So, you can actually create a static route that is only used when the routing protocol does not find a route, just by giving the static route a higher administrative distance.

OSPF Concepts and Operation

Routing protocols basically exchange information so routers can learn routes. The routers learn information about subnets, routes to those subnets, and metric information about how good each route is compared to others. The routing protocol can then choose the currently best route to each subnet, building the IP routing table.

Link-state protocols like OSPF take a little different approach to the particulars of what information they exchange and what the routers do with that information once learned. This next (second) major section narrows the focus to only link-state protocols, specifically OSPFv2.

This section begins with an overview of what OSPF does by exchanging data about the network in data structures called *link-state advertisements* (LSA). Then the discussion backs up a bit to provide more details about each of three fundamental parts of how OSPF operates: how OSPF routers use neighbor relationships, how routers exchange LSAs with neighbors, and then how routers calculate the best routes once they learn all the LSAs.

OSPF Overview

Link-state protocols build IP routes with a couple of major steps. First, the routers together build a lot of information about the network: routers, links, IP addresses, status information, and so on. Then the routers flood the information, so all routers know the same information. At that point, each router can calculate routes to all subnets, but from each router's own perspective.

Topology Information and LSAs

Routers using link-state routing protocols need to collectively advertise practically every detail about the internetwork to all the other routers. At the end of the process of *flooding* the information to all routers, every router in the internetwork has the exact same information about the internetwork. Flooding a lot of detailed information to every router sounds like a lot of work, and relative to distance vector routing protocols, it is.

Open Shortest Path First (OSPF), the most popular link-state IP routing protocol, organizes topology information using LSAs and the link-state database (LSDB). Figure 19-4 represents the ideas. Each LSA is a data structure with some specific information about the network topology; the LSDB is simply the collection of all the LSAs known to a router.

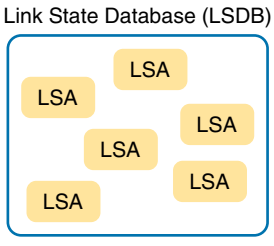


Figure 19-4 *LSA and LSDB Relationship*

Figure 19-5 shows the general idea of the flooding process, with R8 creating and flooding its *router LSA*. The router LSA for Router R8 describes the router itself, including the existence of subnet 172.16.3.0/24, as seen on the right side of the figure. (Note that Figure 19-5 actually shows only a subset of the information in R8's router LSA.)

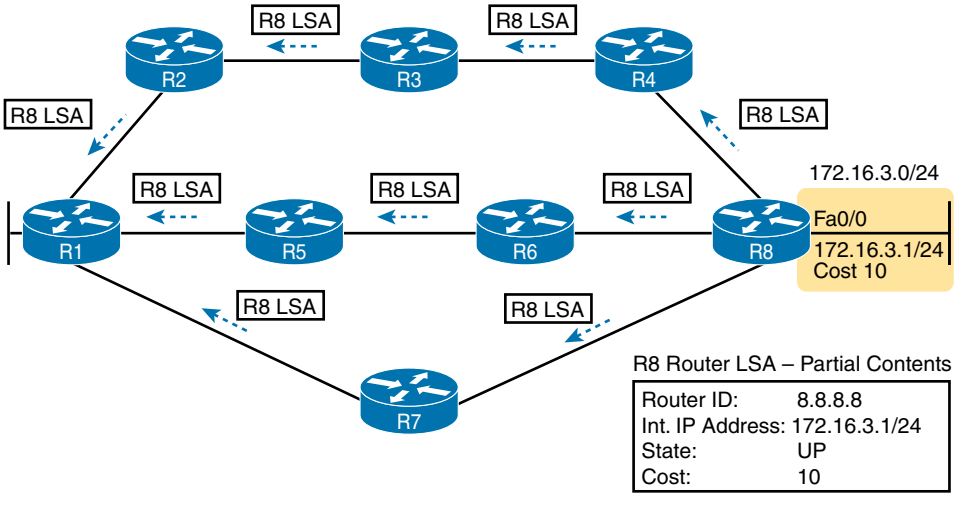


Figure 19-5 *Flooding LSAs Using a Link-State Routing Protocol*

Figure 19-5 shows the rather basic flooding process, with R8 sending the original LSA for itself, and the other routers flooding the LSA by forwarding it until every router has a copy. The flooding process causes every router to learn the contents of the LSA while preventing

the LSA from being flooded around in circles. Basically, before sending an LSA to yet another neighbor, routers communicate, asking "Do you already have this LSA?," and then sending the LSA to the next neighbor only if the neighbor has not yet learned about the LSA.

Once flooded, routers do occasionally reflood each LSA. Routers reflood an LSA when some information changes (for example, when a link goes up or comes down). They also reflood each LSA based on each LSA's separate aging timer (default 30 minutes).

Applying Dijkstra SPF Math to Find the Best Routes

The link-state flooding process results in every router having an identical copy of the LSDB in memory, but the flooding process alone does not cause a router to learn what routes to add to the IP routing table. Although incredibly detailed and useful, the information in the LSDB does not explicitly state each router's best route to reach a destination.

To build routes, link-state routers have to do some math. Thankfully, you and I do not have to know the math! However, all link-state protocols use a type of math algorithm, called the Dijkstra Shortest Path First (SPF) algorithm, to process the LSDB. That algorithm analyzes (with math) the LSDB and builds the routes that the local router should add to the IP routing table—routes that list a subnet number and mask, an outgoing interface, and a next-hop router IP address.

Now that you have the big ideas down, the next several topics walk through the three main phases of how OSPF routers accomplish the work of exchanging LSAs and calculating routes. Those three phases are

Becoming neighbors: A relationship between two routers that connect to the same data link, created so that the neighboring routers have a means to exchange their LSDBs.

Exchanging databases: The process of sending LSAs to neighbors so that all routers learn the same LSAs.

Adding the best routes: The process of each router independently running SPF, on their local copy of the LSDB, calculating the best routes, and adding those to the IPv4 routing table.

Becoming OSPF Neighbors

Of everything you learn about OSPF in this chapter, OSPF neighbor concepts have the most to do with how you will configure and troubleshoot OSPF in Cisco routers. You configure OSPF to cause routers to run OSPF and become neighbors with other routers. Once that happens, OSPF does the rest of the work to exchange LSAs and calculate routers in the background, with no additional configuration required. This section discusses the fundamental concepts of OSPF neighbors.

The Basics of OSPF Neighbors

OSPF neighbors are routers that both use OSPF and both sit on the same data link. Two routers can become OSPF neighbors if connected to the same VLAN, or same serial link, or same Ethernet WAN link.

Two routers need to do more than simply exist on the same link to become OSPF neighbors; they must send OSPF messages and agree to become OSPF neighbors. To do so, the routers

send OSPF Hello messages, introducing themselves to the potential neighbor. Assuming the two potential neighbors have compatible OSPF parameters, the two form an OSPF neighbor relationship, and would be displayed in the output of the **show ip ospf neighbor** command.

The OSPF neighbor relationship also lets OSPF know when a neighbor might not be a good option for routing packets right now. Imagine R1 and R2 form a neighbor relationship, learn LSAs, and calculate routes that send packets through the other router. Months later, R1 notices that the neighbor relationship with R2 fails. That failed neighbor connection to R2 makes R1 react: R1 refloods LSAs impacted by the failed link, and R1 runs SPF to recalculate its own routes.

Finally, the OSPF neighbor model allows new routers to be dynamically discovered. That means new routers can be added to a network without requiring every router to be reconfigured. Instead, OSPF routers listen for OSPF Hello messages from new routers and react to those messages, attempting to become neighbors and exchange LSDBs.

Meeting Neighbors and Learning Their Router ID

The OSPF Hello process, by which new neighbor relationships are formed, works somewhat like when you move to a new house and meet your various neighbors. When you see each other outside, you might walk over, say hello, and learn each other's name. After talking a bit, you form a first impression, particularly as to whether you think you'll enjoy chatting with this neighbor occasionally, or whether you can just wave and not take the time to talk the next time you see him outside.

Similarly, with OSPF, the process starts with messages called OSPF *Hello* messages. The Hellos in turn list each router's *router ID* (RID), which serves as each router's unique name or identifier for OSPF. Finally, OSPF does several checks of the information in the Hello messages to ensure that the two routers should become neighbors.

OSPF RIDs are 32-bit numbers. As a result, most command output lists these as dotted-decimal numbers (DDN). By default, IOS chooses one of the router's interface IPv4 addresses to use as its OSPF RID. However, the OSPF RID can be directly configured, as covered in the section "Configuring the OSPF Router ID" in Chapter 20, "Implementing OSPF."

As soon as a router has chosen its OSPF RID and some interfaces come up, the router is ready to meet its OSPF neighbors. OSPF routers can become neighbors if they are connected to the same subnet. To discover other OSPF-speaking routers, a router sends multicast OSPF Hello packets to each interface and hopes to receive OSPF Hello packets from other routers connected to those interfaces. Figure 19-6 outlines the basic concept.

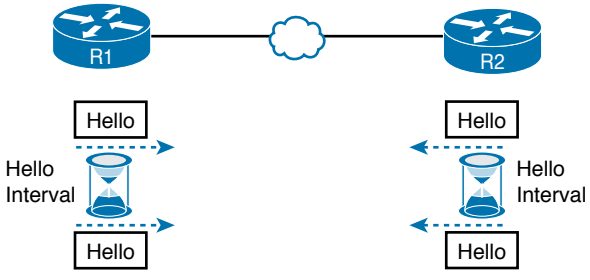


Figure 19-6 *OSPF Hello Packets*

Routers R1 and R2 both send Hello messages onto the link. They continue to send Hellos at a regular interval based on their Hello timer settings. The Hello messages themselves have the following features:

- The Hello message follows the IP packet header, with IP protocol type 89.
- Hello packets are sent to multicast IP address 224.0.0.5, a multicast IP address intended for all OSPF-speaking routers.
- OSPF routers listen for packets sent to IP multicast address 224.0.0.5, in part hoping to receive Hello packets and learn about new neighbors.

Taking a closer look, Figure 19-7 shows several of the neighbor states used by the early formation of an OSPF neighbor relationship. The figure shows the Hello messages in the center and the resulting neighbor states on the left and right edges of the figure. Each router keeps an OSPF state variable for how it views the neighbor.

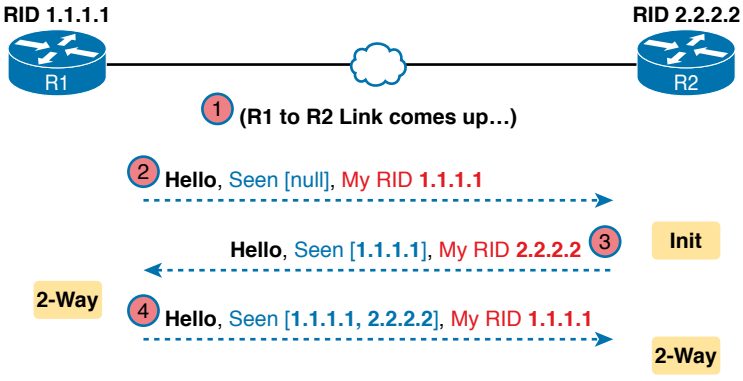


Figure 19-7 *Early Neighbor States*

Following the steps in the figure, the scenario begins with the link down, so the routers have no knowledge of each other as OSPF neighbors. As a result, they have no state (status) information about each other as neighbors, and they would not list each other in the output of the **show ip ospf neighbor** command. At Step 2, R1 sends the first Hello, so R2 learns of the existence of R1 as an OSPF router. At that point, R2 lists R1 as a neighbor, with an interim beginning state of init.

The process continues at Step 3, with R2 sending back a Hello. This message tells R1 that R2 exists, and it allows R1 to move through the init state and quickly to a 2-way state. At Step 4, R2 receives the next Hello from R1, and R2 can also move to a 2-way state.

The 2-way state is a particularly important OSPF state. At that point, the following major facts are true:

Key Topic

- The router received a Hello from the neighbor, with that router's own RID listed as being seen by the neighbor.
- The router has checked all the parameters in the Hello received from the neighbor, with no problems. The router is willing to become an OSPF neighbor.
- If both routers reach a 2-way state with each other, it means that both routers meet all OSPF configuration requirements to become neighbors. Effectively, at that point, they are neighbors and ready to exchange their LSDB with each other.

Exchanging the LSDB Between Neighbors

One purpose of forming OSPF neighbor relationships is to allow the two neighbors to exchange their databases. This next topic works through some of the details of OSPF database exchange.

Fully Exchanging LSAs with Neighbors

The OSPF neighbor state 2-way means that the router is available to exchange its LSDB with the neighbor. In other words, it is ready to begin a 2-way exchange of the LSDB. So, once two routers on a link reach the 2-way state, they can immediately move on to the process of database exchange.

The database exchange process can be quite involved, with several OSPF messages and several interim neighbor states. This chapter is more concerned with a few of the messages and the final state when database exchange has completed: the full state.

After two routers decide to exchange databases, they do not simply send the contents of the entire database. First, they tell each other a list of LSAs in their respective databases—not all the details of the LSAs, just a list. (Think of these lists as checklists.) Then each router can check which LSAs it already has and then ask the other router for only the LSAs that are not known yet.

For instance, R1 might send R2 a checklist that lists 10 LSAs (using an OSPF Database Description, or DD, packet). R2 then checks its LSDB and finds six of those 10 LSAs. So, R2 asks R1 (using a Link-State Request packet) to send the four additional LSAs.

Thankfully, most OSPFv2 work does not require detailed knowledge of these specific protocol steps. However, a few of the terms are used quite a bit and should be remembered. In particular, the OSPF messages that actually send the LSAs between neighbors are called Link-State Update (LSU) packets. That is, the LSU packet holds data structures called link-state advertisements (LSA). The LSAs are not packets, but rather data structures that sit inside the LSDB and describe the topology.

Figure 19-8 pulls some of these terms and processes together, with a general example. The story picks up the example shown in Figure 19-7, with Figure 19-8 showing an example of the database exchange process between Routers R1 and R2. The center shows the protocol messages, and the outer items show the neighbor states at different points in the process. Focus on two items in particular:

- The routers exchange the LSAs inside LSU packets.
- When finished, the routers reach a full state, meaning they have fully exchanged the contents of their LSDBs.

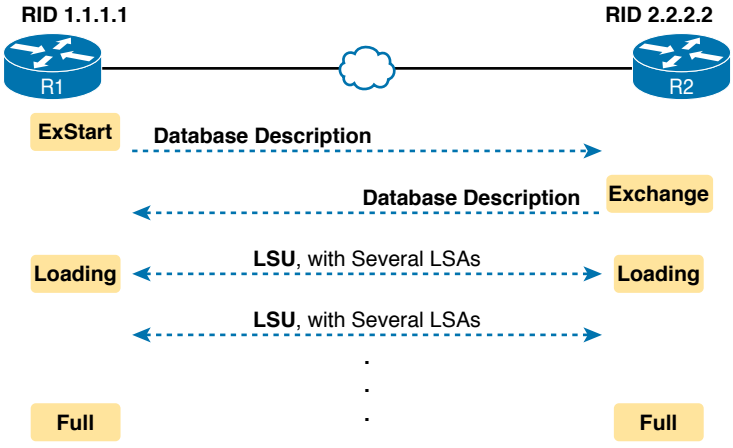


Figure 19-8 *Database Exchange Example, Ending in a Full State*

Maintaining Neighbors and the LSDB

Once two neighbors reach a full state, they have done all the initial work to exchange OSPF information between them. However, neighbors still have to do some small ongoing tasks to maintain the neighbor relationship.

First, routers monitor each neighbor relationship using Hello messages and two related timers: the *Hello Interval* and the *Dead Interval*. Routers send Hellos every Hello Interval to each neighbor. Each router expects to receive a Hello from each neighbor based on the Hello Interval, so if a neighbor is silent for the length of the Dead Interval (by default, four times as long as the Hello Interval), the loss of Hellos means that the neighbor has failed.

Next, routers must react when the topology changes as well, and neighbors play a key role in that process. When something changes, one or more routers change one or more LSAs. Then the routers must flood the changed LSAs to each neighbor so that the neighbor can change its LSDB.

For example, imagine a LAN switch loses power, so a router's G0/0 interface fails from up/up to down/down. That router updates an LSA that shows the router's G0/0 as being down. That router then sends the LSA to its neighbors, and that neighbor in turn sends it to its neighbors, until all routers again have an identical copy of the LSDB. Each router's LSDB now reflects the fact that the original router's G0/0 interface failed, so each router will then use SPF to recalculate any routes affected by the failed interface.

A third maintenance task done by neighbors is to reflood each LSA occasionally, even when the network is completely stable. By default, each router that creates an LSA also has the responsibility to reflood the LSA every 30 minutes (the default), even if no changes occur. (Note that each LSA has a separate timer, based on when the LSA was created, so there is no single big event where the network is overloaded with flooding LSAs.)

The following list summarizes these three maintenance tasks for easier review:

- Maintain neighbor state by sending Hello messages based on the Hello Interval and listening for Hellos before the Dead Interval expires
- Flood any changed LSAs to each neighbor
- Reflood unchanged LSAs as their lifetime expires (default 30 minutes)

Using Designated Routers on Ethernet Links

OSPF behaves differently on some types of interfaces based on a per-interface setting called the OSPF *network type*. On Ethernet links, OSPF defaults to use a network type of *broadcast*, which causes OSPF to elect one of the routers on the same subnet to act as the *designated router* (DR). The DR plays a key role in how the database exchange process works, with different rules than with point-to-point links.

To see how, consider the example that begins with Figure 19-9. The figure shows five OSPFv2 routers on the same Ethernet VLAN. These five OSPF routers elect one router to act as the DR and one router to be a backup DR (BDR). The figure shows A and B as DR and BDR, for no other reason than the Ethernet must have one of each.

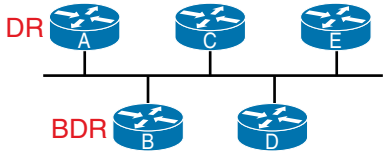


Figure 19-9 *Routers A and B Elected as DR and BDR*

The database exchange process on an Ethernet link does not happen between every pair of routers on the same VLAN/subnet. Instead, it happens between the DR and each of the other routers, with the DR making sure that all the other routers get a copy of each LSA. In other words, the database exchange happens over the flows shown in Figure 19-10.

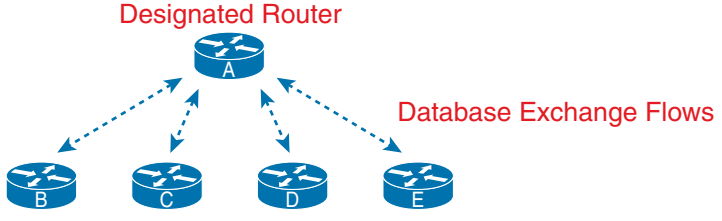


Figure 19-10 *Database Exchange to and from the DR on an Ethernet*

OSPF uses the BDR concept because the DR is so important to the database exchange process. The BDR watches the status of the DR and takes over for the DR if it fails. (When the DR fails, the BDR takes over, and then a new BDR is elected.)

The use of a DR/BDR, along with the use of multicast IP addresses, makes the exchange of OSPF LSDBs more efficient on networks that allow more than two routers on the same link. The DR can send a packet to all OSPF routers in the subnet by using multicast IP address 224.0.0.5. IANA reserves this address as the "All SPF Routers" multicast address just for this

purpose. For instance, in Figure 19-10, the DR can send one set of messages to all the OSPF routers rather than sending one message to each router.

Similarly, any OSPF router needing to send a message to the DR and also to the BDR (so it remains ready to take over for the DR) can send those messages to the "All SPF DRs" multicast address 224.0.0.6. So, instead of having to send one set of messages to the DR and another set to the BDR, an OSPF router can send one set of messages, making the exchange more efficient.

At this point, you might be getting a little tired of some of the theory, but finally, the theory actually shows something that you may see in **show** commands on a router. Because the DR and BDR both do full database exchange with all the other OSPF routers in the LAN, they reach a full state with all neighbors. However, routers that are neither a DR nor a BDR—called *DROthers* by OSPF—never reach a full state because they do not exchange LSDBs directly with each other. As a result, the **show ip ospf neighbor** command on these DROther routers lists some neighbors in a 2-way state, remaining in that state under normal operation.

For instance, with OSPF working normally on the Ethernet LAN in Figure 19-10, a **show ip ospf neighbor** command on router C (which is a DROther router) would show the following:

- Two neighbors (A and B, the DR and BDR, respectively) with a full state (called *fully adjacent neighbors*)
- Two neighbors (D and E, which are DROthers) with a 2-way state (called *neighbors*)

OSPF requires some terms to describe all neighbors versus the subset of all neighbors that reach the full state. First, all OSPF routers on the same link that reach the 2-way state—that is, they send Hello messages and the parameters match—are called *neighbors*. The subset of neighbors for which the neighbor relationship continues on and reaches the full state are called *adjacent neighbors*. Additionally, OSPFv2 RFC 2328 emphasizes the connection between the full state and the term *adjacent neighbor* by using the synonyms of *fully adjacent* and *fully adjacent neighbor*. Finally, while the terms so far refer to the neighbor, two other terms refer to the relationship: *neighbor relationship* refers to any OSPF neighbor relationship, while the term *adjacency* refers to neighbor relationships that reach a full state. Table 19-5 details the terms.

Key Topic

Table 19-5 Stable OSPF Neighbor States and Their Meanings

Neighbor State	Term for Neighbor	Term for Relationship
2-way	Neighbor	Neighbor Relationship
Full	Adjacent Neighbor Fully Adjacent Neighbor	Adjacency

Calculating the Best Routes with SPF

OSPF LSAs contain useful information, but they do not contain the specific information that a router needs to add to its IPv4 routing table. In other words, a router cannot just copy information from the LSDB into a route in the IPv4 routing table. The LSAs individually are more like pieces of a jigsaw puzzle. So, to know what routes to add to the routing table, each

router must do some SPF math to choose the best routes from that router's perspective. The router then adds each route to its routing table: a route with a subnet number and mask, an outgoing interface, and a next-hop router IP address.

Although engineers do not need to know the details of how SPF does the math, they do need to know how to predict which routes SPF will choose as the best route. The SPF algorithm calculates all the routes for a subnet—that is, all possible routes from the router to the destination subnet. If more than one route exists, the router compares the metrics, picking the best (lowest) metric route to add to the routing table. Although the SPF math can be complex, engineers with a network diagram, router status information, and simple addition can calculate the metric for each route, predicting what SPF will choose.

Once SPF has identified a route, OSPF calculates the metric for a route as follows:

Key Topic

The sum of the OSPF interface costs for all outgoing interfaces in the route.

Figure 19-11 shows an example with three possible routes from R1 to Subnet X (172.16.3.0/24) at the bottom of the figure.

Key Topic

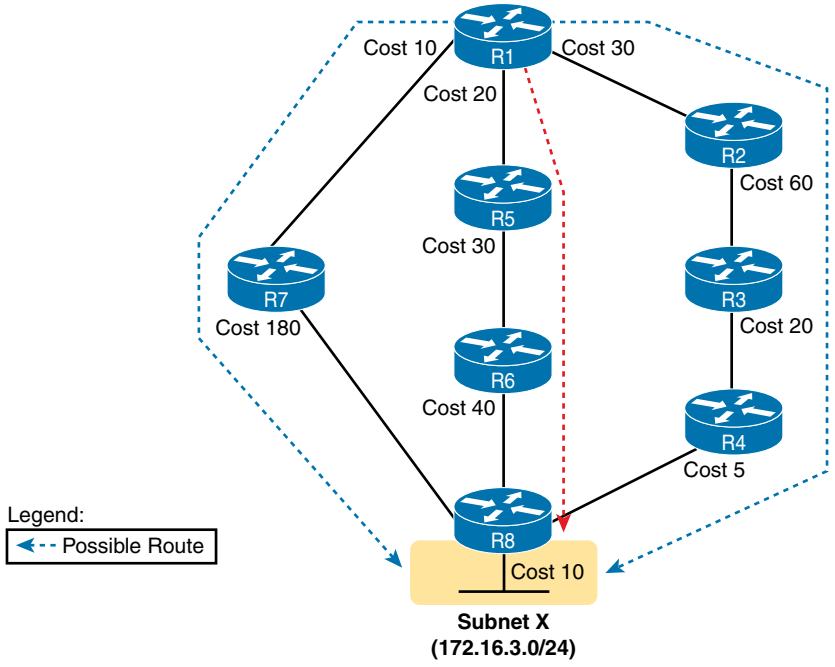


Figure 19-11 *SPF Tree to Find R1's Route to 172.16.3.0/24*

NOTE OSPF considers the costs of the outgoing interfaces (only) in each route. It does not add the cost for incoming interfaces in the route.

Table 19-6 lists the three routes shown in Figure 19-11, with their cumulative costs, showing that R1's best route to 172.16.3.0/24 starts by going through R5.

Table 19-6 Comparing R1's Three Alternatives for the Route to 172.16.3.0/24

Route	Location in Figure 19-11	Cumulative Cost
R1–R7–R8	Left	10 + 180 + 10 = 200
R1–R5–R6–R8	Middle	20 + 30 + 40 + 10 = 100
R1–R2–R3–R4–R8	Right	30 + 60 + 20 + 5 + 10 = 125

As a result of the SPF algorithm's analysis of the LSDB, R1 adds a route to subnet 172.16.3.0/24 to its routing table, with the next-hop router of R5.

In real OSPF networks, an engineer can do the same process by knowing the OSPF cost for each interface. Armed with a network diagram, the engineer can examine all routes, add the costs, and predict the metric for each route.

OSPF Areas and LSAs

OSPF can be used in some networks with very little thought about design issues. You just turn on OSPF in all the routers, put all interfaces into the same area (usually area 0), and it works! Figure 19-12 shows one such network example, with 11 routers and all interfaces in area 0.

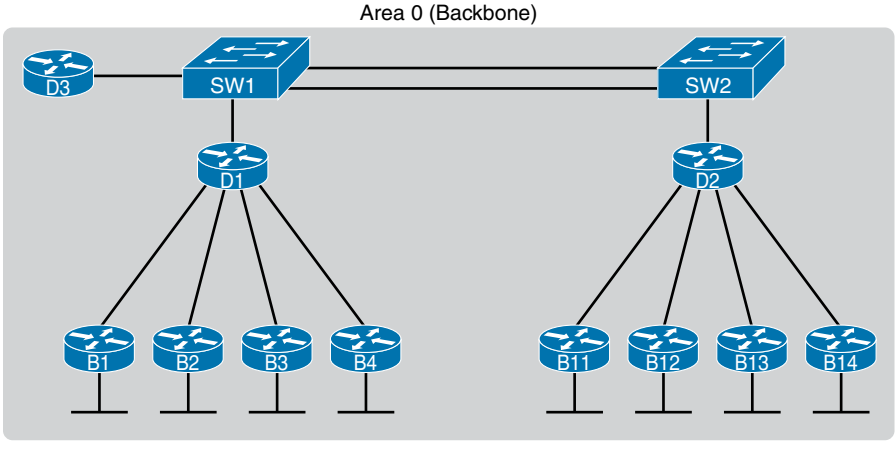


Figure 19-12 *Single-Area OSPF*

Larger OSPFv2 networks suffer with a single-area design. For instance, now imagine an enterprise network with 900 routers, rather than only 11, and several thousand subnets. As it turns out, the CPU time to run the SPF algorithm on all that topology data just takes time. As a result, OSPFv2 convergence time—the time required to react to changes in the network—can be slow. The routers might run low on RAM as well. Additional problems with a single area design include the following:

- A larger topology database requires more memory on each router.
- The SPF algorithm requires processing power that grows exponentially compared to the size of the topology database.
- A single interface status change anywhere in the internetwork (up to down, or down to up) forces *every router* to run SPF again!

The solution is to take the one large LSDB and break it into several smaller LSDBs by using OSPF areas. With areas, each link is placed into one area. SPF does its complicated math on the topology inside the area, and that area's topology only. For instance, an internetwork with 1000 routers and 2000 subnets, broken in 100 areas, would average 10 routers and 20 subnets per area. The SPF calculation on a router would have to only process topology about 10 routers and 20 links, rather than 1000 routers and 2000 links.

So, how large does a network have to be before OSPF needs to use areas? Well, there is no set answer because the behavior of the SPF process depends largely on CPU processing speed, the amount of RAM, the size of the LSDB, and so on. Generally, networks larger than a few dozen routers benefit from areas, and some documents over the years have listed 50 routers as the dividing line at which a network really should use multiple OSPF areas.

The next few pages look at how OSPF area design works, with more reasons as to why areas help make larger OSPF networks work better.

OSPF Areas

OSPF area design follows a couple of basic rules. To apply the rules, start with a clean drawing of the internetwork, with routers, and all interfaces. Then choose the area for each router interface, as follows:

- Put all interfaces connected to the same subnet inside the same area.
- An area should be contiguous.
- Some routers may be internal to an area, with all interfaces assigned to that single area.
- Some routers may be Area Border Routers (ABR) because some interfaces connect to the backbone area, and some connect to nonbackbone areas.
- All nonbackbone areas must have a path to reach the backbone area (area 0) by having at least one ABR connected to both the backbone area and the nonbackbone area.

Figure 19-13 shows one example. An engineer started with a network diagram that showed all 11 routers and their links. On the left, the engineer put four WAN links and the LANs connected to branch routers B1 through B4 into area 1. Similarly, he placed the links to branches B11 through B14 and their LANs in area 2. Both areas need a connection to the backbone area, area 0, so he put the LAN interfaces of D1 and D2 into area 0, along with D3, creating the backbone area.

The figure also shows a few important OSPF area design terms. Table 19-7 summarizes the meaning of these terms, plus some other related terms, but pay closest attention to the terms from the figure.

Key Topic

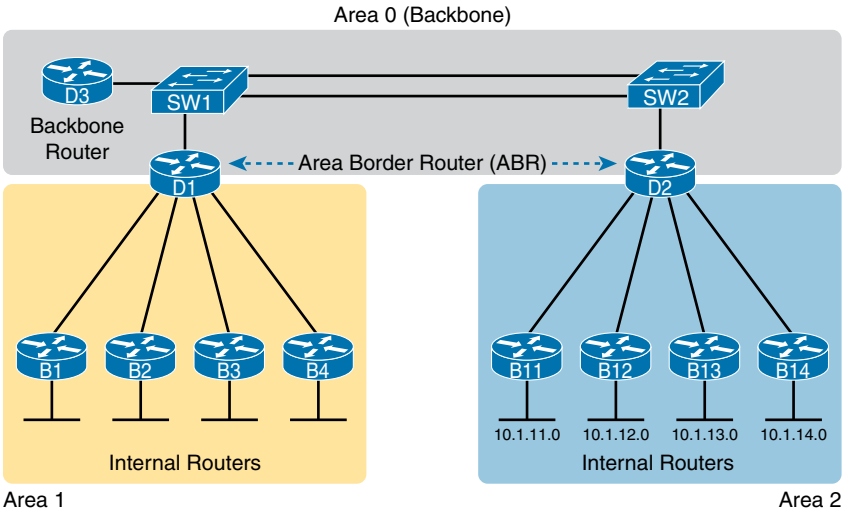


Figure 19-13 *Three-Area OSPF with D1 and D2 as ABRs*

Key Topic

Table 19-7 OSPF Design Terminology

Term	Description
Area Border Router (ABR)	An OSPF router with interfaces connected to the backbone area and to at least one other area
Backbone router	A router connected to the backbone area (includes ABRs)
Internal router	A router in one area (not the backbone area)
Area	A set of routers and links that shares the same detailed LSDB information, but not with routers in other areas, for better efficiency
Backbone area	A special OSPF area to which all other areas must connect—area 0
Intra-area route	A route to a subnet inside the same area as the router
Interarea route	A route to a subnet in an area of which the router is not a part

How Areas Reduce SPF Calculation Time

Figure 19-13 shows a sample area design and some terminology related to areas, but it does not show the power and benefit of the areas. To understand how areas reduce the work SPF has to do, you need to understand what changes about the LSDB inside an area, as a result of the area design.

SPF spends most of its processing time working through all the topology details, namely routers and the links that connect routers. Areas reduce SPF's workload because, for a given area, the LSDB lists only routers and links inside that area, as shown on the left side of Figure 19-14.

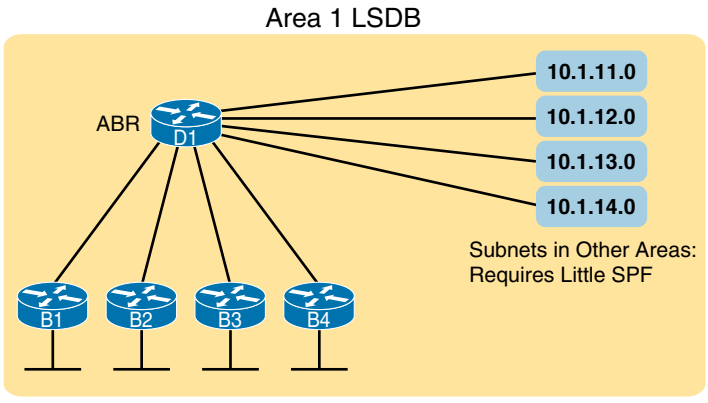


Figure 19-14 *Smaller Area 1 LSDB Concept*

While the LSDB has less topology information, it still has to have information about all subnets in all areas, so that each router can create IPv4 routes for all subnets. So, with an area design, OSPFv2 uses very brief summary information about the subnets in other areas. These summary LSAs do not include topology information about the other areas; however, each summary LSA *does* list a subnet ID and mask of a subnet in some other area. Summary LSAs do not require much SPF processing at all. Instead, these subnets all appear like subnets connected to the ABR (in Figure 19-14, ABR D1).

Using multiple areas improves OSPF operations in many ways for larger networks. The following list summarizes some of the key points arguing for the use of multiple areas in larger OSPF networks:

- Routers require fewer CPU cycles to process the smaller per-area LSDB with the SPF algorithm, reducing CPU overhead and improving convergence time.
- The smaller per-area LSDB requires less memory.
- Changes in the network (for example, links failing and recovering) require SPF calculations only on routers in the area where the link changed state, reducing the number of routers that must rerun SPF.
- Less information must be advertised between areas, reducing the bandwidth required to send LSAs.

(OSPFv2) Link-State Advertisements

Many people tend to get a little intimidated by OSPF LSAs when first learning about them. Commands that list a summary of the LSDB's contents, like the **show ip ospf database** command, actually list a lot of information. Commands that list the details of the LSDB can list overwhelming amounts of information, and those details appear to be in some kind of code, using lots of numbers. It can seem like a bit of a mess.

However, if you examine LSAs while thinking about OSPF areas and area design, some of the most common LSA types will make a lot more sense. For instance, think about the LSDB

in one area. The topology in one area includes routers and the links between the routers. As it turns out, OSPF defines the first two types of LSAs to define those exact details, as follows:

- One *router LSA* for each router in the area
- One *network LSA* for each network that has a DR plus one neighbor of the DR

Next, think about the subnets in the other areas. The ABR creates summary information about each subnet in one area to advertise into other areas—basically just the subnet IDs and masks—as a third type of LSA:

- One *summary* LSA for each subnet ID that exists in a different area

The next few pages discuss these three LSA types in a little more detail; Table 19-8 lists some information about all three for easier reference and study.

Table 19-8 The Three OSPFv2 LSA Types Seen with a Multiarea OSPF Design

LSA Name	LSA Type	Primary Purpose	Contents of LSA
Router	1	Describe a router	RID, interfaces, IP address/mask, current interface state (status)
Network	2	Describe a network that has a DR	DR and BDR IP addresses, subnet ID, mask
Summary	3	Describe a subnet in another area	Subnet ID, mask, RID of ABR that advertises the LSA

Router LSAs Build Most of the Intra-Area Topology

OSPF needs very detailed topology information inside each area. The routers inside area X need to know all the details about the topology inside area X. And the mechanism to give routers all these details is for the routers to create and flood router (Type 1) and network (Type 2) LSAs about the routers and links in the area.

Router LSAs, also known as Type 1 LSAs, describe the router in detail. Each lists a router's RID, its interfaces, its IPv4 addresses and masks, its interface state, and notes about what neighbors the router knows about via each of its interfaces.

To see a specific instance, first review Figure 19-15. It lists internetwork topology, with subnets listed. Because it's a small internetwork, the engineer chose a single-area design, with all interfaces in backbone area 0.

With the single-area design planned for this small internetwork, the LSDB will contain four router LSAs. Each router creates a router LSA for itself, with its own RID as the LSA identifier. The LSA lists that router's own interfaces, IP address/mask, with pointers to neighbors.

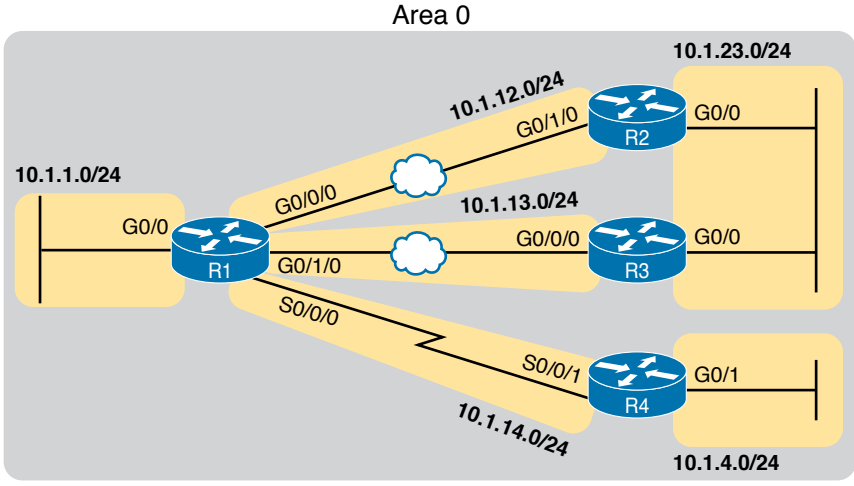


Figure 19-15 *Enterprise Network with Seven IPv4 Subnets*

Once all four routers have copies of all four router LSAs, SPF can mathematically analyze the LSAs to create a model. The model looks a lot like the concept drawing in Figure 19-16. Note that the drawing shows each router with an obvious RID value. Each router has pointers that represent each of its interfaces, and because the LSAs identify neighbors, SPF can figure out which interfaces connect to which other routers.

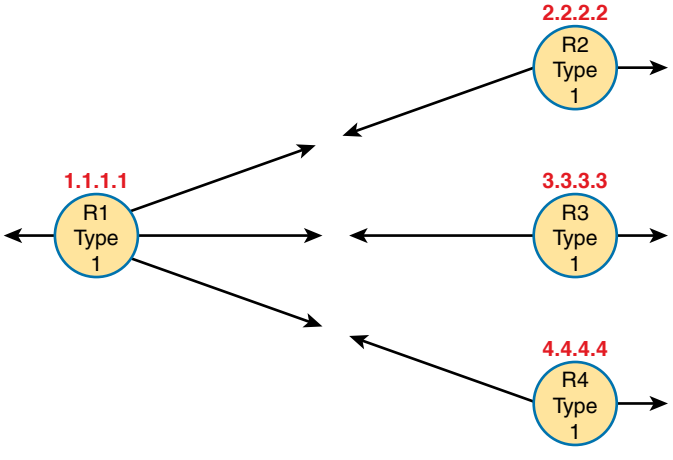


Figure 19-16 *Type 1 LSAs, Assuming a Single-Area Design*

Network LSAs Complete the Intra-Area Topology

Whereas router LSAs define most of the intra-area topology, network LSAs define the rest. As it turns out, when OSPF elects a DR on some subnet *and* that DR has at least one neighbor, OSPF treats that subnet as another node in its mathematical model of the network. To represent that network, the DR creates and floods a network (Type 2) LSA for that network (subnet).

For instance, back in Figure 19-15, one Ethernet LAN and two Ethernet WANs exist. The Ethernet LAN between R2 and R3 will elect a DR, and the two routers will become neighbors; so, whichever router is the DR will create a network LSA. Similarly, R1 and R2 connect with an Ethernet WAN, so the DR on that link will create a network LSA. Likewise, the DR on the Ethernet WAN link between R1 and R3 will also create a network LSA.

Figure 19-17 shows the completed version of the intra-area LSAs in area 0 with this design. Note that the router LSAs actually point to the network LSAs when they exist, which lets the SPF processes connect the pieces together.

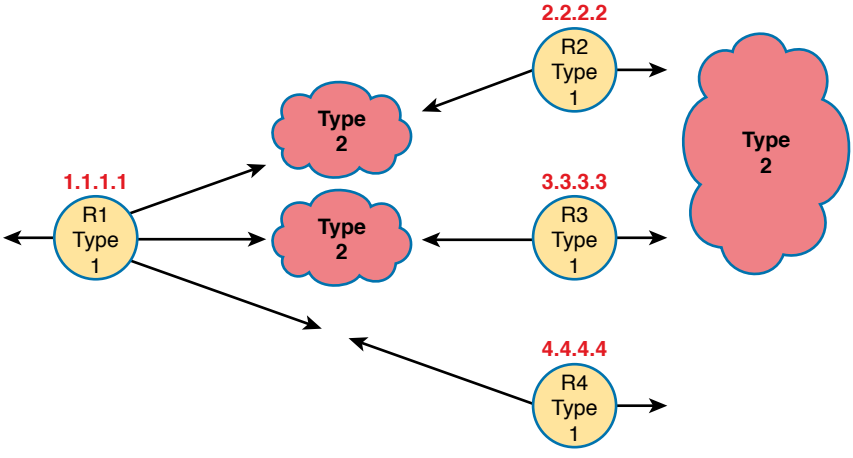


Figure 19-17 *Type 1 and Type 2 LSAs in Area 0, Assuming a Single-Area Design*

Finally, note that in this single-area design example no summary (Type 3) LSAs exist at all. These LSAs represent subnets in other areas, and there are no other areas. Given that the CCNA 200-301 exam topics refer specifically to single-area OSPF designs, this section stops at showing the details of the intra-area LSAs (Types 1 and 2).

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 19-9 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 19-9 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used:
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Key Topic

Table 19-10 Key Topics for Chapter 19

Key Topic Element	Description	Page Number
List	Functions of IP routing protocols	443
List	Definitions of IGP and EGP	444
List	Types of IGP routing protocols	446
Table 19-2	IGP metrics	446
List	Key facts about the OSPF 2-way state	453
Table 19-5	Key OSPF neighbor states	457
Item	Definition of how OSPF calculates the cost for a route	458
Figure 19-11	Example of calculating the cost for multiple competing routes	458
List	OSPF area design rules	460
Figure 19-13	Sample OSPF multiarea design with terminology	461
Table 19-7	OSPF design terms and definitions	461

Key Terms You Should Know

convergence, Shortest Path First (SPF) algorithm, distance vector, Interior Gateway Protocol (IGP), link-state, link-state advertisement (LSA), link-state database (LSDB), metric, 2-way state, full state, Area Border Router (ABR), designated router (DR), backup designated router (BDR), fully adjacent, Hello Interval, Dead Interval, link-state update, neighbor, router ID (RID), topology database, internal router, backbone area

This page intentionally left blank

CHAPTER 20

Implementing OSPF

This chapter covers the following exam topics:

3.0 IP Connectivity

3.2 Determine how a router makes a forwarding decision by default

3.2.b Administrative distance

3.2.c Routing protocol metric

3.4 Configure and verify single area OSPFv2

3.4.a Neighbor adjacencies

3.4.b Point-to-point

3.4.c Broadcast (DR/BR selection)

3.4.d Router ID

OSPFv2 requires only a few configuration commands if you rely on default settings. To use OSPF, all you need to do is enable OSPF on each interface you intend to use in the network, and OSPF uses messages to discover neighbors and learn routes through those neighbors. However, the complexity of OSPFv2 results in a large number of show commands, many of which reveal those default settings. So while you can make OSPFv2 work in a lab with all default settings, to become comfortable working with it, you need to know the most common optional features as well. This chapter begins that process.

The first major section of this chapter focuses on traditional OSPFv2 configuration using the **network** command, along with the large variety of associated show commands. This section teaches you how to make OSPFv2 operate with default settings and convince yourself that it really is working through use of those show commands.

The second major section shows an alternative configuration option called OSPF interface mode, in contrast with the traditional OSPF configuration shown in the first section of the chapter. This mode uses the **ip ospf** *process-id* **area** *area-number* configuration command instead of the **network** command.

The final section then moves on to discuss a variety of optional but popular configuration topics. The features include topics such as how to use passive interfaces, how to change OSPF costs (which influences the routes OSPF chooses), and how to create a default route advertised by OSPF.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 20-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Implementing Single-Area OSPFv2	1–3
OSPFv2 Interface Configuration	4
Additional OSPFv2 Features	5, 6

1. Which of the following **network** commands, following the command **router ospf 1**, tells this router to start using OSPF on interfaces whose IP addresses are 10.1.1.1, 10.1.100.1, and 10.1.120.1?
   a. **network 10.0.0.0 255.0.0.0 area 0**
   b. **network 10.0.0.0 0.255.255.255 area 0**
   c. **network 10.0.0.1 0.0.0.255 area 0**
   d. **network 10.0.0.1 0.0.255.255 area 0**
2. Which of the following **network** commands, following the command **router ospf 1**, tells this router to start using OSPF on interfaces whose IP addresses are 10.1.1.1, 10.1.100.1, and 10.1.120.1?
   a. **network 10.1.0.0 0.0.255.255 area 0**
   b. **network 10.0.0.0 0.255.255.0 area 0**
   c. **network 10.1.1.0 0.x.1x.0 area 0**
   d. **network 10.1.1.0 255.0.0.0 area 0**
   e. **network 10.0.0.0 255.0.0.0 area 0**
3. Which of the following commands list the OSPF neighbors off interface serial 0/0? (Choose two answers.)
   a. **show ip ospf neighbor**
   b. **show ip ospf interface brief**
   c. **show ip neighbor**
   d. **show ip interface**
   e. **show ip ospf neighbor serial 0/0**

4. An engineer migrates from a more traditional OSPFv2 configuration that uses **network** commands in OSPF configuration mode to instead use OSPFv2 interface configuration. Which of the following commands configures the area number assigned to an interface in this new configuration?
   a. The **area** command in interface configuration mode
   b. The **ip ospf** command in interface configuration mode
   c. The **router ospf** command in interface configuration mode
   d. The **network** command in interface configuration mode
5. Which of the following configuration settings on a router does not influence which IPv4 route a router chooses to add to its IPv4 routing table when using OSPFv2?
   a. **auto-cost reference-bandwidth**
   b. **delay**
   c. **bandwidth**
   d. **ip ospf cost**
6. OSPF interface configuration uses the **ip ospf** *process-id* **area** *area-number* configuration command. In which modes do you configure the following settings when using this command?
   a. The router ID is configured explicitly in router mode.
   b. The router ID is configured explicitly in interface mode.
   c. An interface's area number is configured in router mode.
   d. An interface's area number is configured in interface mode.

Foundation Topics

Implementing Single-Area OSPFv2

After an OSPF design has been chosen—a task that can be complex in larger IP internetworks—the configuration can be as simple as enabling OSPF on each router interface and placing that interface in the correct OSPF area. This first major section of the chapter focuses on the required configuration using the traditional OSPFv2 **network** command along with one optional configuration setting: how to set the OSPF router-id. Additionally, this section works through how to show the various lists and tables that confirm how OSPF is working.

For reference and study, the following list outlines the configuration steps covered in this first major section of the chapter:

Config Checklist

Step 1. Use the **router ospf** *process-id* global command to enter OSPF configuration mode for a particular OSPF process.

Step 2. (Optional) Configure the OSPF router ID by doing the following:

A. Use the **router-id** *id-value* router subcommand to define the router ID, or

B. Use the **interface loopback** *number* global command, along with an **ip address** *address mask* command, to configure an IP address on a loopback interface (chooses the highest IP address of all working loopbacks), or

C. Rely on an interface IP address (chooses the highest IP address of all working nonloopbacks).

Step 3. Use one or more **network** *ip-address wildcard-mask* **area** *area-id* router subcommands to enable OSPFv2 on any interfaces matched by the configured address and mask, enabling OSPF on the interface for the listed area.

Figure 20-1 shows the relationship between the OSPF configuration commands, with the idea that the configuration creates a routing process in one part of the configuration, and then indirectly enables OSPF on each interface. The configuration does not name the interfaces on which OSPF is enabled, instead requiring IOS to apply some logic by comparing the OSPF **network** command to the interface **ip address** commands. The upcoming example discusses more about this logic.

Key Topic

Configuration

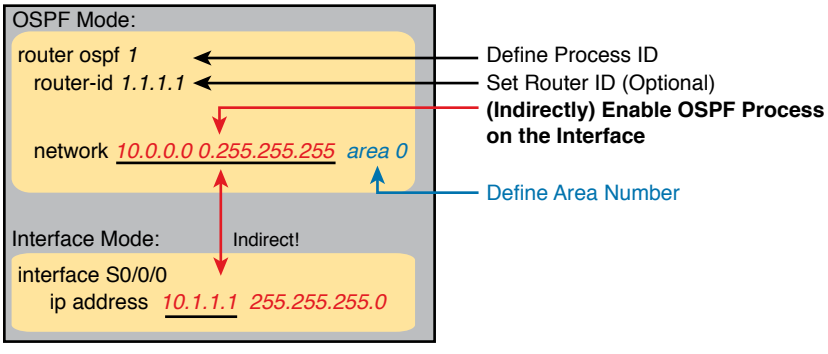


Figure 20-1 *Organization of OSPFv2 Configuration with the* **network** *Command*

OSPF Single-Area Configuration

Figure 20-2 shows a sample network that will be used for most examples throughout this chapter. All links reside in area 0, making the area design a single-area design, with four routers. You can think of Router R1 as a router at a central site, with WAN links to each remote site, and using router-on-a-stick (ROAS) to connect to two LAN subnets on the left. Routers R2 and R3 might be at one large remote site that needs two WAN links and two routers for WAN redundancy, with both routers connected to the LAN at that remote site. Router R4 might be a typical smaller remote site with a single router needed for that site.

NOTE The interface numbering on Router R1, with interfaces G0/0 and G0/0/0, may seem a bit strange. However, real routers, like the Cisco 2901 used in the example, use this numbering. That model includes a built-in Gi0/0 and Gi0/1 port. Additionally, if you add one-port Gigabit WAN Interface Cards (WICs), the router numbers them G0/0/0, G0/1/0, and so on. This is just one example of how router hardware may use two-digit interface numbering, or three-digit, or both.

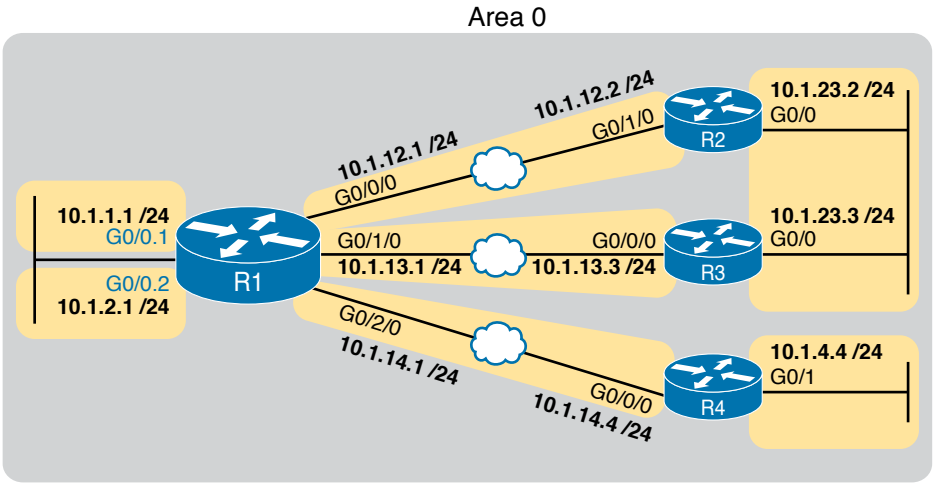


Figure 20-2 *Sample Network for OSPF Single-Area Configuration*

Example 20-1 shows the IPv4 addressing configuration on Router R1, before getting into the OSPF detail. Note that R1 enables 802.1Q trunking (ROAS) on its G0/0 interface and assigns an IP address to each subinterface.

Example 20-1 *IPv4 Address Configuration on R1 (Including VLAN Trunking)*

```
interface GigabitEthernet0/0.1
 encapsulation dot1q 1 native
 ip address 10.1.1.1 255.255.255.0
!
interface GigabitEthernet0/0.2
 encapsulation dot1q 2
 ip address 10.1.2.1 255.255.255.0
!
interface GigabitEthernet0/0/0
 ip address 10.1.12.1 255.255.255.0
!
interface GigabitEthernet0/1/0
 ip address 10.1.13.1 255.255.255.0
!
interface GigabitEthernet0/2/0
 ip address 10.1.14.1 255.255.255.0
```

The OSPF configuration begins with the **router ospf** *process-id* global command, which puts the user in OSPF configuration mode, and sets the OSPF *process-id* value. The *process-id* number just needs to be unique on the local router, allowing the router to support multiple OSPF processes in a single router by using different process IDs. (The

---

Answers to the "Do I Know This Already?" quiz:

1 B **2** A **3** A, E **4** B **5** B **6** A, D

router command uses the *process-id* to distinguish between the processes.) The *process-id* does not have to match on each router, and it can be any integer between 1 and 65,535.

Second, the configuration needs one or more **network** commands in OSPF mode. These commands tell the router to find its local interfaces that match the first two parameters on the **network** command. Then, for each matched interface, the router enables OSPF on those interfaces, discovers neighbors, creates neighbor relationships, and assigns the interface to the area listed in the **network** command. (Note that the area can be configured as either an integer or a dotted-decimal number, but this book makes a habit of configuring the area number as an integer. The integer area numbers range from 0 through 4,294,967,295.)

Example 20-2 shows an example configuration on router R1 from Figure 20-2. The **router ospf 1** command enables OSPF process 1, and the single **network** command enables OSPF on all interfaces shown in the figure.

Example 20-2 *OSPF Single-Area Configuration on R1 Using One* **network** *Command*

```
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
```

For the specific **network** command in Example 20-2, any matched interfaces are assigned to area 0. However, the first two parameters—the *ip_address* and *wildcard_mask* parameter values of 10.0.0.0 and 0.255.255.255—need some explaining. In this case, the command matches both interfaces shown for Router R2; the next topic explains why.

Wildcard Matching with the network Command

The key to understanding the traditional OSPFv2 configuration shown in this first example is to understand the OSPF **network** command. The OSPF **network** command compares the first parameter in the command to each interface IP address on the local router, trying to find a match. However, rather than comparing the entire number in the **network** command to the entire IPv4 address on the interface, the router can compare a subset of the octets, based on the wildcard mask, as follows:

Key Topic

Wildcard 0.0.0.0: Compare all four octets. In other words, the numbers must exactly match.

Wildcard 0.0.0.255: Compare the first three octets only. Ignore the last octet when comparing the numbers.

Wildcard 0.0.255.255: Compare the first two octets only. Ignore the last two octets when comparing the numbers.

Wildcard 0.255.255.255: Compare the first octet only. Ignore the last three octets when comparing the numbers.

Wildcard 255.255.255.255: Compare nothing; this wildcard mask means that all addresses will match the **network** command.

Basically, a wildcard mask value of decimal 0 in an octet tells IOS to compare to see if the numbers match, and a value of 255 tells IOS to ignore that octet when comparing the numbers.

The **network** command provides many flexible options because of the wildcard mask. For example, in Router R1, many **network** commands could be used, with some matching all interfaces, and some matching a subset of interfaces. Table 20-2 shows a sampling of options, with notes.

Table 20-2 Example OSPF **network** Commands on R1, with Expected Results

Command	Logic in Command	Matched Interfaces
network 10.1.0.0 0.0.255.255	Match addresses that begin with 10.1	G0/0.1 G0/0.2 G0/0/0 G0/1/0 G0/2/0
network 10.0.0.0 0.255.255.255	Match addresses that begin with 10	G0/0.1 G0/0.2 G0/0/0 G0/1/0 G0/2/0
network 0.0.0.0 255.255.255.255	Match all addresses	G0/0.1 G0/0.2 G0/0/0 G0/1/0 G0/2/0
network 10.1.13.0 0.0.0.255	Match addresses that begin with 10.1.13	G0/1/0
network 10.1.13.1 0.0.0.0	Match one address: 10.1.13.1	G0/1/0

The wildcard mask gives the local router its rules for matching its own interfaces. To show examples of the different options, Example 20-3 shows the configuration on routers R2, R3, and R4, each using different wildcard masks. Note that all three routers (R2, R3, and R4) enable OSPF on all the interfaces shown in Figure 20-2.

Example 20-3 *OSPF Configuration on Routers R2, R3, and R4*

```
! R2 configuration next - one network command enables OSPF on both interfaces
interface GigabitEthernet0/0
 ip address 10.1.23.2 255.255.255.0
!
interface GigabitEthernet0/1/0
 ip address 10.1.12.2 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
```

```
! R3 configuration next - One network command per interface
interface GigabitEthernet0/0
 ip address 10.1.23.3 255.255.255.0
```

```
!
interface GigabitEthernet0/0/0
 ip address 10.1.13.3 255.255.255.0
!
router ospf 1
 network 10.1.13.3 0.0.0.0 area 0
 network 10.1.23.3 0.0.0.0 area 0
```

```
! R4 configuration next - One network command per interface with wildcard 0.0.0.255
interface GigabitEthernet0/1
 ip address 10.1.4.4 255.255.255.0
!
interface GigabitEthernet0/0/0
 ip address 10.1.14.4 255.255.255.0
!
router ospf 1
 network 10.1.14.0 0.0.0.255 area 0
 network 10.1.4.0 0.0.0.255 area 0
```

Finally, note that OSPF uses the same wildcard mask logic as defined by Cisco IOS access control lists. The section titled "Finding the Right Wildcard Mask to Match a Subnet" section in Chapter 2 of the *CCNA 200-301 Official Cert Guide*, *Volume 2*, provides more detail about wildcard masks.

NOTE IOS will change a **network** command if it does not follow a particular rule: by convention, if the wildcard mask octet is 255, the matching address octet should be configured as a 0. Interestingly, IOS will actually accept a **network** command that breaks this rule, but then IOS will change that octet of the address to a 0 before putting it into the running configuration file. For example, IOS will change a typed command that begins with **network 1.2.3.4 0.0.255.255** to **network 1.2.0.0 0.0.255.255**.

Verifying OSPF Operation

As mentioned in Chapter 19, "Understanding OSPF Concepts," OSPF routers use a three-step process to eventually add OSPF-learned routes to the IP routing table. First, they create neighbor relationships. Then they build and flood LSAs between those neighbors so each router in the same area has a copy of the same LSDB. Finally, each router independently computes its own IP routes using the SPF algorithm and adds them to its routing table. This next topic works through how to display the results of each of those steps, which lets you confirm whether OSPF has worked correctly or not.

The **show ip ospf neighbor**, **show ip ospf database**, and **show ip route** commands display information to match each of these three steps, respectively. Figure 20-3 summarizes the commands you can use (and others) when verifying OSPF.

Many engineers begin OSPF verification by looking at the output of the **show ip ospf neighbor** command. For instance, Example 20-4 shows a sample from Router R1, which should have one neighbor relationship each with routers R2, R3, and R4. Example 20-4 shows all three.

Key Topic

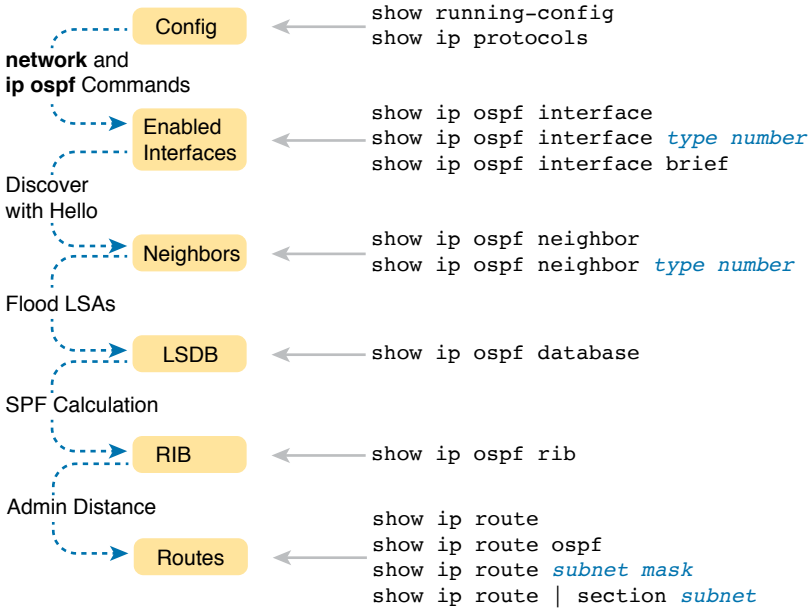


Figure 20-3 *OSPF Verification Commands*

Key Topic

Example 20-4 *OSPF Neighbors on Router R1 from Figure 20-2*

```
R1# show ip ospf neighbor
Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   FULL/DR         00:00:37    10.1.12.2       GigabitEthernet0/0/0
3.3.3.3           1   FULL/DR         00:00:37    10.1.13.3       GigabitEthernet0/1/0
4.4.4.4           1   FULL/BDR        00:00:34    10.1.14.4       GigabitEthernet0/2/0
```

The detail in the output mentions several important facts, and for most people, working right to left works best in this case. For example, look at the headings:

Interface: This is the local router's interface connected to the neighbor. For example, the first neighbor in the list is reachable through R1's G0/0/0 interface.

Address: This is the neighbor's IP address on that link. Again, for this first neighbor, which is R1, uses IP address 10.1.13.1.

State: While many possible states exist, for the details discussed in this chapter, FULL is the correct and fully working state in this case.

Neighbor ID: This is the router ID of the neighbor.

Once OSPF convergence has completed, a router should list each neighbor. On links that use a designated router (DR), the state will also list the role of the neighboring router after the / (DR, BDR, or DROTHER. As a result, the normal working states will be:

Key Topic

FULL/ -: The neighbor state is full, with the "-" instead of letters meaning that the link does not use a DR/BDR.

FULL/DR: The neighbor state is full, and the neighbor is the DR.

FULL/BDR: The neighbor state is full, and the neighbor is the backup DR (BDR).

FULL/DROTHER: The neighbor state is full, and the neighbor is neither the DR nor BDR. (It also implies that the local router is a DR or BDR because the state is FULL.)

2WAY/DROTHER: The neighbor state is 2-way, and the neighbor is neither the DR nor BDR—that is, a DROther router. (It also implies that the local router is also a DROther router because otherwise the state would reach a full state.)

Once a router's OSPF process forms a working neighbor relationship, the routers exchange the contents of their LSDBs, either directly or through the DR on the subnet. Example 20-5 shows the contents of the LSDB on Router R1. Interestingly, with a single-area design, all the routers will have the same LSDB contents once all neighbors are up and all LSAs have been exchanged. So, the **show ip ospf database** command in Example 20-5 should list the same exact information, no matter on which of the four routers it is issued.

Example 20-5 *OSPF Database on Router R1 from Figure 20-2*

```
R1# show ip ospf database

            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         431         0x8000008F 0x00DCCA 5
2.2.2.2         2.2.2.2         1167        0x8000007F 0x009DA1 2
3.3.3.3         3.3.3.3         441         0x80000005 0x002FB1 1
4.4.4.4         4.4.4.4         530         0x80000004 0x007F39 2

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
10.1.12.2       2.2.2.2         1167        0x8000007C 0x00BBD5
10.1.13.3       3.3.3.3         453         0x80000001 0x00A161
10.1.14.1       1.1.1.1         745         0x8000007B 0x004449
10.1.23.3       3.3.3.3         8           0x80000001 0x00658F
```

For the purposes of this book, do not be concerned about the specifics in the output of this command. However, for perspective, note that the LSDB should list one "Router Link State" (Type 1 Router LSA) for each of the routers in the same area, so with the design based on Figure 20-2, the output lists four Type 1 LSAs. Also, with all default settings in this design, the routers will create a total of four Type 2 Network LSAs as shown, one each for the subnets that have a DR and contain at least two routers in that subnet.

Next, Example 20-6 shows R4's IPv4 routing table with the **show ip route** command. As configured, with all links working, the design in Figure 20-2 includes seven subnets. R4 has

connected routes to two of those subnets and should learn OSPF routes to the other five subnets.

Example 20-6 *IPv4 Routes Added by OSPF on Router R4 from Figure 20-2*

```
R4# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
! Additional legend lines omitted for brevity

Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 9 subnets, 2 masks
O        10.1.1.0/24 [110/2] via 10.1.14.1, 00:27:24, GigabitEthernet0/0/0
O        10.1.2.0/24 [110/2] via 10.1.14.1, 00:27:24, GigabitEthernet0/0/0
C        10.1.4.0/24 is directly connected, GigabitEthernet0/1
L        10.1.4.4/32 is directly connected, GigabitEthernet0/1
O        10.1.12.0/24 [110/2] via 10.1.14.1, 00:27:24, GigabitEthernet0/0/0
O        10.1.13.0/24 [110/2] via 10.1.14.1, 00:25:15, GigabitEthernet0/0/0
C        10.1.14.0/24 is directly connected, GigabitEthernet0/0/0
L        10.1.14.4/32 is directly connected, GigabitEthernet0/0/0
O        10.1.23.0/24 [110/3] via 10.1.14.1, 00:27:24, GigabitEthernet0/0/0
```

Any time you want to check OSPF on a router in a small design like the ones in the book, you can count all the subnets, then count the subnets connected to the local router, and know that OSPF should learn routes to the rest of the subnets. Then just use the **show ip route** command and add up how many connected and OSPF routes exist as a quick check of whether all the routes have been learned or not.

In this case, router R4 has two connected subnets, but seven subnets exist per the figure, so router R4 should learn five OSPF routes. Next look for the code of "O" on the left, which identifies a route as being learned by OSPF. The output lists five such IP routes: two for the LAN subnets off Router R1, one for the LAN subnets connected to both R2 and R3, and one each for the WAN subnets from R1 to R2 and R1 to R3.

Next, take a look at the first route (to subnet 10.1.1.0/24). It lists the subnet ID and mask, identifying the subnet. It also lists two numbers in brackets. The first, 110, is the administrative distance of the route. All the OSPF routes in this example use the default of 110 (see Chapter 19's Table 19-4 for the list of administrative distance values). The second number, 2, is the OSPF metric for this route. The route also lists the forwarding instructions: the next-hop IP address (10.1.14.1) and R4's outgoing interface (G0/0/0).

Verifying OSPF Configuration

Once you can configure OSPF with confidence, you will likely verify OSPF focusing on OSPF neighbors and the IP routing table as just discussed. However, if OSPF does not work

immediately, you may need to circle back and check the configuration. To do so, you can use these steps:

- If you have enable mode access, use the **show running-config** command to examine the configuration.
- If you have only user mode access, use the **show ip protocols** command to re-create the OSPF configuration.
- Use the **show ip ospf interface [brief]** command to determine whether the router enabled OSPF on the correct interfaces or not based on the configuration.

NOTE The exam's Sim and Simlet questions can restrict access to enable mode, so knowing how to extract the configuration from show commands other than **show running-config** can be particularly helpful for any configuration topic.

The best way to verify the configuration begins with the **show running-config** command, of course. However, the **show ip protocols** command repeats the details of the OSPFv2 configuration and does not require enable mode access. To see how, consider Example 20-7, which lists the output of the **show ip protocols** command on router R3.

Example 20-7 *Router R3 Configuration and the* **show ip protocols** *Command*

```
! First, a reminder of R3's configuration per Example 20-3:
router ospf 1
 network 10.1.13.3 0.0.0.0 area 0
 network 10.1.23.3 0.0.0.0 area 0
!
! The output from router R3:
R3# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "ospf 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 3.3.3.3
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
    10.1.13.3 0.0.0.0 area 0
    10.1.23.3 0.0.0.0 area 0
  Routing Information Sources:
    Gateway         Distance      Last Update
    1.1.1.1              110      02:05:26
    4.4.4.4              110      02:05:26
    2.2.2.2              110      01:51:16
  Distance: (default is 110)
```

The highlighted output emphasizes some of the configuration. The first highlighted line repeats the parameters on the **router ospf 1** global configuration command. (The second highlighted item points out each router's router ID, which will be discussed in the next section.) The third set of highlighted lines begins with a heading of "Routing for Networks:" followed by two lines that closely resemble the parameters on the configured **network** commands. In fact, closely compare those last two highlighted lines with the **network** configuration commands at the top of the example, and you will see that they mirror each other, but the **show** command just leaves out the word *network*. For instance:

Configuration: network 10.1.13.3 0.0.0.0 area 0

Show Command: 10.1.13.3 0.0.0.0 area 0

IOS interprets the **network** commands to choose interfaces on which to run OSPF, so it could be that IOS chooses a different set of interfaces than you predicted. To check the list of interfaces chosen by IOS, use the **show ip ospf interface brief** command, which lists all interfaces that have been enabled for OSPF processing. Verifying the interfaces can be a useful step if you have issues with OSPF neighbors because OSPF must first be enabled on an interface before a router will attempt to discover neighbors on that interface. Example 20-8 shows a sample from Router R1.

Example 20-8 *Router R1* **show ip ospf interface brief** *Command*

```
R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/0/0      1     0               10.1.12.1/24       1     BDR   1/1
Gi0/1/0      1     0               10.1.13.1/24       1     BDR   1/1
Gi0/2/0      1     0               10.1.14.1/24       1     DR    1/1
Gi0/0.2      1     0               10.1.2.1/24        1     DR    0/0
Gi0/0.1      1     0               10.1.1.1/24        1     DR    0/0
```

First, consider the **show ip ospf interface brief** command shown here. It lists one line per interface, with the list showing all the interfaces on which OSPF has been enabled. Each item in the list identifies the OSPF process ID (per the **router ospf** *process-id* command), the area, the interface IP address, and the number of neighbors found via each interface.

More generally, note that the **show ip ospf interface** command with the **brief** keyword at the end lists a single line of output per interface, but the **show ip ospf interface** command (without the **brief** keyword) displays about 20 lines of output per interface, with much more information about various OSPF per-interface settings.

Configuring the OSPF Router ID

While OSPF has many other optional features, most enterprise networks that use OSPF choose to configure each router's OSPF router ID. OSPF-speaking routers must have a router ID (RID) for proper operation. By default, routers will choose an interface IP address to use as the RID. However, many network engineers prefer to choose each router's router ID, so command output from commands like **show ip ospf neighbor** lists more recognizable router IDs.

To choose its RID, a Cisco router uses the following process when the router reloads and brings up the OSPF process. Note that the router stops looking for a router ID to use once one of the steps identifies a value to use.

Key Topic

1. If the **router-id** *rid* OSPF subcommand is configured, this value is used as the RID.
2. If any loopback interfaces have an IP address configured, and the interface has an interface status of up, the router picks the highest numeric IP address among these loopback interfaces.
3. The router picks the highest numeric IP address from all other interfaces whose interface status code (first status code) is up. (In other words, an interface in up/down state will be included by OSPF when choosing its router ID.)

The first and third criteria should make some sense right away: the RID is either configured or is taken from a working interface's IP address. However, this book has not yet explained the concept of a *loopback interface*, as mentioned in Step 2.

A loopback interface is a virtual interface that can be configured with the **interface loopback** *interface-number* command, where *interface-number* is an integer. Loopback interfaces are always in an "up and up" state unless administratively placed in a shutdown state. For example, a simple configuration of the command **interface loopback 0**, followed by **ip address 2.2.2.2 255.255.255.0**, would create a loopback interface and assign it an IP address. Because loopback interfaces do not rely on any hardware, these interfaces can be up/up whenever IOS is running, making them good interfaces on which to base an OSPF RID.

Example 20-9 shows the configuration that existed in Routers R1 and R2 before the creation of the **show** command output earlier in this chapter. R1 set its router ID using the direct method, while R2 used a loopback IP address.

Example 20-9 *OSPF Router ID Configuration Examples*

```
! R1 Configuration first
router ospf 1
 router-id 1.1.1.1
 network 10.1.0.0 0.0.255.255 area 0
```

```
! R2 Configuration next
!
interface Loopback2
 ip address 2.2.2.2 255.255.255.255
```

Each router chooses its OSPF RID when OSPF is initialized, which happens when the router boots or when a CLI user stops and restarts the OSPF process (with the **clear ip ospf process** command). So, if OSPF comes up, and later the configuration changes in a way that would impact the OSPF RID, OSPF does not change the RID immediately. Instead, IOS waits until the next time the OSPF process is restarted.

Example 20-10 shows the output of the **show ip ospf** command on R1, which identifies the OSPF RID used by R1.

Example 20-10 *Confirming the Current OSPF Router ID*

```
R1# show ip ospf
 Routing Process "ospf 1" with ID 1.1.1.1
! lines omitted for brevity
```

Implementing Multiarea OSPF

Even though the current CCNA 200-301 exam blueprint mentions single-area OSPF and does not mention multiarea OSPF, you only need to learn one more idea to know how to configure multiarea OSPF. So, this chapter takes a brief page to show how.

For example, consider a multiarea OSPF design as shown in Figure 20-4. It uses the same routers and IP addresses as shown earlier in Figure 20-2, on which all the examples in this chapter have been based so far. However, the design shows three areas instead of the single-area design shown in Figure 20-2.

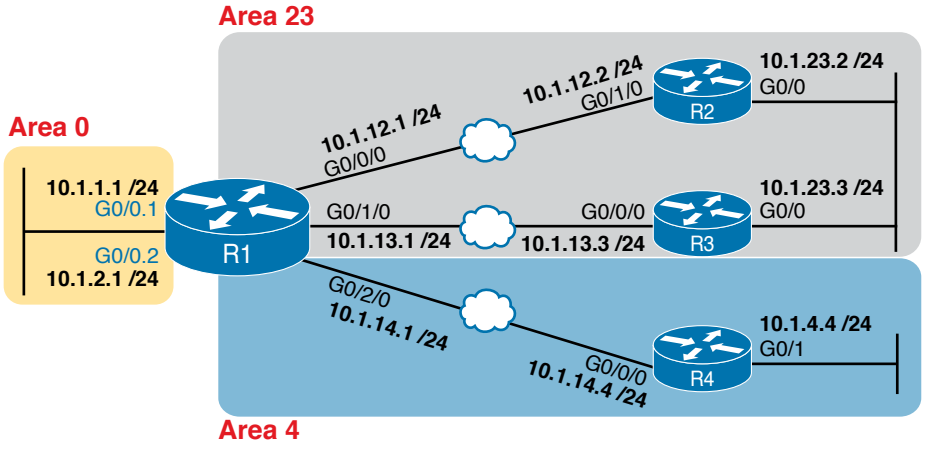


Figure 20-4 *Area Design for an Example Multiarea OSPF Configuration*

Configuring the routers in a multiarea design is almost like configuring OSPFv2 for a single area. To configure multiarea OSPF, all you need is a valid OSPF area design (for instance, like Figure 20-4) and a configuration that places each router interface into the correct area per that design. For example, both of R4's interfaces connect to links in area 4, making R4 an internal router, so any **network** commands on router R4 will list area 4.

Example 20-11 shows a sample configuration for Router R1. To make the configuration clear, it uses **network** commands with a wildcard mask of 0.0.0.0, meaning each **network** command matches a single interface. Each interface will be placed into either area 0, 23, or 4 to match the figure.

Example 20-11 *OSPF Configuration on R1, Placing Interfaces into Different Areas*

```
router ospf 1
 network 10.1.1.1 0.0.0.0 area 0
 network 10.1.2.1 0.0.0.0 area 0
 network 10.1.12.1 0.0.0.0 area 23
 network 10.1.13.1 0.0.0.0 area 23
 network 10.1.14.1 0.0.0.0 area 4
```

Using OSPFv2 Interface Subcommands

From the earliest days of OSPFv2 support in Cisco routers, the configuration used the OSPF **network** command as discussed in this chapter. However, that configuration style can be confusing, and it does require some interpretation of the **network** commands and interface IP addresses to decide on which interfaces IOS will enable OSPF. As a result, Cisco added another option for OSPFv2 configuration called OSPF interface configuration.

The newer interface-style OSPF configuration still enables OSPF on interfaces, but it does so directly with the **ip ospf** interface subcommand instead of using the **network** command in router configuration mode. Basically, instead of matching interfaces with indirect logic using **network** commands, you directly enable OSPFv2 on interfaces by configuring an interface subcommand on each interface.

OSPF Interface Configuration Example

To show how OSPF interface configuration works, this example basically repeats the example shown earlier in the book using the traditional OSPFv2 configuration with **network** commands. So, before looking at the OSPFv2 interface configuration, take a moment to look back to review traditional OSPFv2 configuration with Figure 20-2 and Examples 20-2 and 20-3.

After reviewing the traditional configuration, consider this checklist, which details how to convert from the old-style configuration in Examples 20-2 and 20-3 to use interface configuration:

Config Checklist

Step 1. Use the **no network** *network-id* **area** *area-id* subcommands in OSPF configuration mode to remove the **network** commands.

Step 2. Add one **ip ospf** *process-id* **area** *area-id* command in interface configuration mode under each interface on which OSPF should operate, with the correct OSPF process (*process-id*) and the correct OSPF area number.

Figure 20-5 repeats the design for both the original examples in this chapter and for this upcoming interface configuration example.

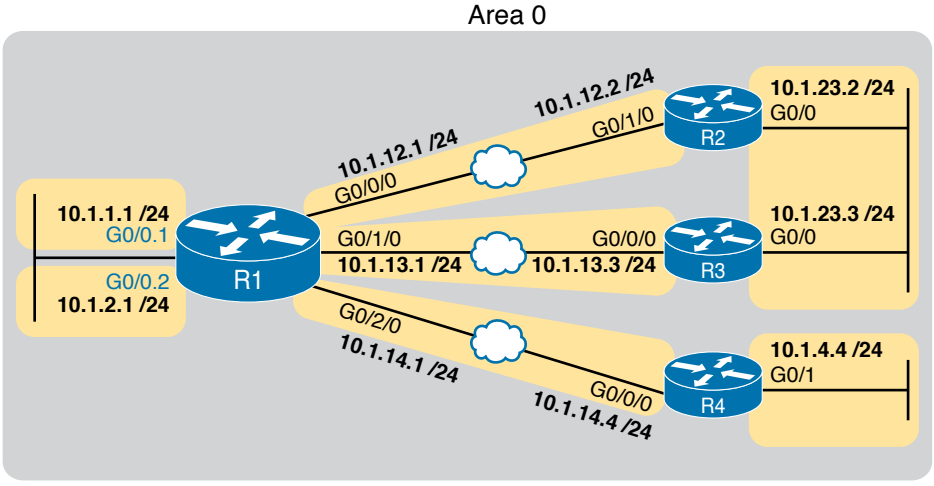


Figure 20-5 *Area Design Used in the Upcoming OSPF Interface Config Example*

Example 20-2 shows a single **network** command: **network 10.0.0.0 0.255.255.255 area 0**. Example 20-12 follows the steps in the migration checklist, beginning with the removal of the previous configuration using the **no network 10.0.0.0 0.255.255.255 area 0** command. The example then shows the addition of the **ip ospf 1 area 0** command on each of the five interfaces on Router R1, enabling OSPF process 1 on the interface and placing each interface into area 0.

Example 20-12 *OSPF Single-Area Configuration on R1 Using One* **network** *Command*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# router ospf 1
R1(config-router)# no network 10.0.0.0 0.255.255.255 area 0
R1(config-router)#
*Apr  8 19:35:24.994: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on GigabitEthernet0/0/0
  from FULL to DOWN, Neighbor Down: Interface down or detached
*Apr  8 19:35:24.994: %OSPF-5-ADJCHG: Process 1, Nbr 3.3.3.3 on GigabitEthernet0/1/0
  from FULL to DOWN, Neighbor Down: Interface down or detached
*Apr  8 19:35:24.994: %OSPF-5-ADJCHG: Process 1, Nbr 4.4.4.4 on GigabitEthernet0/2/0
  from FULL to DOWN, Neighbor Down: Interface down or detached
R1(config-router)# interface g0/0.1
R1(config-subif)# ip ospf 1 area 0
R1(config-subif)# interface g0/0.2
R1(config-subif)# ip ospf 1 area 0
R1(config-subif)# interface g0/0/0
R1(config-if)# ip ospf 1 area 0
R1(config-if)#
*Apr  8 19:35:52.970: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on GigabitEthernet0/0/0
  from LOADING to FULL, Loading Done
R1(config-if)# interface g0/1/0
R1(config-if)# ip ospf 1 area 0
```

```
R1(config-if)#
*Apr  8 19:36:13.362: %OSPF-5-ADJCHG: Process 1, Nbr 3.3.3.3 on GigabitEthernet0/1/0
  from LOADING to FULL, Loading Done
R1(config-if)# interface g0/2/0
R1(config-if)# ip ospf 1 area 0
R1(config-if)#
*Apr  8 19:37:05.398: %OSPF-5-ADJCHG: Process 1, Nbr 4.4.4.4 on GigabitEthernet0/2/0
  from LOADING to FULL, Loading Done
R1(config-if)#
```

When reading the example, read from top to bottom, and also consider the details about the failed and recovered neighbor relationships shown in the log messages. Removing the **network** command disabled OSPF on all interfaces on Router R1, causing all three neighbor relationships to fail. The example then shows the addition of the **ip ospf 1 area 0** command on the two LAN subinterfaces, which enables OSPF. Then the example shows the same command added to each of the WAN links in succession, and in each case, the OSPF neighbor available over that WAN link comes up (as noted in the log messages.)

Verifying OSPF Interface Configuration

OSPF operates the same way whether you use the new style or old style of configuration. The OSPF area design works the same, neighbor relationships form the same way, routers negotiate to become the DR and BDR the same way, and so on. However, you can see a few small differences in show command output when using the newer OSPFv2 configuration if you look closely.

The **show ip protocols** command relists most of the routing protocol configuration, so it does list some different details if you use interface configuration versus the **network** command. With the newer-style configuration, the output lists the phrase "Interfaces Configured Explicitly," with the list of interfaces configured with the new **ip ospf** *process-id* **area** *area-id* commands, as highlighted in Example 20-13. The example first shows the relevant parts of the **show ip protocols** command when using interface configuration on Router R1, and then lists the same portions of the command from when R1 used **network** commands.

Example 20-13 *Differences in* **show ip protocols** *Output: Old- and New-Style OSPFv2 Configuration*

```
! First, with the new interface configuration
R1# show ip protocols
! … beginning lines omitted for brevity
  Routing for Networks:
  Routing on Interfaces Configured Explicitly (Area 0):
    GigabitEthernet0/2/0
    GigabitEthernet0/1/0
    GigabitEthernet0/0/0
    GigabitEthernet0/0.2
    GigabitEthernet0/0.1
  Routing Information Sources:
    Gateway         Distance      Last Update
```

```
    4.4.4.4               110        00:09:30
    2.2.2.2               110        00:10:49
    3.3.3.3               110        05:20:07
  Distance: (default is 110)
```

```
! For comparison, the old results with the use of the OSPF network command
R1# show ip protocols
! … beginning lines omitted for brevity
  Routing for Networks:
    10.1.0.0 0.0.255.255 area 0
! … ending line omitted for brevity
```

Another small piece of different output exists in the **show ip ospf interface** [*interface*] command. The command lists details about OSPF settings for the interface(s) on which OSPF is enabled. The output also makes a subtle reference to whether that interface was enabled for OSPF with the old or new configuration style. Example 20-14 also begins with output based on interface configuration on Router R1, followed by the output that would exist if R1 still used the old-style **network** command.

Key Topic

Example 20-14 *Differences in* **show ip ospf interface** *Output with OSPFv2 Interface Configuration*

```
! First, with the new interface configuration
R1# show ip ospf interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Internet Address 10.1.12.1/24, Area 0, Attached via Interface Enable
! Lines omitted for brevity
```

```
! For comparison, the old results with the use of the OSPF network command
R1# show ip ospf interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Internet Address 10.1.12.1/24, Area 0, Attached via Network Statement
! … ending line omitted for brevity
```

Other than these small differences in a few show commands, the rest of the commands show nothing different depending on the style of configuration. For instance, the **show ip ospf interface brief** command does not change depending on the configuration style, nor do the **show ip ospf database**, **show ip ospf neighbor**, or **show ip route** commands.

Additional OSPFv2 Features

This final major section of the chapter discusses some very popular but optional OSPFv2 configuration features, as listed here in their order of appearance:

- Passive interfaces
- Default routes
- Metrics
- Load balancing

OSPF Passive Interfaces

Once OSPF has been enabled on an interface, the router tries to discover neighboring OSPF routers and form a neighbor relationship. To do so, the router sends OSPF Hello messages on a regular time interval (called the Hello Interval). The router also listens for incoming Hello messages from potential neighbors.

Sometimes, a router does not need to form neighbor relationships with neighbors on an interface. Often, no other routers exist on a particular link, so the router has no need to keep sending those repetitive OSPF Hello messages. In such cases, an engineer can make the interface passive, which means

- OSPF continues to advertise about the subnet that is connected to the interface.
- OSPF no longer sends OSPF Hellos on the interface.
- OSPF no longer processes any received Hellos on the interface.

The result of enabling OSPF on an interface but then making it passive is that OSPF still advertises about the connected subnet, but OSPF also does not form neighbor relationships over the interface.

To configure an interface as passive, two options exist. First, you can add the following command to the configuration of the OSPF process, in router configuration mode:

passive-interface *type number*

Alternately, the configuration can change the default setting so that all interfaces are passive by default and then add a **no passive-interface** command for all interfaces that need to not be passive:

passive-interface default

no passive-interface *type number*

For example, in the sample internetwork in Figure 20-2 (and in Figure 20-5), Router R1, on the left side of the figure, has a LAN interface configured for VLAN trunking. The only router connected to both VLANs is Router R1, so R1 will never discover an OSPF neighbor on these subnets. Example 20-15 shows two alternative configurations to make the two LAN subinterfaces passive to OSPF.

Example 20-15 *Configuring Passive Interfaces on R1 from Figure 20-5*

```
! First, make each subinterface passive directly
router ospf 1
 passive-interface GigabitEthernet0/0.1
 passive-interface GigabitEthernet0/0.2

! Or, change the default to passive, and make the other interfaces not be passive
router ospf 1
 passive-interface default
 no passive-interface GigabitEthernet0/0/0
 no passive-interface GigabitEthernet0/1/0
 no passive-interface GigabitEthernet0/2/0
```

In real internetworks, the choice of configuration style reduces to which option requires the least number of commands. For example, a router with 20 interfaces, 18 of which are passive to OSPF, has far fewer configuration commands when using the **passive-interface default** command to change the default to passive. If only two of those 20 interfaces need to be passive, use the default setting, in which all interfaces are not passive, to keep the configuration shorter.

Interestingly, OSPF makes it a bit of a challenge to use **show** commands to find whether or not an interface is passive. The **show running-config** command lists the configuration directly, but if you cannot get into enable mode to use this command, note these two facts:

The **show ip ospf interface brief** command lists all interfaces on which OSPF is enabled, *including passive interfaces*.

The **show ip ospf interface** command lists a single line that mentions that the interface is passive.

Example 20-16 shows these two commands on Router R1, based on the configuration shown in the top of Example 20-15. Note that subinterfaces G0/0.1 and G0/0.2 both show up in the output of **show ip ospf interface brief**.

Example 20-16 *Displaying Passive Interfaces*

```
R1# show ip ospf interface brief
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES manual up                    up
GigabitEthernet0/0.1       10.1.1.1        YES manual up                    up
GigabitEthernet0/0.2       10.1.2.1        YES manual up                    up
GigabitEthernet0/1         unassigned      YES manual administratively down down
GigabitEthernet0/0/0       10.1.12.1       YES manual up                    up
GigabitEthernet0/1/0       10.1.13.1       YES manual up                    up
GigabitEthernet0/2/0       10.1.14.1       YES manual up                    up

R1# show ip ospf interface g0/0.1
GigabitEthernet0/0.1 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0, Attached via Network Statement
  Process ID 1, Router ID 1.1.1.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 1.1.1.1, Interface address 10.1.1.1
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    No Hellos (Passive interface)
! Lines omitted for brevity
```

OSPF Default Routes

Chapter 16, "Configuring IPv4 Addressing and Static Routes," showed some of the uses and benefits of default routes, with examples of static default routes. For those exact same reasons, networks can use OSPF to advertise default routes.

The most classic case for using a routing protocol to advertise a default route has to do with an enterprise's connection to the Internet. As a strategy, the enterprise engineer uses these design goals:

- All routers learn specific (nondefault) routes for subnets inside the company; a default route is not needed when forwarding packets to these destinations.
- One router connects to the Internet, and it has a default route that points toward the Internet.
- All routers should dynamically learn a default route, used for all traffic going to the Internet, so that all packets destined to locations in the Internet go to the one router connected to the Internet.

Figure 20-6 shows the idea of how OSPF advertises the default route, with the specific OSPF configuration. In this case, a company connects to an ISP with its Router R1. That router has a static default route (destination 0.0.0.0, mask 0.0.0.0) with a next-hop address of the ISP router. Then the use of the OSPF **default-information originate** command (Step 2) makes the router advertise a default route using OSPF to the remote routers (B1 and B2).

Key Topic

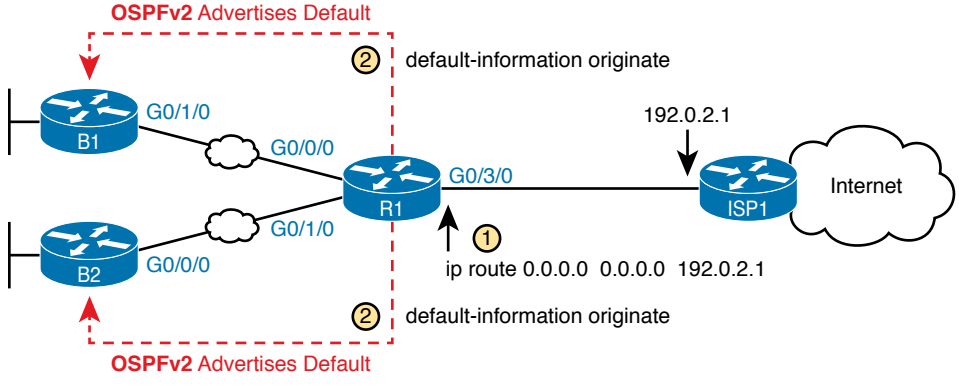


Figure 20-6 *Using OSPF to Create and Flood a Default Route*

Figure 20-7 shows the default routes that result from OSPF's advertisements in Figure 20-6. On the far left, the branch routers all have OSPF-learned default routes, pointing to R1. R1 itself also needs a default route, pointing to the ISP router, so that R1 can forward all Internet-bound traffic to the ISP.

Finally, this feature gives the engineer control over when the router originates this default route. First, R1 needs a default route, either defined as a static default route, learned from the ISP with DHCP or learned from the ISP with a routing protocol like eBGP. The OSPF subcommand **default-information originate** then tells OSPF on R1 to advertise a default route when its own default route is working and to advertise the default route as down when its own default route fails.

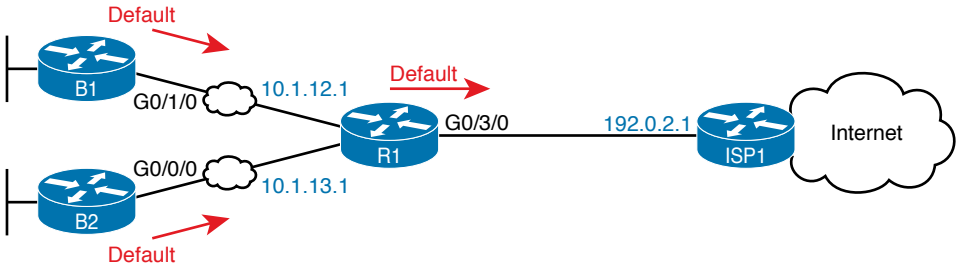


Figure 20-7 *Default Routes Resulting from the* **default-information originate** *Command*

NOTE Interestingly, the **default-information originate always** router subcommand tells the router to always advertise the default route, no matter whether the router's default route is working or not.

Example 20-17 shows details of the default route on both R1 and branch router B1 from Figure 20-7. R1 then creates a static default route with the ISP router's IP address of 192.0.2.1 as the next-hop address, as highlighted in the output of the **show ip route static** command output.

Example 20-17 *Default Routes on Routers R1 and B1*

```
! The next command is from Router R1. Note the static code for the default route
R1# show ip route static
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! Rest of the legend omitted for brevity

Gateway of last resort is 192.0.2.1 to network 0.0.0.0

S*      0.0.0.0/0 [254/0] via 192.0.2.1
```

```
! The next command is from router B01; notice the External route code for the default
B1# show ip route ospf
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
! Rest of the legend omitted for brevity

Gateway of last resort is 10.1.12.1 to network 0.0.0.0

O*E2    0.0.0.0/0 [110/1] via 10.1.12.1, 00:20:51, GigabitEthernet0/1/0
       10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O         10.1.3.0/24 [110/3] via 10.1.12.1, 00:20:51, GigabitEthernet0/1/0
O         10.1.13.0/24 [110/2] via 10.1.12.1, 00:20:51, GigabitEthernet0/1/0
```

Keeping the focus on the command on Router R1, note that R1 indeed has a default route—that is, a route to 0.0.0.0/0. The "Gateway of last resort," which refers to the default route currently used by the router, points to next-hop IP address 192.0.2.1, which is the ISP router's IP address. (Refer to Figure 20-7 for the particulars.)

Next look to the bottom half of the example and router B1's OSPF-learned default route. B1 lists a route for 0.0.0.0/0 as well. The next-hop router in this case is 10.1.12.1, which is Router R1's IP address on the WAN link. The code on the far left is O*E2, meaning an OSPF-learned route, which is a default route, and is specifically an external OSPF route. Finally, B1's gateway of last resort setting uses that one OSPF-learned default route, with next-hop router 10.1.12.1.

OSPF Metrics (Cost)

The section "Calculating the Best Routes with SPF" in Chapter 19 discussed how SPF calculates the metric for each route, choosing the route with the best metric for each destination subnet. OSPF routers can influence that choice by changing the OSPF interface cost on any and all interfaces.

Cisco routers allow three different ways to change the OSPF interface cost:

- Directly, using the interface subcommand **ip ospf cost** *x*.
- Using the default calculation per interface, and changing the *interface bandwidth* setting, which changes the calculated value.
- Using the default calculation per interface, and changing the OSPF *reference bandwidth* setting, which changes the calculated value.

Setting the Cost Directly

Setting the cost directly requires a simple configuration command, as shown in Example 20-18. The example sets the cost of two interfaces on Router R1. (This example uses the Figure 20-2 design, as configured in Examples 20-2 and 20-3.) The **show ip ospf interface brief** command that follows details the cost of each interface. Note that the show command confirms the cost settings.

Example 20-18 *Confirming OSPF Interface Costs*

```
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# interface g0/0/0
R1(config-if)# ip ospf cost 4
R1(config-if)# interface g0/1/0
R1(config-if)# ip ospf cost 5
R1(config-if)# ^Z
R1#
R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/0.2      1     0               10.1.2.1/24        1     DR    0/0
Gi0/0.1      1     0               10.1.1.1/24        1     DR    0/0
```

```
Gi0/0/0         1     0               10.1.12.1/24       4     DR    1/1
Gi0/1/0         1     0               10.1.13.1/24       5     BDR   1/1
Gi0/2/0         1     0               10.1.14.1/24       1     DR    1/1
```

The output also shows a cost value of 1 for the other Gigabit interfaces, which is the default OSPF cost for any interface faster than 100 Mbps. The next topic discusses how IOS determines the default cost values.

Setting the Cost Based on Interface and Reference Bandwidth

Routers use a per-interface bandwidth setting to describe the speed of the interface. Note that the interface bandwidth setting does not influence the actual transmission speed. Instead, the interface bandwidth acts as a configurable setting to represent the speed of the interface, with the option to configure the bandwidth to match the actual transmission speed…or not. To support this logic, IOS sets a default interface bandwidth value that matches the physical transmission speed when possible, but also allows the configuration of the interface bandwidth using **bandwidth** *speed* interface subcommand.

OSPF (as well as other IOS features) uses the interface bandwidth to make decisions, with OSPF using the interface bandwidth in its calculation of the default OSPF cost for each interface. IOS uses the following formula to choose an interface's OSPF cost if the cost for cases in which the **ip ospf cost** command is not configured on the interface. IOS puts the interface's bandwidth in the denominator and an OSPF setting called the *reference bandwidth* in the numerator:

Reference_bandwidth / Interface_bandwidth

Note that while you can change both the interface bandwidth and reference bandwidth via configuration, because several IOS features make use of the bandwidth setting, you should avoid changing the interface bandwidth as a means to influence the default OSPF cost.

That being said, many enterprises do use default cost settings while influencing the default by changing the OSPF reference bandwidth while leaving the interface bandwidth as an accurate representation of link speed. Cisco chose the IOS default reference bandwidth setting decades ago in an era with much slower links. As a result, any interface with an interface bandwidth of 100 Mbps or faster ties with a calculated OSPF cost of 1 when using the default reference bandwidth. So, when relying on the default OSPF cost calculation, it helps to configure the reference bandwidth to another value.

To see the issue, consider Table 20-3, which lists several types of interfaces, the default interface bandwidth on those interfaces, and the OSPF cost calculated with the default OSPF reference bandwidth of 100 MBps (that is, 100,000 Kbps). (OSPF rounds up for these calculations, resulting in a lowest possible OSPF interface cost of 1.)

Table 20-3 Faster Interfaces with Equal OSPF Costs

Interface	Interface Default Bandwidth (Kbps)	Formula (Kbps)	OSPF Cost
Serial	1544 Kbps	100,000 / 1544	64
Ethernet	10,000 Kbps	100,000 / 10,000	10
Fast Ethernet	100,000 Kbps	100,000/100,000	1
Gigabit Ethernet	1,000,000 Kbps	100,000/1,000,000	1
10 Gigabit Ethernet	10,000,000 Kbps	100,000/10,000,000	1
100 Gigabit Ethernet	100,000,000 Kbps	100,000/100,000,000	1

As you can see from the table, with a default reference bandwidth, all interfaces from Fast Ethernet's 100 Mbps and faster tie with their default OSPF cost. As a result, OSPF would treat a 100-Mbps link as having the same cost as a 10- or 100-Gbps link, which is probably not the right basis for choosing routes.

You can still use OSPF's default cost calculation (and many do) just by changing the reference bandwidth with the **auto-cost reference-bandwidth** *speed* OSPF mode subcommand. This command sets a value in a unit of megabits per second (Mbps). Set the reference bandwidth value to a value at least as much as the fastest link speed in the network, but preferably higher, in anticipation of adding even faster links in the future.

For instance, in an enterprise whose fastest links are 10 Gbps (10,000 Mbps), you could set all routers to use **auto-cost reference-bandwidth 10000**, meaning 10,000 Mbps or 10 Gbps. In that case, by default, a 10-Gbps link would have an OSPF cost of 1, while a 1-Gbps link would have a cost of 10, and a 100-MBps link a cost of 100.

Better still, in that same enterprise, use a reference bandwidth of a faster speed than the fastest interface in the network, to allow room for higher speeds. For instance, in that same enterprise, whose fastest link is 10 Gbps, set the reference bandwidth to 40 Gbps or even 100 Gbps to be ready for future upgrades to use 40-Gbps links, or even 100-Gbps links. (For example, use the **auto-cost reference-bandwidth 100000** command, meaning 100,000 Mbps or 100 Gbps.) That causes 100-Gbps links to have an OSPF cost of 1, 40-Gbps links to have a cost of 4, 10-Gbps links to have a cost of 10, and 1-Gbps links to have a cost of 100.

NOTE Cisco recommends making the OSPF reference bandwidth setting the same on all OSPF routers in an enterprise network.

For convenient study, the following list summarizes the rules for how a router sets its OSPF interface costs:

Key Topic

1. Set the cost explicitly, using the **ip ospf cost** *x* interface subcommand, to a value between 1 and 65,535, inclusive.
2. Although it should be avoided, change the interface bandwidth with the **bandwidth** *speed* command, with *speed* being a number in kilobits per second (Kbps).
3. Change the reference bandwidth, using router OSPF subcommand **auto-cost reference-bandwidth** *ref-bw*, with a unit of megabits per second (Mbps).

OSPF Load Balancing

When a router uses SPF to calculate the metric for each of several routes to reach one subnet, one route may have the lowest metric, so OSPF puts that route in the routing table. However, when the metrics tie for multiple routes to the same subnet, the router can put multiple equal-cost routes in the routing table (the default is four different routes) based on the setting of the **maximum-paths** *number* router subcommand. For example, if an internetwork has six possible paths between some parts of the network, and the engineer wants all routes to be used, the routers can be configured with the **maximum-paths 6** subcommand under **router ospf**.

The more challenging concept relates to how the routers use those multiple routes. A router could load balance the packets on a per-packet basis. For example, if the router has three equal-cost OSPF routes for the same subnet in the routing table, the router could send the one packet over the first route, the next packet over the second route, the next packet over the third route, and then start over with the first route for the next packet. Note that per-packet load balancing is generally a poor choice because it causes the most overhead work on the router. Alternatively, using the default (and better) method, the load balancing could be on a per-destination IP address basis.

Note that the default setting of **maximum-paths** varies by router platform.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 20-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 20-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used:
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review Config Checklists		Book, website
Review command tables		Book
Do labs		Blog

Review All the Key Topics

Key Topic

Table 20-5 Key Topics for Chapter 20

Key Topic Element	Description	Page Number
Figure 20-1	Organization of OSPFv2 configuration with the **network** command	471
List	Example OSPF wildcard masks and their meaning	473
Figure 20-3	OSPF verification commands	476
Example 20-4	Example of the **show ip ospf neighbor** command	476
List	Neighbor states and their meanings	477
List	Rules for setting the router ID	481
Example 20-14	Differences in **show ip ospf interface** output with OSPF interface configuration	486
List	Actions IOS takes when an OSPF interface is passive	487
Figure 20-6	Actions taken by the OSPF **default-information originate** command	489
List	Rules for setting OSPF interface cost	493

Key Terms You Should Know

reference bandwidth, interface bandwidth, maximum paths

Command References

Tables 20-6 and 20-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 20-6 Chapter 20 Configuration Command Reference

Command	Description
router ospf *process-id*	Router subcommand that enters OSPF configuration mode for the listed process.
network *ip-address wildcard-mask* area *area-id*	Router subcommand that enables OSPF on interfaces matching the address/wildcard combination and sets the OSPF area.
ip ospf *process-id* **area** *area-number*	Interface subcommand to enable OSPF on the interface and to assign the interface to a specific OSPF area.

Command	Description
ip ospf cost *interface-cost*	Interface subcommand that sets the OSPF cost associated with the interface.
bandwidth *bandwidth*	Interface subcommand that directly sets the interface bandwidth (Kbps).
auto-cost reference-bandwidth *number*	Router subcommand that tells OSPF the numerator in the Reference_bandwidth / Interface_bandwidth formula used to calculate the OSPF cost based on the interface bandwidth.
router-id *id*	OSPF command that statically sets the router ID.
interface loopback *number*	Global command to create a loopback interface and to navigate to interface configuration mode for that interface.
maximum-paths *number-of-paths*	Router subcommand that defines the maximum number of equal-cost routes that can be added to the routing table.
passive-interface *type number*	Router subcommand that makes the interface passive to OSPF, meaning that the OSPF process will not form neighbor relationships with neighbors reachable on that interface.
passive-interface *default*	OSPF subcommand that changes the OSPF default for interfaces to be passive instead of active (not passive).
no passive-interface *type number*	OSPF subcommand that tells OSPF to be active (not passive) on that interface or subinterface.
default-information originate [**always**]	OSPF subcommand to tell OSPF to create and advertise an OSPF default route, as long as the router has some default route (or to always advertise a default, if the **always** option is configured).

Table 20-7 Chapter 20 EXEC Command Reference

Command	Description
show ip ospf	Lists information about the OSPF process running on the router, including the OSPF router ID, areas to which the router connects, and the number of interfaces in each area.
show ip ospf interface brief	Lists the interfaces on which the OSPF protocol is enabled (based on the **network** commands), including passive interfaces.
show ip ospf interface [*type number*]	Lists a long section of settings, status, and counters for OSPF operation on all interfaces, or on the listed interface, including the Hello and Dead Timers.
show ip protocols	Shows routing protocol parameters and current timer values.

Command	Description
show ip ospf neighbor [*type number*]	Lists brief output about neighbors, identified by neighbor router ID, including current state, with one line per neighbor; optionally, limits the output to neighbors on the listed interface.
show ip ospf neighbor *neighbor-ID*	Lists the same output as the **show ip ospf neighbor** detail command, but only for the listed neighbor (by neighbor RID).
show ip ospf database	Lists a summary of the LSAs in the database, with one line of output per LSA. It is organized by LSA type (first type 1, then type 2, and so on).
show ip route	Lists all IPv4 routes.
show ip route ospf	Lists routes in the routing table learned by OSPF.
show ip route *ip-address mask*	Shows a detailed description of the route for the listed subnet/mask.
clear ip ospf process	Resets the OSPF process, resetting all neighbor relationships and also causing the process to make a choice of OSPF RID.

CHAPTER 21

OSPF Network Types and Neighbors

This chapter covers the following exam topics:

3.0 IP Connectivity

3.4 Configure and verify single area OSPFv2

3.4.a Neighbor adjacencies

3.4.b Point-to-point

3.4.c Broadcast (DR/BDR selection)

3.4.d Router ID

Chapter 20, "Implementing OSPF," discussed the required and most common optional OSPF configuration settings, along with the many verification commands to show how OSPF works with those settings. This chapter continues with more OSPF implementation topics, both to round out the discussion of OSPF and to focus even more on the specific CCNA 200-301 exam topics.

The first of two major sections of this chapter focuses on OSPF network types, specifically types point-to-point and broadcast. The CCNA 200-301 exam topics mention those by name. Chapter 20 showed how OSPF operates on Ethernet interfaces when using their default network type (broadcast). This first section of the chapter discusses the meaning of OSPF network types, default settings, how to configure to use other settings, and how OSPF works differently with different settings.

The second major section then focuses on neighbors and neighbor adjacencies as mentioned in yet another of the OSPF exam topics. OSPF routers cannot exchange LSAs with another router unless they first become neighbors. This second section discusses the various OSPF features that can prevent OSPF routers from becoming neighbors and how you can go about discovering if those bad conditions exist—even if you do not have access to the running configuration.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 21-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
OSPF Network Types	1–3
OSPF Neighbor Relationships	4–6

1. Routers R1 and R2, with router IDs 1.1.1.1 and 2.2.2.2, connect over an Ethernet WAN link. If using all default OSPF settings, if the WAN link initializes for both routers at the same time, which of the following answers are true? (Choose two answers.)
   a. Router R1 will become the DR.
   b. Router R1 will dynamically discover the existence of router R2.
   c. Router R2 will be neither the DR nor the BDR.
   d. Router R1's **show ip ospf neighbor** command will list R2 with a state of "FULL/DR."
2. Routers R1 and R2, with router IDs 1.1.1.1 and 2.2.2.2, connect over an Ethernet WAN link. The configuration uses all defaults, except giving R1 an interface priority of 11 and changing both routers to use OSPF network type point-to-point. If the WAN link initializes for both routers at the same time, which of the following answers are true? (Choose two answers.)
   a. Router R1 will become the DR.
   b. Router R1 will dynamically discover the existence of router R2.
   c. Router R2 will be neither the DR nor the BDR.
   d. Router R2's **show ip ospf neighbor** command will list R1 with a state of "FULL/DR."
3. Per the command output, with how many routers is router R9 full adjacent over its Gi0/0 interface?

```
R9# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/0        1     0               10.1.1.1/24        1     DROTH 2/5
```

   a. 7
   b. 0
   c. 5
   d. 2

4. An engineer connects routers R11 and R12 to the same Ethernet LAN and configures them to use OSPFv2. Which answers describe a combination of settings that would prevent the two routers from becoming OSPF neighbors? (Choose two answers.)
   a. R11's interface uses area 11 while R12's interface uses area 12.
   b. R11's OSPF process uses process ID 11 while R12 uses process ID 12.
   c. R11's interface uses OSPF priority 11 while R12's uses OSPF priority 12.
   d. R11's interface uses an OSPF Hello timer value of 11 while R12's uses 12.
5. An engineer connects routers R13 and R14 to the same Ethernet LAN and configures them to use OSPFv2. Which answers describe a combination of settings that would prevent the two routers from becoming OSPF neighbors?
   a. Both routers' interface IP addresses reside in the same subnet.
   b. Both routers' OSPF process uses process ID 13.
   c. Both routers' OSPF process uses router ID 13.13.13.13.
   d. Both routers' interfaces use an OSPF Dead interval of 40.
6. Router R15 has been a working part of a network that uses OSPFv2. An engineer then issues the **shutdown** command in OSPF configuration mode on R15. Which of the following occurs?
   a. R15 empties its IP routing table of all OSPF routes but keeps its LSDB intact.
   b. R15 empties its LSDB but keeps OSPF neighbor relationships active.
   c. R15 keeps OSPF neighbors open but does not accept new OSPF neighbors.
   d. R15 keeps all OSPF configuration but ceases all OSPF activities (routes, LSDB, neighbors).

Foundation Topics

OSPF Network Types

Two CCNA 200-301 exam topics might be completely misunderstood without taking a closer look at yet more default OSPF settings. In particular, the following exam topics refer to a specific per-interface OSPF setting called the *network type*—even listing the keywords used to configure the setting in the exam topics:

3.4.b: **point-to-point**

3.4.c: **broadcast** (DR/BDR selection)

OSPF includes a small number of network types as a setting on each OSPF-enabled interface. The setting tells the router whether or not to dynamically discover OSPF neighbors (versus requiring the static configuration of the neighboring router's IP address) and whether or not the router should attempt to use a designated router (DR) and backup DR (BDR) in the subnet. Of the two OSPF network types included in the CCNA exam topics, both cause routers to dynamically discover neighbors, but one calls for the use of a DR while the other does not. Table 21-2 summarizes the features of the two OSPF network types mentioned in the exam topics.

Key Topic

Table 21-2 Two OSPF Network Types and Key Behaviors

Network Type Keyword	Dynamically Discovers Neighbors	Uses a DR/BDR
broadcast	Yes	Yes
point-to-point	Yes	No

The rest of this first major section of the chapter explores each type.

The OSPF Broadcast Network Type

OSPF defaults to use a *broadcast* network type on all types of Ethernet interfaces. Note that all the Ethernet interfaces in examples in Chapter 20 relied on that default setting.

To see all the details of how the OSPF broadcast network type works, this chapter begins with a different design than the examples in Chapter 20, instead using a single area design that connects four routers to the same subnet, as shown in Figure 21-1. All links reside in area 0, making the design a single area design.

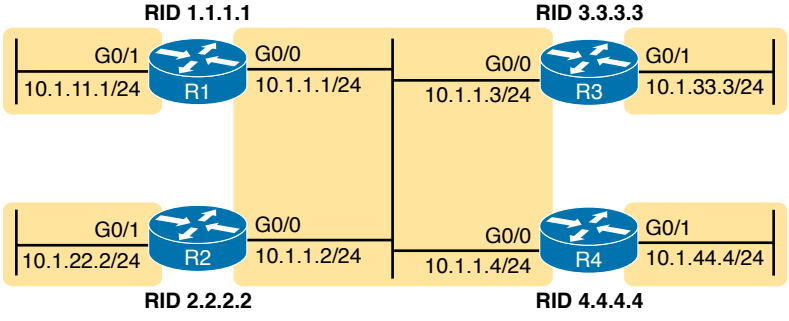


Figure 21-1 *The Single Area Design Used in This Chapter*

To get a sense for how OSPF operates with the broadcast network type, imagine that all four routers use a straightforward OSPF interface configuration like the router R1 configuration shown in Example 21-1. Both GigabitEthernet interfaces on all four routers default to use network type broadcast. Note that the configuration on routers R2, R3, and R4 mirrors R1's configuration except that they use router IDs 2.2.2.2, 3.3.3.3, and 4.4.4.4, respectively, and they use the IP addresses shown in the figure.

Example 21-1 *R1 OSPF Configuration to Match Figure 21-1*

```
router ospf 1
 router-id 1.1.1.1
!
interface gigabitEthernet0/0
 ip ospf 1 area 0
!
interface gigabitEthernet0/1
 ip ospf 1 area 0
```

This simple design gives us a great backdrop from which to observe the results of the broadcast network type on each router. Both interfaces (G0/0 and G0/1) on each router use the broadcast network type and perform the following actions:

- Attempt to discover neighbors by sending OSPF Hellos to the 224.0.0.5 multicast address (an address reserved for sending packets to all OSPF routers in the subnet)
- Attempt to elect a DR and BDR on each subnet
- On the interface with no other routers on the subnet (G0/1), become the DR
- On the interface with three other routers on the subnet (G0/0), be either DR, BDR, or a DROther router
- When sending OSPF messages to the DR or BDR, send the messages to the all-OSPF-DRs multicast address 224.0.0.6

Example 21-2 shows some of the results using the **show ip ospf neighbor** command. Note that R1 lists R2, R3, and R4 as neighbors (based on their 2.2.2.2, 3.3.3.3, and 4.4.4.4 router IDs), confirming that R1 dynamically discovered the other routers. Also, note that the output lists 4.4.4.4 as the DR and 3.3.3.3 as the BDR.

Example 21-2 *R1's List of Neighbors*

```
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   2WAY/DROTHER    00:00:35    10.1.1.2        GigabitEthernet0/0
3.3.3.3           1   FULL/BDR        00:00:33    10.1.1.3        GigabitEthernet0/0
4.4.4.4           1   FULL/DR         00:00:35    10.1.1.4        GigabitEthernet0/0
```

Verifying Operations with Network Type Broadcast

As discussed in the section "Using Designated Routers on Ethernet Links" in Chapter 19, "Understanding OSPF Concepts," all discovered routers on the link should become neighbors and at least reach the *2-way* state. For all neighbor relationships that include the DR and/or BDR, the neighbor relationship should further reach the *full* state. That section defined the term *fully adjacent* as a special term that refers to neighbors that reach this full state.

The design in Figure 21-1, with four routers on the same LAN, provides just enough routers so that one neighbor relationship will remain in a 2-way state and not reach the full state, as a perfectly normal way for OSPF to operate. Figure 21-2 shows the current conditions when the **show** commands in this chapter were gathered, with R4 as the DR, R3 as the BDR, and with R1 and R2 as DROther routers.

Now consider router R1's neighbors as listed in Example 21-2. R1 has three neighbors, all reachable out its G0/0 interface. However, R1's **show ip ospf neighbor** command refers to the state of R1's relationship with the neighbor: 2-way with router 2.2.2.2. Because both R1 and R2 currently serve as DROther routers—that is, they wait ready to become the BDR if either the DR or BDR fails—their neighbor relationship remains in a 2-way state.

Answers to the "Do I Know This Already?" quiz:

1 B, D **2** B, C **3** D **4** A, D **5** C **6** D

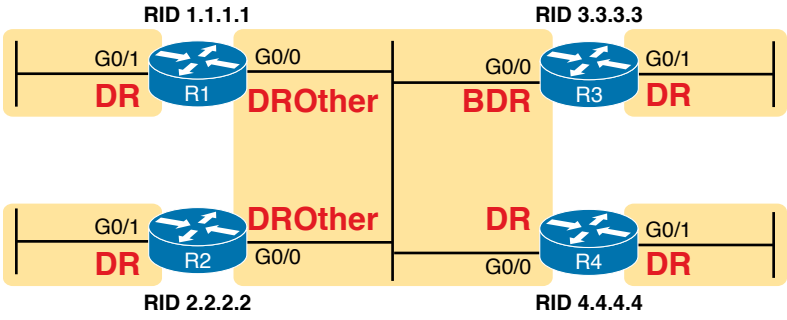


Figure 21-2 *OSPF DR/BDR/DROther Roles in the Network*

Examining Example 21-2 one last time, R1, as a DROther router itself, has two neighbor relationships that reach a full state: R1's neighbor adjacency with DR R4 and R1's neighbor adjacency with BDR R3. But R1 has a total of three neighbors, all reachable off R1's G0/0 interface.

The idea that R1 has three neighbors off its G0/0 interface, with two being fully adjacent, is reflected on the far right of the output of the **show ip ospf interface brief** command output in Example 21-3. It shows "2/3," meaning two neighbors in the full state off port G0/0, with three total neighbors on that interface. Also, note that this command's "State" column differs from the **show ip ospf neighbor** commands, in that the **show ip ospf interface brief** command lists the local router's role on the interface (as shown in Figure 21-2), with R1's G0/1 acting as DR and R1's G0/0 acting as a DROther router.

Key Topic

Example 21-3 *Router R1 OSPF Interfaces: Local Role and Neighbor Counts*

```
R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1     0               10.1.11.1/24       1     DR    0/0
Gi0/0        1     0               10.1.1.1/24        1     DROTH 2/3
```

So far, this topic has described the effect of the OSPF broadcast network type by taking advantage of the default setting on Ethernet interfaces. To see the setting, use the **show ip ospf interface** command, as shown in Example 21-4. The first highlighted item identifies the network type. However, this command's output restates many of the facts seen in both the **show ip ospf neighbor** and **show ip ospf interface brief** commands in Examples 21-2 and 21-3, so take the time to browse through all of Example 21-4 and focus on the additional highlights to see those familiar items.

Example 21-4 *Displaying OSPF Network Type Broadcast*

```
R1# show ip ospf interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0, Attached via Interface Enable
  Process ID 1, Router ID 1.1.1.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Enabled by interface config, including secondary ip addresses
```

```
  Transmit Delay is 1 sec, State DROTHER, Priority 1
  Designated Router (ID) 4.4.4.4, Interface address 10.1.1.4
  Backup Designated router (ID) 3.3.3.3, Interface address 10.1.1.3
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:00
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 3, Adjacent neighbor count is 2
    Adjacent with neighbor 3.3.3.3  (Backup Designated Router)
    Adjacent with neighbor 4.4.4.4  (Designated Router)
  Suppress hello for 0 neighbor(s)
```

Although you would not need to configure an Ethernet interface to use the broadcast network type, some older types of interfaces over the years have used different defaults and with the option to use the broadcast network type. In those cases, the **ip ospf network broadcast** interface subcommand would configure the setting.

Configuring to Influence the DR/BDR Election

In some cases, you may want to influence the OSPF DR election. However, before deciding that makes sense in every case, note that OSPF DR/BDR election rules will not result in a specific router always being the DR, and another always being the BDR, assuming that each is up and working. In short, here are the rules once a DR and BDR have been elected:

- If the DR fails, the BDR becomes the DR, and a new BDR is elected.
- When a better router enters the subnet, no preemption of the existing DR or BDR occurs.

As a result of these rules, while you can configure a router to be the best (highest priority) router to become the DR in an election, doing so only increases that router's statistical chances of being the DR at a given point in time. If the router fails, other routers will become DR and BDR, and the best router will not be DR again until the current DR and BDR fail.

NOTE If you have begun to think about STP elections, note that the rules are similar, but with two key differences. STP uses a lowest-is-best approach and allows new switches to preempt the existing root switch to become the root. OSPF uses a highest-is-best approach and does not preempt the DR as just noted.

In some cases, you may want to influence the DR/BDR election with two configurable settings, listed here in order of precedence:

- **The highest OSPF interface priority:** The highest value wins during an election, with values ranging from 0 to 255.
- **The highest OSPF Router ID:** If the priority ties, the election chooses the router with the highest OSPF RID.

For example, imagine all four routers in the design shown in Figure 21-1 trying to elect the DR and BDR at the same time—for instance, after a power hit in which all four routers power off and back on again. They all participate in the election. They all tie with default priority values of 1 (see Example 21-4 for R1's priority in the **show ip ospf interface** command output.) In this case, R4, with the numerically highest RID of 4.4.4.4, wins the election, and R3, with the next highest RID of 3.3.3.3, becomes the BDR.

To influence the election, you could set the various RIDs with your preferred router with the highest RID value. However, many networks choose OSPF router IDs to help identify the router easily. Instead, using the OSPF priority setting makes better sense. For instance, if an engineer preferred that R1 be the DR, the engineer could add the configuration in Example 21-5 to set R1's interface priority to 99.

Example 21-5 *Influencing DR/BDR Election Using OSPF Priority*

```
R1# configure terminal
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# interface g0/0
R1(config-if)# ip ospf priority 99
R1(config-if)# ^Z
R1#
R1# show ip ospf interface g0/0 | include Priority
  Transmit Delay is 1 sec, State DROTHER, Priority 99

R1# show ip ospf neighbor
Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   2WAY/DROTHER    00:00:36    10.1.1.2        GigabitEthernet0/0
3.3.3.3           1   FULL/BDR        00:00:30    10.1.1.3        GigabitEthernet0/0
4.4.4.4           1   FULL/DR         00:00:37    10.1.1.4        GigabitEthernet0/0

R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1     0               10.1.11.1/24       1     DR    0/0
Gi0/0        1     0               10.1.1.1/24        1     DROTH 2/3
```

The configuration shows R1's interface priority value now as 99, and the **show ip ospf interface G0/0** command that follows confirms the setting. However, the last two commands in the example seem to show that the DR and BDR have not changed at all—and that output is indeed correct. In the example, note that the **show ip ospf neighbor** command still lists R4's state as DR, meaning R4 still acts as the DR, while the **show ip ospf interface brief** command lists R1's State (role) as DROTH.

Just to complete the process, Example 21-6 shows the results after forcing a free election (by failing the LAN switch that sits between the four routers). As expected, R1 wins and becomes DR due to its higher priority, with the other three routers tying based on priority. R4 wins between R2, R3, and R4 due to its higher RID to become the BDR.

Example 21-6 *Results of a Completely New DR/BDR Election*

```
! Not shown: LAN fails, and then recovers, causing a new OSPF Election
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   FULL/DROTHER    00:00:37    10.1.1.2        GigabitEthernet0/0
3.3.3.3           1   FULL/DROTHER    00:00:38    10.1.1.3        GigabitEthernet0/0
4.4.4.4           1   FULL/BDR        00:00:38    10.1.1.4        GigabitEthernet0/0

R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1     0               10.1.11.1/24       1     DR    0/0
Gi0/0        1     0               10.1.1.1/24        1     DR    3/3
```

The OSPF Point-to-Point Network Type

The other OSPF network type mentioned in the CCNA 200-301 exam topics, point-to-point, works well for data links that by their nature have just two routers on the link. For example, consider the topology in Figure 21-3, which shows router R1 with three WAN links—two Ethernet WAN links and one serial link.

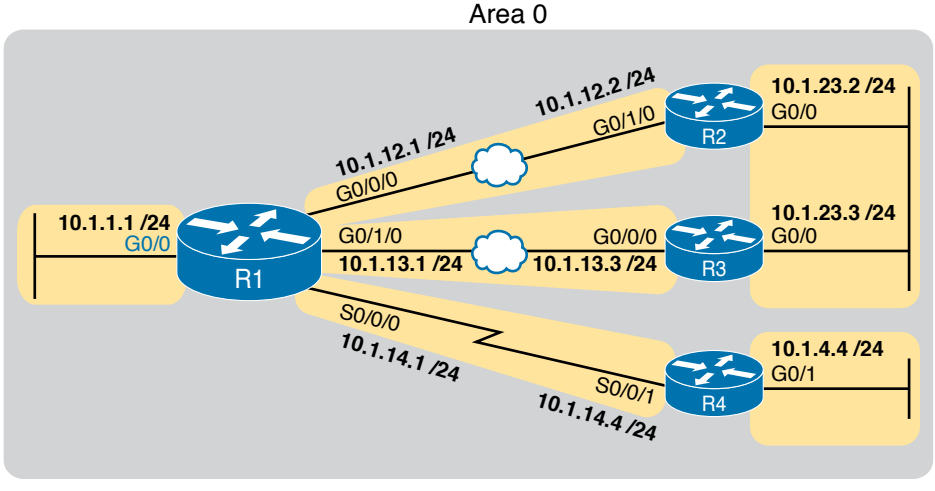


Figure 21-3 *Sample OSPF Design with Serial and Ethernet WAN*

First, focus on the serial link itself. To review, the jagged line represents a physical link that can at most have two devices using the link, specifically R1 and R4 in this case. The link does not support the ability to add a third router to the link. As you might guess, the data-link protocols to control a link with at most two devices can work differently than Ethernet.

For instance, the data-link protocols most often used on the link (HDLC and PPP) do not support data-link broadcasts.

Next, consider the OSPF point-to-point network type: it exists for serial links and other links that use a point-to-point topology. These links often do not support data-link broadcasts. Additionally, with only two devices on the link, using a DR/BDR is not a help, and it actually adds a little extra convergence time. Using a network type of point-to-point tells the router to not use a DR/BDR on the link.

While you may see some serial links in networks today, the CCNA and CCNP Enterprise exams make no specific mention of serial technology at this point. However, you will see other point-to-point links—like some Ethernet WAN links.

To connect the thoughts, note that all the Ethernet WAN links used in this book happen to use a point-to-point Ethernet WAN service called an Ethernet Private Wire Service or simply an Ethernet Line (E-Line). For that service, the service provider will send Ethernet frames between two devices (routers) connected to the service, but only those two devices. In other words, an E-line is a point-to-point service in concept. So while the Ethernet data-link protocol supports broadcast frames, only two devices can exist on the link, and there is no advantage to using a DR/BDR. As a result, many engineers prefer to instead use an OSPF point-to-point network type on Ethernet WAN links that in effect act as a point-to-point link.

Example 21-7 shows the configuration of router R1's G0/0/0 interface in Figure 21-3 to use OSPF network type point-to-point. R2, on the other end of the WAN link, would need the same configuration command on its matching interface.

Example 21-7 *OSPF Network Type Point-to-Point on an Ethernet WAN Interface on R1*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# interface g0/0/0
R1(config-if)# ip ospf network point-to-point
R1(config-if)#

R1# show ip ospf interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Internet Address 10.1.12.1/24, Area 0, Attached via Interface Enable
  Process ID 1, Router ID 1.1.1.1, Network Type POINT_TO_POINT, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           4         no          no            Base
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT_TO_POINT
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:01
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/3/3, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
```

```
Last flood scan length is 1, maximum is 3
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 1, Adjacent neighbor count is 1
  Adjacent with neighbor 2.2.2.2
Suppress hello for 0 neighbor(s)
```

Note the highlighted portions of the **show** command in Example 21-6. The first two highlights note the network type. The final highlight with two lines notes that R1 has one neighbor on the interface, a neighbor with which it has become fully adjacent per the output.

Example 21-8 closes this section with a confirmation of some of those facts with two more commands. Note that the **show ip ospf neighbor** command on R1 lists router R2 (RID 2.2.2.2) with a full state, but with no DR nor BDR designation, instead listing a -. The - acts as a reminder that the link does not use a DR/BDR. The second command, **show ip ospf interface brief**, shows the state (the local router's role) as P2P, which is short for point-to-point, with a counter of 1 for the number of fully adjacent neighbors and total number of neighbors.

Key Topic

Example 21-8 *OSPF Network Type Point-to-Point on an Ethernet WAN Interface on R1*

```
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           0   FULL/  -        00:00:39    10.1.12.2       GigabitEthernet0/0/0
! lines omitted for brevity

R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/0/0      1     0               10.1.12.1/24       4     P2P   1/1
! lines omitted for brevity
```

When using Ethernet WAN links that behave as a point-to-point link, consider using OSPF network type point-to-point rather than using the default broadcast type.

OSPF Neighbor Relationships

A router's OSPF configuration enables OSPF on a set of interfaces. IOS then attempts to discover other neighbors on those interfaces by sending and listening for OSPF Hello messages. However, once discovered, two routers may not become neighbors. They must have compatible values for several settings as listed in the Hellos exchanged between the two routers. This second major section of the chapter examines those reasons.

OSPF Neighbor Requirements

After an OSPF router hears a Hello from a new neighbor, the routing protocol examines the information in the Hello and compares that information with the local router's own settings. If the settings match, great. If not, the routers do not become neighbors. Because there is no formal term for all these items that a routing protocol considers, this book just calls them *neighbor requirements*. Table 21-3 lists the neighbor requirements for OSPF, with some comments about the various issues following the table.

Key Topic

Table 21-3 Neighbor Requirements for OSPF

Requirement	Required for OSPF	Neighbor Missing if Incorrect
Interfaces must be in an up/up state.	Yes	Yes
Access control lists (ACL) must not filter routing protocol messages.	Yes	Yes
Interfaces must be in the same subnet.	Yes	Yes
They must pass routing protocol neighbor authentication (if configured).	Yes	Yes
Hello and hold/dead timers must match.	Yes	Yes
Router IDs (RID) must be unique.	Yes	Yes
They must be in the same area.	Yes	Yes
OSPF process must not be shut down.	Yes	Yes
Neighboring interfaces must use same MTU setting.	Yes	No
Neighboring interfaces must use same OSPF network type.	Yes	No

First, consider the meaning of the two rightmost columns. The column labeled "Required for OSPF" means that the item must be working correctly for the neighbor relationship to work correctly. Note that all the items in this column list a "yes," meaning that all must be correct for the neighbor relationship to work correctly. The last column heading states "Neighbor Missing if Incorrect." For items listing a "yes" in this column, if that item is configured incorrectly, the neighbor will not appear in lists of OSPF neighbors—for instance, with the **show ip ospf neighbor** command.

Next, focus on the shaded items at the top of the table. The symptom that occurs if either of these is a problem is that the **show ip ospf neighbor** command would not list the other router. For instance, the first item states that the router interfaces must be up and working. If the router interface is not working, the router cannot send any OSPF messages and discover any OSPF neighbors on that interface.

The middle section of the table (the unshaded rows) focuses on some OSPF settings. These items must be correct, but if not, they also result in the neighbor not being listed in the output of the **show ip ospf neighbor** command.

As you can see, using the **show ip ospf neighbor** command can give you a good starting point to troubleshoot OSPF on the exam and in real life. If you see the neighbor you expect to see, great! If not, the table gives you a good list to use for items to investigate.

Finally, the last section (shaded) lists a couple of OSPF settings that give a different symptom when incorrect. Again, those two items must be correct for OSPF neighbors to work. However, for these two items, when incorrect, a router can list the other router as a neighbor, but the neighbor relationship does not work properly in that the routers do not exchange LSAs as they should.

For reference, Table 21-4 relists some of the requirements from Table 21-3, along with the most useful commands with which to find the answers.

Key Topic

Table 21-4 OSPF Neighbor Requirements and the Best **show/debug** Commands

Requirement	Best show Command
Hello and dead timers must match.	**show ip ospf interface**
They must be in the same area.	**show ip ospf interface brief**
RIDs must be unique.	**show ip ospf**
They must pass any neighbor authentication.	**show ip ospf interface**
OSPF process must not be shut down.	**show ip ospf, show ip ospf interface**

The rest of this section looks at some of the items from Table 21-3 in a little more detail.

NOTE One configuration choice that people sometimes think is an issue, but is not, is the process ID as defined by the **router ospf** *process-id* command. Neighboring routers can use the same process ID values, or different process ID values, with no impact on whether two routers become OSPF neighbors.

Issues That Prevent Neighbor Adjacencies

The next few pages look at three of the topics from Table 21-3 for which, if a problem exists, the router does not become a neighbor (that is, the unshaded parts of the table.). To show the issues, this section uses the same topology shown earlier in Figure 21-1 but now with some incorrect configuration introduced. In other words, the configuration matches Example 21-1 that began this chapter, but with the following errors introduced:

- R2 has been configured with both LAN interfaces in area 1, whereas the other three routers' G0/0 interfaces are assigned to area 0.
- R3 is using the same RID (1.1.1.1) as R1.
- R4 has been configured with a Hello/dead timer of 5/20 on its G0/0 interface, instead of the 10/40 used (by default) on R1, R2, and R3.

Figure 21-4 shows these same problems for reference.

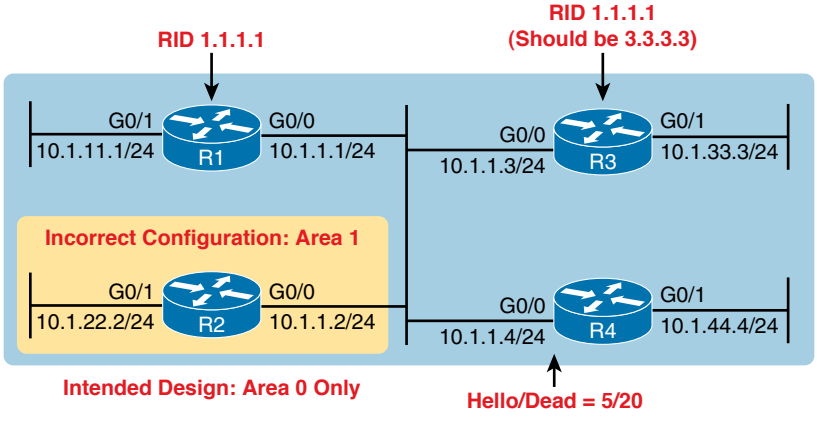


Figure 21-4 *Summary of Problems That Prevent OSPF Neighbors on the Central LAN*

Finding Area Mismatches

To create an area mismatch, the configuration on some router must place the interface into the wrong area per the design. As shown in Figure 21-4, router R2 was configured incorrectly, placing both its interfaces into area 1 instead of area 0. Example 21-9 shows the configuration, which uses the correct syntax (and is therefore accepted by the router) but sets the wrong area number.

Example 21-9 *Setting Area 1 on R2's Interfaces, When They Should Be in Area 0*

```
router ospf 1
 router-id 2.2.2.2
!
interface gigabitEthernet0/0
 ip ospf 1 area 1
!
interface gigabitEthernet0/1
 ip ospf 1 area 1
```

With an area mismatch error, the **show ip ospf neighbor** command will not list the neighbor. Because you see nothing in the OSPF neighbor table, to troubleshoot this problem, you need to find the area configuration on each interface on potentially neighboring routers. To do so:

- Check the output of **show running-config** to look for
  - **ip ospf** *process-id* **area** *area-number* interface subcommands
  - **network** commands in OSPF configuration mode
- Use the **show ip ospf interface [brief]** command to list the area number

Finding Duplicate OSPF Router IDs

Next, Example 21-10 shows R1 and R3 both trying to use RID 1.1.1.1. Interestingly, both routers automatically generate a log message for the duplicate OSPF RID problem between R1 and R3; the end of Example 21-10 shows one such message. For the exams, just use the **show ip ospf** commands on both R3 and R1 to easily list the RID on each router, noting that they both use the same value.

Example 21-10 *Comparing OSPF Router IDs on R1 and R3*

```
! Next, on R3: R3 lists the RID of 1.1.1.1
!
R3# show ip ospf
Routing Process "ospf 1" with ID 1.1.1.1
Start time: 00:00:37.136, Time elapsed: 02:20:37.200
! lines omitted for brevity
```

```
! Back to R1: R1 also uses RID 1.1.1.1
R1# show ip ospf
Routing Process "ospf 1" with ID 1.1.1.1
```

```
Start time: 00:01:51.864, Time elapsed: 12:13:50.904
! lines omitted for brevity

*May 29 00:01:25.679: %OSPF-4-DUP_RTRID_NBR: OSPF detected duplicate router-id
1.1.1.1 from 10.1.1.3 on interface GigabitEthernet0/0
```

First, focus on the problem: the duplicate RIDs. The first line of the **show ip ospf** command on the two routers quickly shows the duplicate use of 1.1.1.1. To solve the problem, assuming R1 should use 1.1.1.1 and R3 should use another RID (maybe 3.3.3.3), change the RID on R3 and restart the OSPF process. To do so, use the **router-id 3.3.3.3** OSPF subcommand and use the EXEC mode command **clear ip ospf process**. (OSPF will not begin using a new RID value until the process restarts, either via command or reload.)

Finding OSPF Hello and Dead Timer Mismatches

First, as a reminder from chapters past:

- **Hello interval/timer:** The per-interface timer that tells a router how often to send OSPF Hello messages on an interface.
- **Dead interval/timer:** The per-interface timer that tells the router how long to wait without having received a Hello from a neighbor before believing that neighbor has failed. (Defaults to four times the Hello timer.)

Next, consider the problem created on R4, with the configuration of a different Hello timer and dead timer (5 and 20, respectively) as compared with the default settings on R1, R2, and R3 (10 and 40, respectively). A Hello or Dead interval mismatch prevents R4 from becoming neighbors with any of the other three OSPF routers. Routers list their Hello and Dead interval settings in their Hello messages and choose to not become neighbors if the values do not match. As a result, none of the routers become neighbors with router R4 in this case.

Example 21-11 shows the easiest way to find the mismatch using the **show ip ospf interface** command on both R1 and R4. This command lists the Hello and dead timers for each interface, as highlighted in the example. Note that R1 uses 10 and 40 (Hello and dead), whereas R4 uses 5 and 20.

Example 21-11 *Finding Mismatched Hello/Dead Timers*

```
R1# show ip ospf interface G0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0, Attached via Network Statement
  Process ID 1, Router ID 1.1.1.1, Network Type BROADCAST, Cost: 1
  Topology-MTID  Cost  Disabled  Shutdown   Topology Name
        0         1       no        no         Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 1.1.1.1, Interface address 10.1.1.1
  No backup designated router on this network
```

```
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
! lines omitted for brevity
```

```
! Moving on to R4 next
!
R4# show ip ospf interface Gi0/0
GigabitEthernet0/0 is up, line protocol is up
   Internet Address 10.1.1.4/24, Area 0, Attached via Network Statement
   Process ID 4, Router ID 10.1.44.4, Network Type BROADCAST, Cost: 1
  Topology-MTID  Cost  Disabled  Shutdown   Topology Name
        0         1       no        no         Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.1.44.4, Interface address 10.1.1.4
  No backup designated router on this network
  Timer intervals configured, Hello 5, Dead 20, Wait 20, Retransmit 5
! lines omitted for brevity
```

Shutting Down the OSPF Process

Similar to administratively disabling and enabling an interface, IOS also allows the OSPFv2 routing protocol process to be disabled and enabled with the **shutdown** and **no shutdown** router mode subcommands, respectively. When a routing protocol process is shut down, IOS does the following:

- Brings down all neighbor relationships and clears the OSPF neighbor table
- Clears the LSDB
- Clears the IP routing table of any OSPF-learned routes

At the same time, shutting down OSPF does retain some important details about OSPF, in particular:

- IOS retains all OSPF configuration.
- IOS still lists all OSPF-enabled interfaces in the OSPF interface list (**show ip ospf interface**) but in a DOWN state.

Basically, shutting down the OSPF routing protocol process gives the network engineer a way to stop using the routing protocol on that router without having to remove all the configuration. Once shut down, the **show ip ospf interface [brief]** command should still list some output, as will the **show ip ospf** command, but the rest of the commands will list nothing.

Example 21-12 shows an example on Router R5, as shown in Figure 21-5. R5 is a different router than the one used in earlier examples, but it begins the example with two OSPF neighbors, R2 and R3, with router IDs 2.2.2.2 and 3.3.3.3. The example shows the OSPF process being shut down, the neighbors failing, and those two key OSPF **show** commands: **show ip ospf neighbor** and **show ip ospf interface brief**.

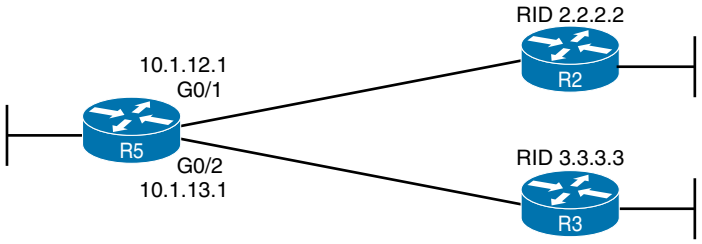


Figure 21-5 *Example Network to Demonstrate OSPF Process Shutdown*

Example 21-12 *Shutting Down an OSPF Process, and the Resulting Neighbor States*

```
R5# show ip ospf neighbor
Neighbor ID     Pri   State           Dead Time    Address         Interface
2.2.2.2           1   FULL/DR         00:00:35     10.1.12.2       GigabitEthernet0/1
3.3.3.3           1   FULL/DR         00:00:33     10.1.13.3       GigabitEthernet0/2
R5# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R5(config)# router ospf 1
R5(config-router)# shutdown
R5(config-router)# ^Z
*Mar 23 12:43:30.634: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on GigabitEthernet0/1
  from FULL to DOWN, Neighbor Down: Interface down or detached
*Mar 23 12:43:30.635: %OSPF-5-ADJCHG: Process 1, Nbr 3.3.3.3 on GigabitEthernet0/2
  from FULL to DOWN, Neighbor Down: Interface down or detached
R5# show ip ospf interface brief
Interface    PID     Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1       0               10.1.12.1/24       1     DOWN  0/0
Gi0/2        1       0               10.1.13.1/24       1     DOWN  0/0

R5# show ip ospf
 Routing Process "ospf 1" with ID 5.5.5.5
 Start time: 5d23h, Time elapsed: 1d04h
 Routing Process is shutdown
! lines omitted for brevity

R5# show ip ospf neighbor
R5#
R5# show ip ospf database
            OSPF Router with ID (3.3.3.3) (Process ID 1)
R5#
```

First, before the **shutdown**, the **show ip ospf neighbor** command lists two neighbors. After the **shutdown**, the same command lists no neighbors at all. Second, the **show ip ospf interface brief** command does list the interfaces on which OSPF is enabled, on the local router's own IP addresses. However, it lists a state of DOWN, which is a reference to the local router's state. Also, note that the **show ip ospf** command positively states that the

OSPF process is in a shutdown state, while the **show ip ospf database** command output lists only a heading line, with no LSAs.

Issues That Allow Adjacencies but Prevent IP Routes

The last two issues to discuss in this section have a symptom in which the **show ip ospf neighbor** command does list a neighbor, but some other problem exists that prevents the eventual addition of OSPF routes to the routing table. The two issues: a mismatched MTU setting and a mismatched OSPF network type.

Mismatched MTU Settings

The MTU size defines a per-interface setting used by the router for its Layer 3 forwarding logic, defining the largest network layer packet that the router will forward out each interface. For instance, the IPv4 MTU size of an interface defines the maximum size IPv4 packet that the router can forward out an interface.

Routers often use a default MTU size of 1500 bytes, with the ability to set the value as well. The **ip mtu** *size* interface subcommand defines the IPv4 MTU setting, and the **ipv6 mtu** *size* command sets the equivalent for IPv6 packets.

In an odd twist, two OSPFv2 routers can actually become OSPF neighbors, be listed in the output of the **show ip ospf neighbor** command, and reach 2-way state, even if they happen to use different IPv4 MTU settings on their interfaces. However, they fail to exchange their LSDBs. Eventually, after trying and failing to exchange their LSDBs, the neighbor relationship also fails. So also keep a watch for MTU mismatches, although they may be unusual and obscure, by looking at the running-config and by using the **show interfaces** command (which lists the IP MTU).

Mismatched OSPF Network Types

Earlier in this chapter you read about the OSPF broadcast network type, which uses a DR/BDR, and the OSPF point-to-point network type, which does not. Interestingly, if you misconfigure network type settings such that one router uses broadcast, and the other uses point-to-point, the following occurs:

- The two routers become fully adjacent neighbors (that is, they reach a full state).
- They exchange their LSDBs.
- They do not add IP routes to the IP routing table.

The reason for not adding the routes has to do with the details of LSAs and how the use of a DR (or not) changes those LSAs. Basically, the two routers expect different details in the LSAs, and the SPF algorithm notices those differences and cannot trust the LSAs because of those differences.

For instance, earlier in Example 21-7, the configuration showed router R1 using network type point-to-point on its G0/0/0 interface, with the expectation that router R2 would also use point-to-point on its matching G0/1/0 interface. Example 21-13 shows some of the results if the engineer neglected to configure R2, leaving it with the default setting of broadcast.

Example 21-13 *Shutting Down an OSPF Process, and the Resulting Neighbor States*

```
*Apr 10 16:31:01.951: %OSPF-4-NET_TYPE_MISMATCH: Received Hello from 2.2.2.2 on
  GigabitEthernet0/0/0 indicating a potential network type mismatch
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           0   FULL/  -        00:00:38    10.1.12.2       GigabitEthernet0/0/0
R1#
```

```
R2# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           1   FULL/BDR        00:00:30    10.1.12.1       GigabitEthernet0/1/0
```

As you can see, both routers list the other as an OSPF neighbor in the full state. However, R1, with network type point-to-point, does not list a DR or BDR role in the output, while R2 does, which is one clue for this type of problem. The other comes with noticing that the expected routes are not in the IP routing table.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 21-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 21-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used:
Review key topics		Book, website
Review command tables		Book
Review memory tables		Website
Watch video		Website

Review All the Key Topics

Key Topic

Table 21-6 Key Topics for Chapter 21

Key Topic Element	Description	Page Number
Table 21-2	Two OSPF Network Types and Key Behaviors	501
Example 21-3	OSPF interfaces, local roles, and neighbor counts	503
List	Rules for electing an OSPF DR/BDR	505
Example 21-8	Evidences of OSPF network type point-to-point	508

Key Topic Element	Description	Page Number
Table 21-3	Neighbor requirements for OSPF	509
Table 21-4	Useful commands to discover OSPF neighbor issues	510
List	Symptoms of an OSPF network type mismatch	515

Command References

Tables 21-7 and 21-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 21-7 Chapter 21 Configuration Command Reference

Command	Description
ip ospf hello-interval *seconds*	Interface subcommand that sets the interval for periodic Hellos
ip ospf dead-interval *number*	Interface subcommand that sets the OSPF dead timer
passive-interface *type number*	Router subcommand, for both OSPF and EIGRP, that tells the routing protocol to stop sending Hellos and stop trying to discover neighbors on that interface
ip ospf priority *value*	Interface subcommand that sets the OSPF priority, used when electing a new DR or BDR
ip ospf network {broadcast | point-to-point}	Interface subcommand used to set the OSPF network type on the interface
[no] shutdown	An OSPF configuration mode command to disable (**shutdown**) or enable (**no shutdown**) the OSPF process

Table 21-8 Chapter 21 **show** Command Reference

Command	Description
show ip protocols	Shows routing protocol parameters and current timer values, including an effective copy of the routing protocols' **network** commands and a list of passive interfaces
show ip ospf interface brief	Lists the interfaces on which the OSPF protocol is enabled (based on the **network** commands), including passive interfaces
show ip ospf interface [*type number*]	Lists detailed OSPF settings for all interfaces, or the listed interface, including Hello and dead timers and OSPF area
show ip ospf neighbor	Lists neighbors and current status with neighbors, per interface
show ip ospf	Lists a group of messages about the OSPF process itself, listing the OSPF Router ID in the first line
show interfaces	Lists a long set of messages, per interface, that lists configuration, state, and counter information

Part VI Review

Keep track of your part review progress with the checklist in Table P6-1. Details about each task follow the table.

Table P6-1 Part VI Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Watch Videos		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book using the PTP software. See the section "How to View Only DIKTA Questions by Chapter or Part" in the Introduction to this book to learn how to make the PTP software show you DIKTA questions for this part only.

Answer Part Review Questions

For this task, answer the Part Review questions for this part of the book using the PTP software. See the section "How to View Part Review Questions" in the Introduction to this book to learn how to make the PTP software show you Part Review questions for this part only.

Review Key Topics

Review all Key Topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Do Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson ICND1 or CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog: Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book, as found at http://blog.certskills.com. Then navigate to the Hands-on Config labs.

Other: If using other lab tools, here are a few suggestions: Make sure to experiment heavily with VLAN configuration and VLAN trunking configuration.

Watch Videos

Chapter 21 recommends one video from the companion website about troubleshooting OSPF neighbors. Take a few minutes to watch the video if you haven't done so already.

So far, this book has mostly ignored IP version 6 (IPv6). This part reverses the trend, collecting all the specific IPv6 topics into four chapters.

The chapters in Part VII walk you through the same topics discussed throughout this book for IPv4, often using IPv4 as a point of comparison. Certainly, many details differ when comparing IPv4 and IPv6. However, many core concepts about IP addressing, subnetting, routing, and routing protocols remain the same. The chapters in this part build on those foundational concepts, adding the specific details about how IPv6 forwards IPv6 packets from one host to another.

Part VII

IP Version 6

Chapter 22: Fundamentals of IP Version 6

Chapter 23: IPv6 Addressing and Subnetting

Chapter 24: Implementing IPv6 Addressing on Routers

Chapter 25: Implementing IPv6 Routing

Part VII Review

CHAPTER 22

Fundamentals of IP Version 6

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.8 Configure and verify IPv6 addressing and prefix

IPv4 has been a solid and highly useful part of the growth of TCP/IP and the Internet. For most of the long history of the Internet, and for most corporate networks that use TCP/IP, IPv4 is the core protocol that defines addressing and routing. However, even though IPv4 has many great qualities, it does have some shortcomings, creating the need for a replacement protocol: IP version 6 (IPv6).

IPv6 defines the same general functions as IPv4, but with different methods of implementing those functions. For example, both IPv4 and IPv6 define addressing, the concepts of subnetting larger groups of addresses into smaller groups, headers used to create an IPv4 or IPv6 packet, and the rules for routing those packets. At the same time, IPv6 handles the details differently; for example, using a 128-bit IPv6 address rather than the 32-bit IPv4 address.

This chapter focuses on the core network layer functions of addressing and routing. The first section of this chapter looks at the big concepts, while the second section looks at the specifics of how to write and type IPv6 addresses.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 22-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Introduction to IPv6	1–2
IPv6 Addressing Formats and Conventions	3–6

1. Which of the following was a short-term solution to the IPv4 address exhaustion problem?
   a. IP version 6
   b. IP version 5
   c. NAT/PAT
   d. ARP

2. A router receives an Ethernet frame that holds an IPv6 packet. The router then makes a decision to route the packet out a serial link. Which of the following statements is true about how a router forwards an IPv6 packet?
   a. The router discards the Ethernet data-link header and trailer of the received frame.
   b. The router makes the forwarding decision based on the packet's source IPv6 address.
   c. The router keeps the Ethernet header, encapsulating the entire frame inside a new IPv6 packet before sending it over the serial link.
   d. The router uses the IPv4 routing table when choosing where to forward the packet.
3. Which of the following is the shortest valid abbreviation for FE80:0000:0000:0100:0000:0000:0000:0123?
   a. FE80::100::123
   b. FE8::1::123
   c. FE80::100:0:0:0:123:4567
   d. FE80:0:0:100::123
4. Which of the following is the shortest valid abbreviation for 2000:0300:0040:0005:6000:0700:0080:0009?
   a. 2:3:4:5:6:7:8:9
   b. 2000:300:40:5:6000:700:80:9
   c. 2000:300:4:5:6000:700:8:9
   d. 2000:3:4:5:6:7:8:9
5. Which of the following is the unabbreviated version of IPv6 address 2001:DB8::200:28?
   a. 2001:0DB8:0000:0000:0000:0000:0200:0028
   b. 2001:0DB8::0200:0028
   c. 2001:0DB8:0:0:0:0:0200:0028
   d. 2001:0DB8:0000:0000:0000:0000:200:0028
6. Which of the following is the prefix for address 2000:0000:0000:0005:6000:0700:0080:0009, assuming a mask of /64?
   a. 2000::5::/64
   b. 2000::5:0:0:0:0/64
   c. 2000:0:0:5::/64
   d. 2000:0:0:5:0:0:0:0/64

Foundation Topics

Introduction to IPv6

IP version 6 (IPv6) serves as the replacement protocol for IP version 4 (IPv4).

Unfortunately, that one bold statement creates more questions than it answers. Why does IPv4 need to be replaced? If IPv4 needs to be replaced, when will that happen—and will it happen quickly? What exactly happens when a company or the Internet replaces IPv4 with IPv6? And the list goes on.

While this introductory chapter cannot get into every detail of why IPv4 needs to eventually be replaced by IPv6, the clearest and most obvious reason for migrating TCP/IP networks to use IPv6 is growth. IPv4 uses a 32-bit address, which totals to a few billion addresses. Interestingly, that seemingly large number of addresses is too small. IPv6 increases the address to 128 bits in length. For perspective, IPv6 supplies more than 10,000,000,000,000,000,000,000,000,000,000 times as many addresses as IPv4.

The fact that IPv6 uses a different size address field, with some different addressing rules, means that many other protocols and functions change as well. For example, IPv4 routing—in other words, the packet-forwarding process—relies on an understanding of IPv4 addresses. To support IPv6 routing, routers must understand IPv6 addresses and routing. To dynamically learn routes for IPv6 subnets, routing protocols must support these different IPv6 addressing rules, including rules about how IPv6 creates subnets. As a result, the migration from IPv4 to IPv6 is much more than changing one protocol (IP), but it impacts many protocols.

This first section of the chapter discusses some of the reasons for the change from IPv4 to IPv6, along with the protocols that must change as a result.

The Historical Reasons for IPv6

In the last 40+ years, the Internet has gone from its infancy to being a huge influence in the world. It first grew through research at universities, from the ARPANET beginnings of the Internet in the late 1960s into the 1970s. The Internet kept growing fast in the 1980s, with the Internet's fast growth still primarily driven by research and the universities that joined in that research. By the early 1990s, the Internet began to transform to allow commerce, allowing people to sell services and products over the Internet, which drove yet another steep spike upward in the growth of the Internet. Eventually, fixed Internet access (primarily through dial, digital subscriber line [DSL], and cable) became common, followed by the pervasive use of the Internet from mobile devices like smartphones. Figure 22-1 shows some of these major milestones with general dates.

The incredible growth of the Internet over a fairly long time created a big problem for public IPv4 addresses: the world was running out of addresses. For instance, in 2011, IANA allocated the final /8 address blocks (the same size as a Class A network), allocating one final /8 block to each of the five Regional Internet Registries (RIR). At that point, RIRs could no

---

Answers to the "Do I Know This Already?" quiz:

1 C **2** A **3** D **4** B **5** A **6** C

longer receive new allocations of public addresses from IANA to then turn around and assign smaller address blocks to companies or ISPs.

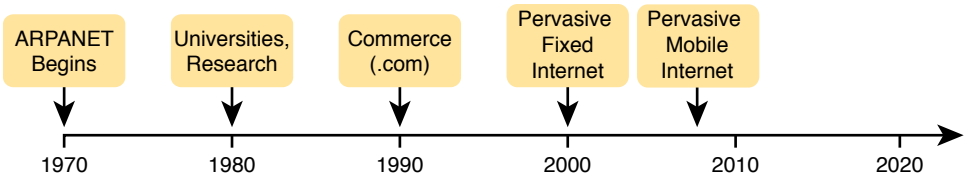


Figure 22-1 *Some Major Events in the Growth of the Internet*

At that point in 2011, each of the five RIRs still had public addresses to allocate or assign. However, that same year, APNIC (Asia Pacific) became the first RIR to exhaust its available IPv4 address allocation. In late 2015, ARIN (North America) announced that it had exhausted its supply. When we were revising this chapter in 2019, IANA considered all RIRs except AFRINIC to have exhausted their supply of IPv4 addresses, with AFRINIC expected to run out of IPv4 address during the year 2019.

These events are significant in that the day has finally come in which new companies can attempt to connect to the Internet, but they can no longer simply use IPv4, ignoring IPv6. Their only option will be IPv6 because IPv4 has no public addresses left.

NOTE You can track ARIN's progress through this interesting transition in the history of the Internet at its IPv4 address depletion site: http://teamarin.net/category/ipv4-depletion/. You can also see a summary report at http://ipv4.potaroo.net.

Even though the press has rightfully made a big deal about running out of IPv4 addresses, those who care about the Internet knew about this potential problem since the late 1980s. The problem, generally called the *IPv4 address exhaustion* problem, could literally have caused the huge growth of the Internet in the 1990s to have come to a screeching halt! Something had to be done.

The IETF came up with several short-term solutions to make IPv4 addresses last longer, and one long-term solution: IPv6. However, several other tools like Network Address Translation (NAT) and classless interdomain routing (CIDR) helped extend IPv4's life another couple of decades. IPv6 creates a more permanent and long-lasting solution, replacing IPv4, with a new IPv6 header and new IPv6 addresses. The address size supports a huge number of addresses, solving the address shortage problem for generations (we hope). Figure 22-2 shows some of the major address exhaustion timing.

The rest of this first section examines IPv6, comparing it to IPv4, focusing on the common features of the two protocols. In particular, this section compares the protocols (including addresses), routing, routing protocols, and miscellaneous other related topics.

NOTE You might wonder why the next version of IP is not called IP version 5. There was an earlier effort to create a new version of IP, and it was numbered version 5. IPv5 did not progress to the standards stage. However, to prevent any issues, because version 5 had been used in some documents, the next effort to update IP was numbered as version 6.

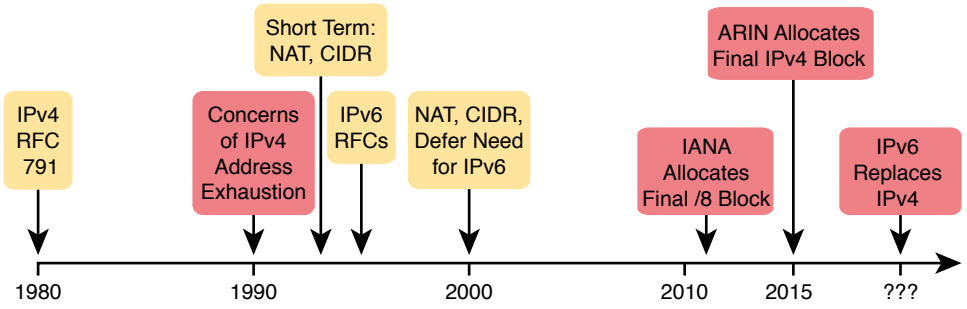


Figure 22-2 *Timeline for IPv4 Address Exhaustion and Short-/Long-Term Solutions*

The IPv6 Protocols

The primary purpose of the core IPv6 protocol mirrors the same purpose of the IPv4 protocol. That core IPv6 protocol, as defined in RFC 2460, defines a packet concept, addresses for those packets, and the role of hosts and routers. These rules allow the devices to forward packets sourced by hosts, through multiple routers, so that they arrive at the correct destination host. (IPv4 defines those same concepts for IPv4 back in RFC 791.)

However, because IPv6 impacts so many other functions in a TCP/IP network, many more RFCs must define details of IPv6. Some other RFCs define how to migrate from IPv4 to IPv6. Others define new versions of familiar protocols or replace old protocols with new ones. For example:

Older OSPF Version 2 Upgraded to OSPF Version 3: The older Open Shortest Path First (OSPF) version 2 works for IPv4, but not for IPv6, so a newer version, OSPF version 3, was created to support IPv6. (Note: OSPFv3 was later upgraded to support advertising both IPv4 and IPv6 routes.)

ICMP Upgraded to ICMP Version 6: Internet Control Message Protocol (ICMP) worked well with IPv4 but needed to be changed to support IPv6. The new name is ICMPv6.

ARP Replaced by Neighbor Discovery Protocol: For IPv4, Address Resolution Protocol (ARP) discovers the MAC address used by neighbors. IPv6 replaces ARP with a more general Neighbor Discovery Protocol (NDP).

NOTE If you go to any website that lists the RFCs, like http://www.rfc-editor.org,you can find almost 300 RFCs that have IPv6 in the title.

Although the term IPv6, when used broadly, includes many protocols, the one specific protocol called IPv6 defines the new 128-bit IPv6 address. Of course, writing these addresses in binary would be a problem—they probably would not even fit on the width of a piece of paper! IPv6 defines a shorter hexadecimal format, requiring at most 32 hexadecimal digits (one hex digit per 4 bits), with methods to abbreviate the hexadecimal addresses as well.

For example, all of the following are IPv6 addresses, each with 32 or fewer hex digits:

```
2345:1111:2222:3333:4444:5555:6666:AAAA
2000:1:2:3:4:5:6:A
FE80::1
```

The upcoming section "IPv6 Addressing Formats and Conventions" discusses the specifics of how to represent IPv6 addresses, including how to legally abbreviate the hex address values.

Like IPv4, IPv6 defines a header, with places to hold both the source and destination address fields. Compared to IPv4, the IPv6 header does make some other changes besides simply making the address fields larger. However, even though the IPv6 header is larger than an IPv4 header, the IPv6 header is actually simpler (on purpose), to reduce the work done each time a router must route an IPv6 packet. Figure 22-3 shows the required 40-byte part of the IPv6 header.

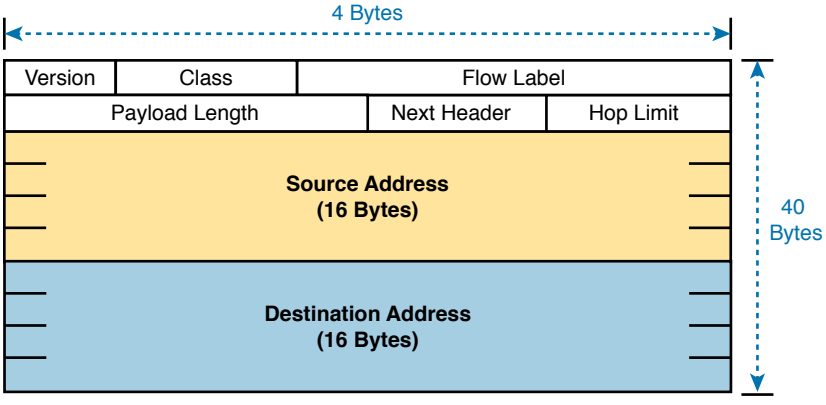


Figure 22-3 *IPv6 Header*

IPv6 Routing

As with many functions of IPv6, IPv6 routing looks just like IPv4 routing from a general perspective, with the differences being clear only once you look at the specifics. Keeping the discussion general for now, IPv6 uses these ideas the same way as IPv4:

Key Topic

- To be able to build and send IPv6 packets out an interface, end-user devices need an IPv6 address on that interface.
- End-user hosts need to know the IPv6 address of a default router, to which the host sends IPv6 packets if the host is in a different subnet.
- IPv6 routers de-encapsulate and re-encapsulate each IPv6 packet when routing the packet.
- IPv6 routers make routing decisions by comparing the IPv6 packet's destination address to the router's IPv6 routing table; the matched route lists directions of where to send the IPv6 packet next.

NOTE You could take the preceding list and replace every instance of IPv6 with IPv4, and all the statements would be true of IPv4 as well.

While the list shows some concepts that should be familiar from IPv4, the next few figures show the concepts with an example. First, Figure 22-4 shows a few settings on a host.

The host (PC1) has an address of 2345::1. PC1 also knows its default gateway of 2345::2. (Both values are valid abbreviations for real IPv6 addresses.) To send an IPv6 packet to host PC2, on another IPv6 subnet, PC1 creates an IPv6 packet and sends it to R1, PC1's default gateway.

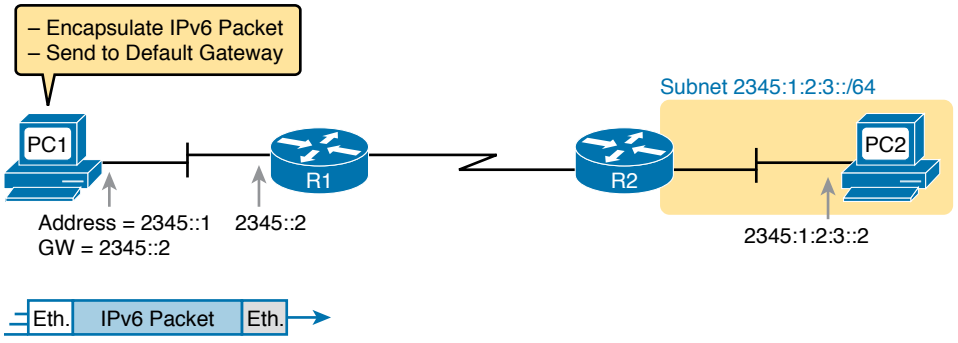


Figure 22-4 *IPv6 Host Building and Sending an IPv6 Packet*

The router (R1) has many small tasks to do when forwarding this IPv6 packet, but for now, focus on the work R1 does related to encapsulation. As seen in Step 1 of Figure 22-5, R1 receives the incoming data-link frame and extracts (de-encapsulates) the IPv6 packet from inside the frame, discarding the original data-link header and trailer. At Step 2, once R1 knows to forward the IPv6 packet to R2, R1 adds a correct outgoing data-link header and trailer to the IPv6 packet, encapsulating the IPv6 packet.

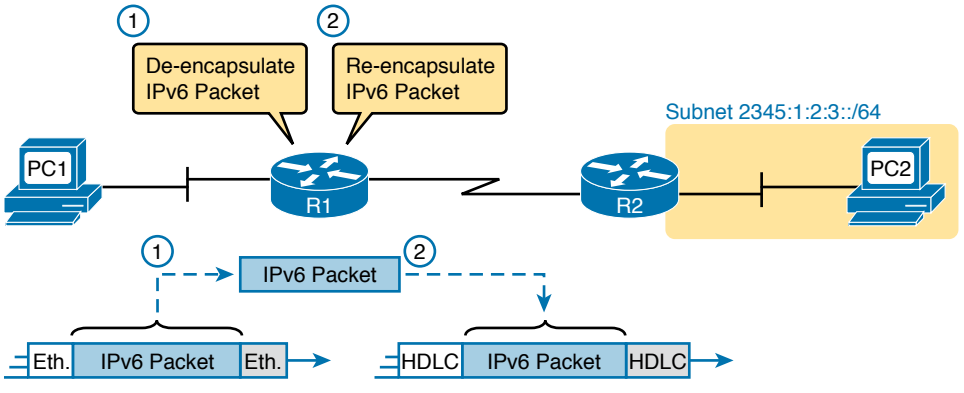


Figure 22-5 *IPv6 Router Performing Routine Encapsulation Tasks When Routing IPv6*

When a router like R1 de-encapsulates the packet from the data-link frame, it must also decide what type of packet sits inside the frame. To do so, the router must look at a protocol type field in the data-link header, which identifies the type of packet inside the data-link frame. Today, most data-link frames carry either an IPv4 packet or an IPv6 packet.

To route an IPv6 packet, a router must use its IPv6 routing table instead of the IPv4 routing table. The router must look at the packet's destination IPv6 address and compare that address to the router's current IPv6 routing table. The router uses the forwarding instructions in the matched IPv6 route to forward the IPv6 packet. Figure 22-6 shows the overall process.

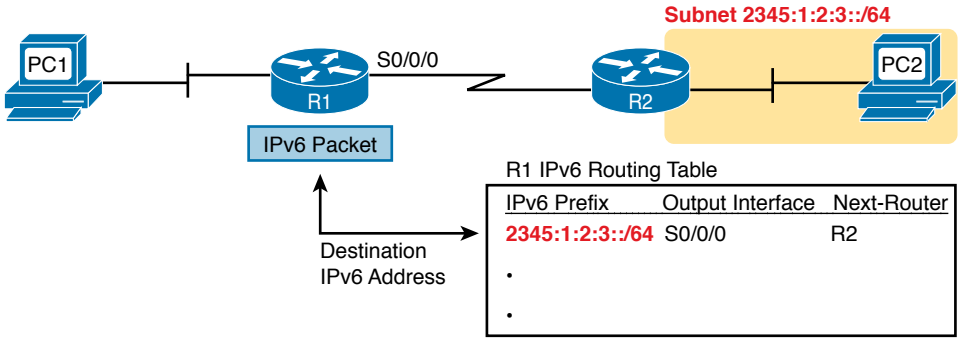


Figure 22-6 *Comparing an IPv6 Packet to R1's IPv6 Routing Table*

Note that again, the process works like IPv4, except that the IPv6 packet lists IPv6 addresses, and the IPv6 routing table lists routing information for IPv6 subnets (called prefixes).

Finally, in most enterprise networks, the routers will route both IPv4 and IPv6 packets at the same time. That is, your company will not decide to adopt IPv6, and then late one weekend night turn off all IPv4 and enable IPv6 on every device. Instead, IPv6 allows for a slow migration, during which some or all routers forward both IPv4 and IPv6 packets. (The migration strategy of running both IPv4 and IPv6 is called *dual stack*.) All you have to do is configure the router to route IPv6 packets, in addition to the existing configuration for routing IPv4 packets.

IPv6 Routing Protocols

IPv6 routers need to learn routes for all the possible IPv6 prefixes (subnets). Just like with IPv4, IPv6 routers use routing protocols, with familiar names, and generally speaking, with familiar functions.

None of the IPv4 routing protocols could be used to advertise IPv6 routes originally. They all required some kind of update to add messages, protocols, and rules to support IPv6. Over time, Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), and Border Gateway Protocol (BGP) were all updated to support IPv6. Table 22-2 lists the names of these routing protocols, with a few comments.

Table 22-2 IPv6 Routing Protocols

Routing Protocol	Defined By	Notes
RIPng (RIP next generation)	RFC	The "next generation" is a reference to a TV series, *Star Trek: the Next Generation*.
OSPFv3 (OSPF version 3)	RFC	The OSPF you have worked with for IPv4 is actually OSPF version 2, so the new version for IPv6 is OSPFv3.
EIGRPv6 (EIGRP for IPv6)	Cisco	Cisco owns the rights to the EIGRP protocol, but Cisco also now publishes EIGRP as an informational RFC.
MP BGP-4 (Multiprotocol BGP version 4)	RFC	BGP version 4 was created to be highly extendable; IPv6 support was added to BGP version 4 through one such enhancement, MP BGP-4.

In addition, these routing protocols also follow the same interior gateway protocol (IGP) and exterior gateway protocol (EGP) conventions as their IPv4 cousins. RIPng, EIGRPv6, and OSPFv3 act as interior gateway protocols, advertising IPv6 routes inside an enterprise.

As you can see from this introduction, IPv6 uses many of the same big ideas as IPv4. Both define headers with a source and destination address. Both define the routing of packets, with the routing process discarding old data-link headers and trailers when forwarding the packets. And routers use the same general process to make a routing decision, comparing the packet's destination IP address to the routing table.

The big differences between IPv4 and IPv6 revolve around the bigger IPv6 addresses. The next topic begins looking at the specifics of these IPv6 addresses.

IPv6 Addressing Formats and Conventions

The CCNA exam requires some fundamental skills in working with IPv4 addresses. For example, you need to be able to interpret IPv4 addresses, like 172.21.73.14. You need to be able to work with prefix-style masks, like /25, and interpret what that means when used with a particular IPv4 address. And you need to be able to take an address and mask, like 172.21.73.14/25, and find the subnet ID.

This second major section of this chapter discusses these same ideas for IPv6 addresses. In particular, this section looks at

- How to write and interpret unabbreviated 32-digit IPv6 addresses
- How to abbreviate IPv6 addresses and how to interpret abbreviated addresses
- How to interpret the IPv6 prefix length mask
- How to find the IPv6 prefix (subnet ID), based on an address and prefix length mask

The biggest challenge with these tasks lies in the sheer size of the numbers. Thankfully, the math to find the subnet ID—often a challenge for IPv4—is easier for IPv6, at least to the depth discussed in this book.

Representing Full (Unabbreviated) IPv6 Addresses

IPv6 uses a convenient hexadecimal (hex) format for addresses. To make it more readable, IPv6 uses a format with eight sets of four hex digits, with each set of four digits separated by a colon. For example:

```
2340:1111:AAAA:0001:1234:5678:9ABC:1234
```

NOTE For convenience, the author uses the term *quartet* for one set of four hex digits, with eight quartets in each IPv6 address. Note that the IPv6 RFCs do not use the term *quartet*.

IPv6 addresses also have a binary format as well, but thankfully, most of the time you do not need to look at the binary version of the addresses. However, in those cases, converting from hex to binary is relatively easy. Just change each hex digit to the equivalent 4-bit value listed in Table 22-3.

Table 22-3 Hexadecimal/Binary Conversion Chart

Hex	Binary	Hex	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Abbreviating and Expanding IPv6 Addresses

IPv6 also defines ways to abbreviate or shorten how you write or type an IPv6 address. Why? Although using a 32-digit hex number works much better than working with a 128-bit binary number, 32 hex digits are still a lot of digits to remember, recognize in command output, and type on a command line. The IPv6 address abbreviation rules let you shorten these numbers.

Computers and routers typically use the shortest abbreviation, even if you type all 32 hex digits of the address. So even if you would prefer to use the longer unabbreviated version of the IPv6 address, you need to be ready to interpret the meaning of an abbreviated IPv6 address as listed by a router or host. This section first looks at abbreviating addresses and then at expanding addresses.

Abbreviating IPv6 Addresses

Two basic rules let you, or any computer, shorten or abbreviate an IPv6 address:

Key Topic

1. Inside each quartet of four hex digits, remove the leading 0s (0s on the left side of the quartet) in the three positions on the left. (Note: at this step, a quartet of 0000 will leave a single 0.)
2. Find any string of two or more consecutive quartets of all hex 0s, and replace that set of quartets with a double colon (::). The :: means "two or more quartets of all 0s." However, you can use :: only once in a single address because otherwise the exact IPv6 might not be clear.

For example, consider the following IPv6 address. The bold digits represent digits in which the address could be abbreviated.

FE00:**0000:0000:000**1:**0000:0000:0000:00**56

Applying the first rule, you would look at all eight quartets independently. In each, remove all the leading 0s. Note that five of the quartets have four 0s, so for these, remove only three 0s, leaving the following value:

FE00:0:0:1:0:0:0:56

While this abbreviation is valid, the address can be abbreviated more, using the second rule. In this case, two instances exist where more than one quartet in a row has only a 0. Pick the longest such sequence, and replace it with ::, giving you the shortest legal abbreviation:

```
FE00:0:0:1::56
```

While FE00:0:0:1::56 is indeed the shortest abbreviation, this example happens to make it easier to see the two most common mistakes when abbreviating IPv6 addresses. First, never remove trailing 0s in a quartet (0s on the right side of the quartet). In this case, the first quartet of FE00 cannot be shortened at all because the two 0s trail. So, the following address, which begins now with only FE in the first quartet, is not a correct abbreviation of the original IPv6 address:

```
FE:0:0:1::56
```

The second common mistake is to replace all series of all 0 quartets with a double colon. For example, the following abbreviation would be incorrect for the original IPv6 address listed in this topic:

```
FE00::1::56
```

The reason this abbreviation is incorrect is that now you do not know how many quartets of all 0s to substitute into each :: to find the original unabbreviated address.

Expanding Abbreviated IPv6 Addresses

To expand an IPv6 address back into its full unabbreviated 32-digit number, use two similar rules. The rules basically reverse the logic of the previous two rules:

1. In each quartet, add leading 0s as needed until the quartet has four hex digits.
2. If a double colon (::) exists, count the quartets currently shown; the total should be less than 8. Replace the :: with multiple quartets of 0000 so that eight total quartets exist.

The best way to get comfortable with these addresses and abbreviations is to do some yourself. Table 22-4 lists some practice problems, with the full 32-digit IPv6 address on the left and the best abbreviation on the right. The table gives you either the expanded or abbreviated address, and you need to supply the opposite value. The answers sit at the end of the chapter, in the section "Answers to Earlier Practice Problems."

Table 22-4 IPv6 Address Abbreviation and Expansion Practice

Full	Abbreviation
2340:0000:0010:0100:1000:ABCD:0101:1010	
	30A0:ABCD:EF12:3456:ABC:B0B0:9999:9009
2222:3333:4444:5555:0000:0000:6060:0707	
	3210::
210F:0000:0000:0000:CCCC:0000:0000:000D	
	34BA:B:B::20
FE80:0000:0000:0000:DEAD:BEFF:FEEF:CAFE	
	FE80::FACE:BAFF:FEBE:CAFE

Representing the Prefix Length of an Address

IPv6 uses a mask concept, called the *prefix length*, similar to IPv4 subnet masks. Similar to the IPv4 prefix-style mask, the IPv6 prefix length is written as a /, followed by a decimal number. The prefix length defines how many bits of the IPv6 address define the IPv6 prefix, which is basically the same concept as the IPv4 subnet ID.

When writing an IPv6 address and prefix length in documentation, you can choose to leave a space before the /, or not, as shown in the next two examples.

```
2222:1111:0:1:A:B:C:D/64
2222:1111:0:1:A:B:C:D /64
```

Finally, note that the prefix length is a number of bits, so with IPv6, the legal value range is from 0 through 128, inclusive.

Calculating the IPv6 Prefix (Subnet ID)

With IPv4, you can take an IP address and the associated subnet mask, and calculate the subnet ID. With IPv6 subnetting, you can take an IPv6 address and the associated prefix length, and calculate the IPv6 equivalent of the subnet ID: an *IPv6 prefix*.

Like with different IPv4 subnet masks, some IPv6 prefix lengths make for an easy math problem to find the IPv6 prefix, while some prefix lengths make the math more difficult. This section looks at the easier cases, mainly because the size of the IPv6 address space lets us all choose to use IPv6 prefix lengths that make the math much easier.

Finding the IPv6 Prefix

In IPv6, a prefix represents a group of IPv6 addresses. For now, this section focuses on the math, and only the math, for finding the number that represents that prefix. Chapter 23, "IPv6 Addressing and Subnetting," then starts putting more meaning behind the actual numbers.

Each IPv6 prefix, or subnet if you prefer, has a number that represents the group. Per the IPv6 RFCs, the number itself is also called the *prefix*, but many people just call it a subnet number or subnet ID, using the same terms as IPv4.

As with IPv4, you can start with an IPv6 address and prefix length, and find the prefix, with the same general rules that you use in IPv4. If the prefix length is /P, use these rules:

Key Topic

1. Copy the first P bits.
2. Change the rest of the bits to 0.

When using a prefix length that happens to be a multiple of 4, you do not have to think in terms of bits, but in terms of hex digits. A prefix length that is a multiple of 4 means that each hex digit is either copied or changed to hex 0. Just for completeness, if the prefix length is indeed a multiple of 4, the process becomes

1. Identify the number of hex digits in the prefix by dividing the prefix length (which is in bits) by 4.
2. Copy the hex digits determined to be in the prefix per the first step.
3. Change the rest of the hex digits to 0.

Figure 22-7 shows an example, with a prefix length of 64. In this case, Step 1 looks at the /64 prefix length and calculates that the prefix has 16 hex digits. Step 2 copies the first 16 digits of the IPv6 address, while Step 3 records hex 0s for the rest of the digits.

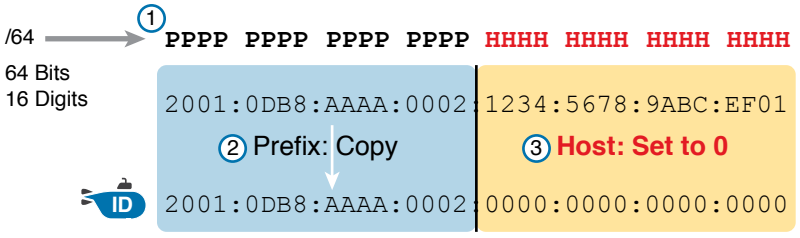


Figure 22-7 *Creating the IPv6 Prefix from an Address/Length*

After you find the IPv6 prefix, you should also be ready to abbreviate the IPv6 prefix using the same rules you use to abbreviate IPv6 addresses. However, you should pay extra attention to the end of the prefix because it often has several octets of all 0 values. As a result, the abbreviation typically ends with two colons (::).

For example, consider the following IPv6 address that is assigned to a host on a LAN:

```
2000:1234:5678:9ABC:1234:5678:9ABC:1111/64
```

This example shows an IPv6 address that itself cannot be abbreviated. After you calculate the prefix for the subnet in which the address resides, by zeroing out the last 64 bits (16 digits) of the address, you find the following prefix value:

```
2000:1234:5678:9ABC:0000:0000:0000:0000/64
```

This value can be abbreviated, with four quartets of all 0s at the end, as follows:

```
2000:1234:5678:9ABC::/64
```

To get better at the math, take some time to work through finding the prefix for several practice problems, as listed in Table 22-5. The answers sit at the end of the chapter, in the section "Answers to Earlier Practice Problems."

Table 22-5 Finding the IPv6 Prefix from an Address/Length Value

Address/Length	Prefix
2340:0:10:100:1000:ABCD:101:1010/64	
30A0:ABCD:EF12:3456:ABC:B0B0:9999:9009/64	
2222:3333:4444:5555::6060:707/64	
3210::ABCD:101:1010/64	

Address/Length	Prefix
210F::CCCC:B0B0:9999:9009/64	
34BA:B:B:0:5555:0:6060:707/64	
3124::DEAD:CAFE:FF:FE00:1/64	
2BCD::FACE:BEFF:FEBE:CAFE/64	

Working with More-Difficult IPv6 Prefix Lengths

Some prefix lengths make the math to find the prefix very easy, some mostly easy, and some require you to work in binary. If the prefix length is a multiple of 16, the process of copying part of the address copies entire quartets. If the prefix length is not a multiple of 16 but is a multiple of 4, at least the boundary sits at the edge of a hex digit, so you can avoid working in binary.

Although the /64 prefix length is by far the most common prefix length, you should be ready to find the prefix when using a prefix length that is any multiple of 4. For example, consider the following IPv6 address and prefix length:

2000:1234:5678:9ABC:1234:5678:9ABC:1111/56

Because this example uses a /56 prefix length, the prefix includes the first 56 bits, or first 14 complete hex digits, of the address. The rest of the hex digits will be 0, resulting in the following prefix:

2000:1234:5678:9A**00:0000:0000:0000:0000**/56

This value can be abbreviated, with four quartets of all 0s at the end, as follows:

2000:1234:5678:9A**00**::/56

This example shows an easy place to make a mistake. Sometimes, people look at the /56 and think of that as the first 14 hex digits, which is correct. However, they then copy the first 14 hex digits and add a double colon, showing the following:

2000:1234:5678:**9A**::/56

This abbreviation is not correct because it removed the trailing "00" at the end of the fourth quartet. If you expanded the abbreviated value, it would begin with 2000:1234:5678:009A, not 2000:1234:5678:9A00. So, be careful when abbreviating when the boundary is not at the edge of a quartet.

Once again, some extra practice can help. Table 22-6 uses examples that have a prefix length that is a multiple of 4, but is not on a quartet boundary, just to get some extra practice. The answers sit at the end of the chapter, in the section "Answers to Earlier Practice Problems."

Table 22-6 Finding the IPv6 Prefix from an Address/Length Value

Address/Length	Prefix
34BA:B:B:0:5555:0:6060:707/80	
3124::DEAD:CAFE:FF:FE00:1/80	
2BCD::FACE:BEFF:FEBE:CAFE/48	
3FED:F:E0:D00:FACE:BAFF:FE00:0/48	
210F:A:B:C:CCCC:B0B0:9999:9009/40	
34BA:B:B:0:5555:0:6060:707/36	
3124::DEAD:CAFE:FF:FE00:1/60	
2BCD::FACE:1:BEFF:FEBE:CAFE/56	

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 22-7 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 22-7 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review command tables		Book
Review memory table		Book, website

Review All the Key Topics

Key Topic

Table 22-8 Key Topics for Chapter 22

Key Topic Element	Description	Page Number
List	Similarities between IPv4 and IPv6	527
List	Rules for abbreviating IPv6 addresses	531
List	Rules for expanding an abbreviated IPv6 address	532
List	Process steps to find an IPv6 prefix, based on the IPv6 address and prefix length	533

Key Terms You Should Know

IPv4 address exhaustion, IP version 6 (IPv6), OSPF version 3 (OSPFv3), EIGRP version 6 (EIGRPv6), prefix, prefix length, quartet

Additional Practice for This Chapter's Processes

For additional practice with IPv6 abbreviations, you may do the same set of practice problems based on Appendix G, "Practice for Chapter 22: Fundamentals of IP Version 6." You have two options to use:

PDF: Navigate to the companion website and open the PDF for Appendix G.

Application: Navigate to the companion website and use these applications:

"Practice Exercise: Abbreviating and Expanding Addresses"

"Practice Exercise: Calculating the IPv6 Prefix"

"Practice Exercise: Calculating the IPv6 Prefix Round 2"

Answers to Earlier Practice Problems

This chapter includes practice problems spread around different locations in the chapter. The answers are located in Tables 22-9, 22-10, and 22-11.

Table 22-9 Answers to Questions in the Earlier Table 22-4

Full	Abbreviation
2340:0000:0010:0100:1000:ABCD:0101:1010	2340:0:10:100:1000:ABCD:101:1010
30A0:ABCD:EF12:3456:0ABC:B0B0:9999:9009	30A0:ABCD:EF12:3456:ABC:B0B0:9999:9009
2222:3333:4444:5555:0000:0000:6060:0707	2222:3333:4444:5555::6060:707
3210:0000:0000:0000:0000:0000:0000:0000	3210::
210F:0000:0000:0000:CCCC:0000:0000:000D	210F::CCCC:0:0:D
34BA:000B:000B:0000:0000:0000:0000:0020	34BA:B:B::20
FE80:0000:0000:0000:DEAD:BEFF:FEEF:CAFE	FE80::DEAD:BEFF:FEEF:CAFE
FE80:0000:0000:0000:FACE:BAFF:FEBE:CAFE	FE80::FACE:BAFF:FEBE:CAFE

Table 22-10 Answers to Questions in the Earlier Table 22-5

Address/Length	Prefix
2340:0:10:100:1000:ABCD:101:1010/64	2340:0:10:100::/64
30A0:ABCD:EF12:3456:ABC:B0B0:9999:9009/64	30A0:ABCD:EF12:3456::/64
2222:3333:4444:5555::6060:707/64	2222:3333:4444:5555::/64
3210::ABCD:101:1010/64	3210::/64
210F::CCCC:B0B0:9999:9009/64	210F::/64
34BA:B:B:0:5555:0:6060:707/64	34BA:B:B::/64
3124::DEAD:CAFE:FF:FE00:1/64	3124:0:0:DEAD::/64
2BCD::FACE:BEFF:FEBE:CAFE/64	2BCD::/64

Table 22-11 Answers to Questions in the Earlier Table 22-6

Address/Length	Prefix
34BA:B:B:0:5555:0:6060:707/80	34BA:B:B:0:5555::/80
3124::DEAD:CAFE:FF:FE00:1/80	3124:0:0:DEAD:CAFE::/80
2BCD::FACE:BEFF:FEBE:CAFE/48	2BCD::/48
3FED:F:E0:D00:FACE:BAFF:FE00:0/48	3FED:F:E0::/48
210F:A:B:C:CCCC:B0B0:9999:9009/40	210F:A::/40
34BA:B:B:0:5555:0:6060:707/36	34BA:B::/36
3124::DEAD:CAFE:FF:FE00:1/60	3124:0:0:DEA0::/60
2BCD::FACE:1:BEFF:FEBE:CAFE/56	2BCD:0:0:FA00::/56

This page intentionally left blank

CHAPTER 23

IPv6 Addressing and Subnetting

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.8 Configure and verify IPv6 addressing and prefix

1.9 Compare and contrast IPv6 address types

1.9.a Global unicast

1.9.b Unique local

IPv4 organizes the address space in a couple of ways. First, IPv4 splits addresses by class, with Classes A, B, and C defining unicast IPv4 addresses. (The term *unicast* refers to the fact that each address is used by only one interface.) Then, within the Class A, B, and C address range, the Internet Assigned Numbers Authority (IANA) and the Internet Corporation for Assigned Names and Numbers (ICANN) reserve most of the addresses as public IPv4 addresses, with a few reserved as private IPv4 addresses.

IPv6 does not use any concept like the classful network concept used by IPv4. However, IANA does still reserve some IPv6 address ranges for specific purposes, even with some address ranges that serve as both public IPv6 addresses and private IPv6 addresses. IANA also attempts to take a practical approach to reserving ranges of the entire IPv6 address space for different purposes, using the wisdom gained from several decades of fast growth in the IPv4 Internet.

This chapter has two major sections. The first examines *global unicast addresses*, which serve as public IPv6 addresses. The second major section looks at *unique local addresses*, which serve as private IPv6 addresses.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 23-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Global Unicast Addressing Concepts	1–4
Unique Local Unicast Addresses	5

1. Which of the following IPv6 addresses appears to be a unique local unicast address, based on its first few hex digits?
   a. 3123:1:3:5::1
   b. FE80::1234:56FF:FE78:9ABC
   c. FDAD::1
   d. FF00::5
2. Which of the following IPv6 addresses appears to be a global unicast address, based on its first few hex digits?
   a. 3123:1:3:5::1
   b. FE80::1234:56FF:FE78:9ABC
   c. FDAD::1
   d. FF00::5
3. When subnetting an IPv6 address block, an engineer shows a drawing that breaks the address structure into three pieces. Comparing this concept to a three-part IPv4 address structure, which part of the IPv6 address structure is most like the IPv4 network part of the address?
   a. Subnet
   b. Interface ID
   c. Network
   d. Global routing prefix
   e. Subnet router anycast
4. When subnetting an IPv6 address block, an engineer shows a drawing that breaks the address structure into three pieces. Assuming that all subnets use the same prefix length, which of the following answers lists the name of the field on the far right side of the address?
   a. Subnet
   b. Interface ID
   c. Network
   d. Global routing prefix
   e. Subnet router anycast
5. For the IPv6 address FD00:1234:5678:9ABC:DEF1:2345:6789:ABCD, which part of the address is considered the global ID of the unique local address?
   a. None; this address has no global ID.
   b. 00:1234:5678:9ABC
   c. DEF1:2345:6789:ABCD
   d. 00:1234:5678
   e. FD00

Foundation Topics

Global Unicast Addressing Concepts

This first major section of the chapter focuses on one type of unicast IPv6 addresses: global unicast addresses. As it turns out, many of the general concepts and processes behind these global unicast IPv6 addresses follow the original intent for public IPv4 addresses. So, this section begins with a review of some IPv4 concepts, followed by the details of how a company can use global unicast addresses.

This first section also discusses IPv6 subnetting and the entire process of taking a block of global unicast addresses and creating subnets for one company. This process takes a globally unique global routing prefix, creates IPv6 subnets, and assigns IPv6 addresses from within each subnet, much like with IPv4.

Public and Private IPv6 Addresses

In the history of IPv4 addressing, the world started out with a plan that gave every single host a globally unique public IPv4 address. However, as discussed in several places already, the IPv4 address space had too few addresses. So, in the 1990s, companies started using addresses from the private IPv4 address range, as defined in RFC 1918. These companies either simply did not connect to the Internet, or to connect to the Internet, they used Network Address Translation (NAT), sharing a few public globally unique IPv4 addresses for all host connections into the Internet.

IPv6 allows two similar options of public and private unicast addressing, beginning with *global unicast* addresses as the public IPv6 address space. Similar to public IPv4 addresses, IPv6 global unicast addresses rely on an administrative process that assigns each company a unique IPv6 address block. Each company then subnets this IPv6 address block and only uses addresses from within that block. The result: that company uses addresses that are unique across the globe as well.

The second IPv6 option uses *unique local* IPv6 addresses, which work more like the IPv4 private addresses. Companies that do not plan to connect to the Internet and companies that plan to use IPv6 NAT can use these private unique local addresses. The process also works similarly to IPv4: The engineer can read the details in an RFC, pick some numbers, and start assigning IPv6 addresses without having to register with IANA or any other authority.

The following lists summarizes the comparisons between global unicast addresses and unique local addresses:

Global unicast: Addresses that work like public IPv4 addresses. The organization that needs IPv6 addresses asks for a registered IPv6 address block, which is assigned as a global routing prefix. After that, only that organization uses the addresses inside that block of addresses—that is, the addresses that begin with the assigned prefix.

Unique local: Works somewhat like private IPv4 addresses, with the possibility that multiple organizations use the exact same addresses, and with no requirement for registering with any numbering authority.

Answers to the "Do I Know This Already?" quiz:

1 C **2** A **3** D **4** B **5** D

The rest of this first major section of the chapter examines global unicast addresses in more detail, while the second major section of the chapter examines unique local addresses.

The IPv6 Global Routing Prefix

IPv6 global unicast addresses allow IPv6 to work more like the original design of the IPv4 Internet. Each organization asks for a block of IPv6 addresses, which no one else can use. That organization further subdivides the address block into smaller chunks, called *subnets*. Finally, to choose what IPv6 address to use for any host, the engineer chooses an address from the right subnet.

That reserved block of IPv6 addresses—a set of addresses that only one company can use—is called a *global routing prefix*. Each organization that wants to connect to the Internet and use IPv6 global unicast addresses should ask for and receive a global routing prefix. Very generally, you can think of the global routing prefix like an IPv4 Class A, B, or C network number from the range of public IPv4 addresses.

The term *global routing prefix* might not make you think of a block of IPv6 addresses at first. The term actually refers to the idea that Internet routers can have one route that refers to all the addresses inside the address block, without a need to have routes for smaller parts of that block. For example, Figure 23-1 shows three companies, with three different IPv6 global routing prefixes; the router on the right (R4) has one IPv6 route for each global routing prefix.

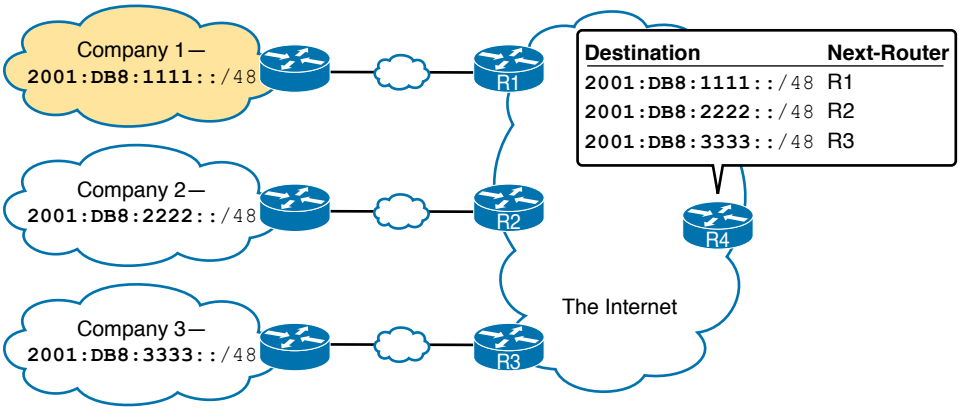


Figure 23-1 *Three Global Routing Prefixes, with One Route per Prefix*

The global routing prefix sets those IPv6 addresses apart for use by that one company, just like a public IPv4 network or CIDR address block does in IPv4. All IPv6 addresses inside that company should begin with that global routing prefix, to avoid using other companies' IPv6 addresses. No other companies should use IPv6 addresses with that same prefix. And thankfully, IPv6 has plenty of space to allow all companies to have a global routing prefix, with plenty of addresses.

Both the IPv6 and IPv4 address assignment processes rely on the same organizations: IANA (along with ICANN), the Regional Internet Registries (RIR), and ISPs. For example,

an imaginary company, Company1, received the assignment of a global routing prefix. The prefix means "All addresses whose first 12 hex digits are 2001:0DB8:1111," as represented by prefix 2001:0DB8:1111::/48. To receive that assignment, the process shown in Figure 23-2 happened.

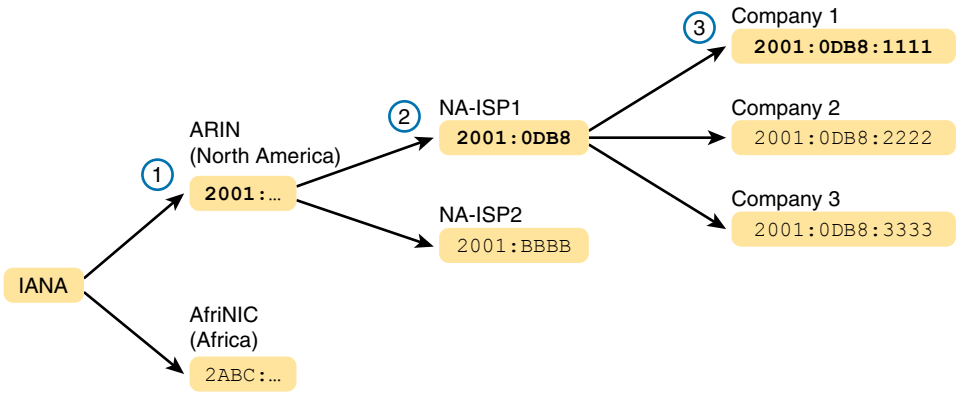


Figure 23-2 *Prefix Assignment with IANA, RIRs, and ISPs*

The event timeline in the figure uses a left-to-right flow; in other words, the event on the far left must happen first. Following the flow from left to right in the figure:

1. **IANA allocates ARIN prefix 2001::/16:** ARIN (the RIR for North America) asks IANA for the allocation of a large block of addresses. In this imaginary example, IANA gives ARIN a prefix of "all addresses that begin 2001," or 2001::/16.
2. **ARIN allocates NA-ISP1 prefix 2001:0DB8::/32:** NA-ISP1, an imaginary ISP based in North America, asks ARIN for a new IPv6 prefix. ARIN takes a subset of its 2001::/16 prefix, specifically all addresses that begin with the 32 bits (8 hex digits) 2001:0DB8, and allocates it to the ISP.
3. **NA-ISP1 assigns Company 1 2001:0DB8:1111::/48:** Company 1 decides to start supporting IPv6, so it goes to its ISP, NA-ISP1, to ask for a block of global unicast addresses. NA-ISP1 assigns Company 1 a "small" piece of NA-ISP1's address block, in this case the addresses that begin with the 48 bits (12 hex digits) of 2001:0DB8:1111 (2001:0DB8:1111::/48).

NOTE If you do not plan to connect to the Internet using IPv6 for a while and just want to experiment, you do not need to ask for an IPv6 global routing prefix to be assigned. Just make up IPv6 addresses and configure your devices, or use unique local addresses as discussed toward the end of this chapter.

Address Ranges for Global Unicast Addresses

Global unicast addresses make up the majority of the IPv6 address space. However, unlike IPv4, the rules for which IPv6 addresses fall into which category are purposefully more flexible than they were with IPv4 and the rules for IPv4 Classes A, B, C, D, and E.

Originally, IANA reserved all IPv6 addresses that begin with hex 2 or 3 as global unicast addresses. (This address range can be written succinctly as prefix 2000::/3.)

Later IANA made the global unicast address range wider, basically to include all IPv6 addresses not otherwise allocated for other purposes. For example, the unique local unicast addresses, discussed later in this chapter, all start with hex FD. So, while global unicast addresses would not include any addresses that begin with FD, any address ranges that are not specifically reserved, for now, are considered to be global unicast addresses.

Finally, just because an amazingly enormous number of addresses sit within the global unicast address range, IANA does not assign prefixes from all over the address range. IPv4 has survived well for more than 30 years with an admittedly too-small address size because IANA has adopted good practices to conserve the IPv4 address space. By making smart and practical choices in assigning IPv6 addresses, the IPv6 address space could last much longer than IPv4.

Table 23-2 lists the address prefixes discussed in this book and their purpose.

Key Topic

Table 23-2 Some Types of IPv6 Addresses and Their First Hex Digit(s)

Address Type	First Hex Digits
Global unicast	2 or 3 (originally); all not otherwise reserved (today)
Unique local	FD
Multicast	FF
Link local	FE80

IPv6 Subnetting Using Global Unicast Addresses

After an enterprise has a block of reserved global unicast addresses—in other words, a global routing prefix—the company needs to subdivide that large address block into subnets.

Subnetting IPv6 addresses works generally like IPv4, but with mostly simpler math (hoorah!). Because of the absolutely large number of addresses available, most everyone uses the easiest possible IPv6 prefix length: /64. Using /64 as the prefix length for all subnets makes the IPv6 subnetting math just as easy as using a /24 mask for all IPv4 subnets. In addition, the dynamic IPv6 address assignment process works better with a /64 prefix length as well; so in practice, and in this book, expect IPv6 designs to use a /64 prefix length.

This section does walk you through the different parts of IPv6 subnetting, while mostly using examples that use a /64 prefix length. The discussion defines the rules about which addresses should be in the same subnet and which addresses need to be in different subnets. Plus this section looks at how to analyze the global routing prefix and associated prefix length to find all the IPv6 prefixes (subnet IDs) and the addresses in each subnet.

NOTE If the IPv4 subnetting concepts are a little vague, you might want to reread Chapter 11, "Perspectives on IPv4 Subnetting," which discusses the subnetting concepts for IPv4.

Deciding Where IPv6 Subnets Are Needed

First, IPv6 and IPv4 both use the same concepts about where a subnet is needed: one for each VLAN and one for each point-to-point WAN connection (serial and Ethernet). Figure 23-3 shows an example of the idea, using the small enterprise internetwork of Company 1. Company 1 has two LANs, with a point-to-point serial link connecting the sites. It also has an Ethernet WAN link connected to an ISP. Using the same logic you would use for IPv4, Company 1 needs four IPv6 subnets.

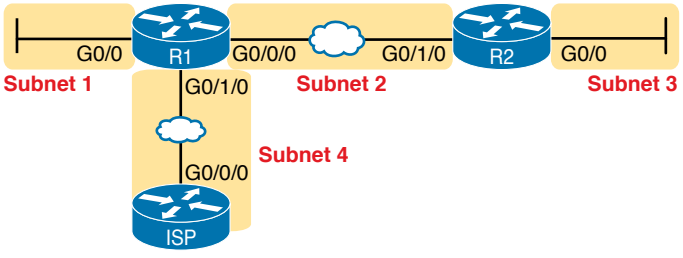


Figure 23-3 *Locations for IPv6 Subnets*

The Mechanics of Subnetting IPv6 Global Unicast Addresses

To understand how to subnet your one large block of IPv6 addresses, you need to understand some of the theory and mechanisms IPv6 uses. To learn those details, it can help to compare IPv6 with some similar concepts from IPv4.

With IPv4, without subnetting, an address has two parts: a network part and a host part. Class A, B, and C rules define the length of the network part, with the host part making up the rest of the 32-bit IPv4 address, as shown in Figure 23-4.

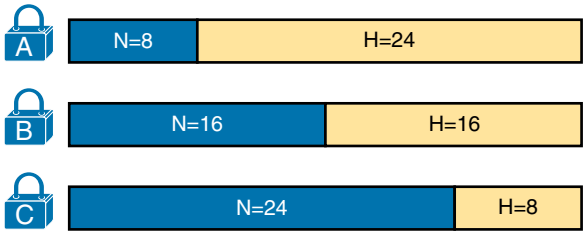


Figure 23-4 *Classful View of Unsubnetted IPv4 Networks*

To subnet an IPv4 Class A, B, or C network, the network engineer for the enterprise makes some choices. Conceptually, the engineer creates a three-part view of the addresses, adding a subnet field in the center while shortening the host field. (Many people call this "borrowing host bits.") The size of the network part stays locked per the Class A, B, and C rules, with the line between the subnet and host part being flexible, based on the choice of subnet mask. Figure 23-5 shows the idea for a subnetted Class B network.

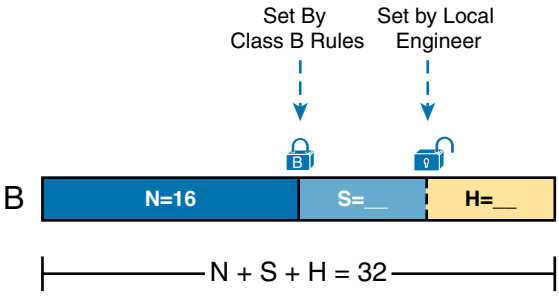


Figure 23-5 *Classful View of Subnetted IPv4 Networks*

IPv6 uses a similar concept, with the details in Figure 23-6. The structure shows three major parts, beginning with the global routing prefix, which is the initial value that must be the same in all IPv6 addresses inside the enterprise. The address ends with the interface ID, which acts like the IPv4 host field. The subnet field sits between the two other fields, used as a way to number and identify subnets, much like the subnet field in IPv4 addresses.

Key Topic

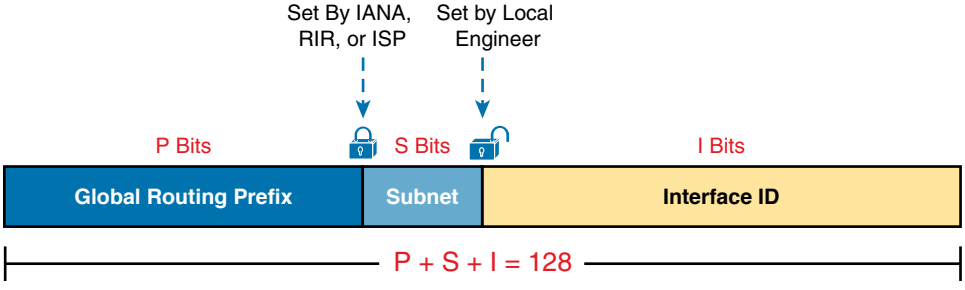


Figure 23-6 *Structure of Subnetted IPv6 Global Unicast Addresses*

First, just think about the general idea with IPv6, comparing Figure 23-6 to Figure 23-5. The IPv6 global routing prefix (the prefix/length assigned by the RIR or ISP) acts like the IPv4 network part of the address structure. The IPv6 subnet part acts like the IPv4 subnet part. And the right side of the IPv6, formally called the *interface ID* (short for interface identifier), acts like the IPv4 host field.

Now focus on the IPv6 global routing prefix and its prefix length. Unlike IPv4, IPv6 has no concept of address classes, so no preset rules determine the prefix length of the global routing prefix. However, when a company applies to an ISP, RIR, or any other organization that can assign a global routing prefix, that assignment includes both the prefix and the prefix length. After a company receives a global routing prefix and that prefix length, the length of the prefix typically does not change over time and is basically locked. (Note that the prefix length of the global routing prefix is often between /32 and /48, or possibly as long as /56.)

Next, look to the right side of Figure 23-6 to the interface ID field. For several reasons that become more obvious the more you learn about IPv6, this field is often 64 bits long. Does it have to be 64 bits long? No. However, using a 64-bit interface ID field works well in real networks, and there are no reasons to avoid using a 64-bit interface ID field.

Finally, look to the subnet field in the center of Figure 23-6. Similar to IPv4, this field creates a place with which to number IPv6 subnets. The length of the subnet field is based on the other two facts: the length of the global routing prefix and the length of the interface ID. And with the commonly used 64-bit interface ID field, the subnet field is typically 64–P bits, with P being the length of the global routing prefix.

Next, consider the structure of a specific global unicast IPv6 address, 2001:0DB8:1111:0001:0000:0000:0000:0001, as seen in Figure 23-7. In this case:

- The company was assigned prefix 2001:0DB8:1111, with prefix length /48.
- The company uses the usual 64-bit interface ID.
- The company has a subnet field of 16 bits, allowing for 2^{16} IPv6 subnets.

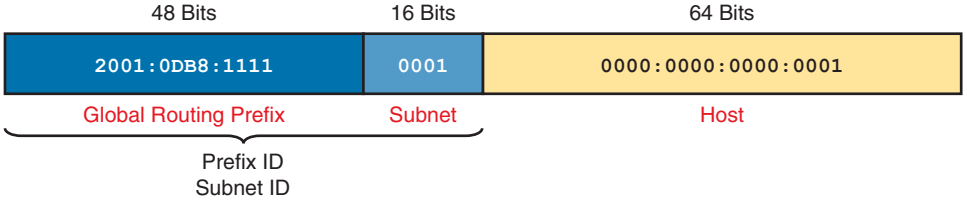


Figure 23-7 *Address Structure for Company 1 Example*

The example in Figure 23-7, along with a little math, shows one reason why so many companies use a /64 prefix length for all subnets. With this structure, Company 1 can support 2^{16} possible subnets (65,536). Few companies need that many subnets. Then, each subnet supports over 10^{18} addresses per subnet (2^{64}, minus some reserved values). So, for both subnets and hosts, the address structure supports far more than are needed. Plus, the /64 prefix length for all subnets makes the math simple because it cuts the 128-bit IPv6 address in half.

Listing the IPv6 Subnet Identifier

Like with IPv4, IPv6 needs to identify each IPv6 subnet with some kind of a subnet identifier, or subnet ID. Figure 23-7 lists the informal names for this number (subnet ID) and the more formal name (prefix ID). Routers then list the IPv6 subnet ID in routing tables, along with the prefix length.

Chapter 22, "Fundamentals of IP Version 6," already discussed how to find the subnet ID, given an IPv6 address and prefix length. The math works the same way when working with global unicast addresses, as well as the unique local addresses discussed later in the chapter. Chapter 22 has already discussed the math, but for completeness, note that the subnet ID shown in Figure 23-7 would be

```
2001:DB8:1111:1::/64
```

List All IPv6 Subnets

With IPv4, if you choose to use a single subnet mask for all subnets, you can sit and write down all the subnets of a Class A, B, or C network using that one subnet mask. With IPv6,

the same ideas apply. If you plan to use a single prefix length for all subnets, you can start with the global routing prefix and write down all the IPv6 subnet IDs as well.

To find all the subnet IDs, you simply need to find all the unique values that will fit inside the subnet part of the IPv6 address, basically following these rules:

- All subnet IDs begin with the global routing prefix.
- Use a different value in the subnet field to identify each different subnet.
- All subnet IDs have all 0s in the interface ID.

As an example, take the IPv6 design shown in Figure 23-7, and think about all the subnet IDs. First, all subnets will use the commonly used /64 prefix length. This company uses a global routing prefix of 2001:0DB8:1111::/48, which defines the first 12 hex digits of all the subnet IDs. To find all the possible IPv6 subnet IDs, think of all the combinations of unique values in the fourth quartet and then represent the last four quartets of all 0s with a :: symbol. Figure 23-8 shows the beginning of just such a list.

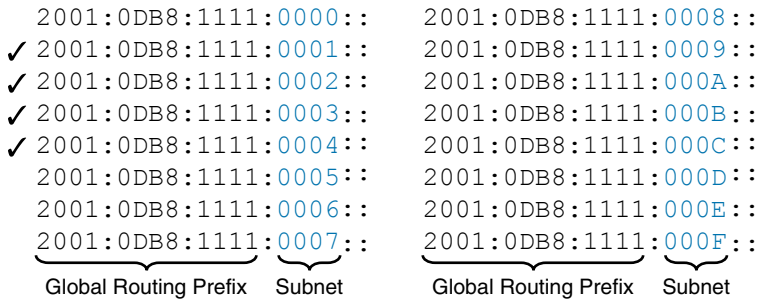


Figure 23-8 *First 16 Possible Subnets with a 16-bit Subnet Field in This Example*

The example allows for 65,536 subnets, so clearly the example will not list all the possible subnets. However, in that fourth quartet, all combinations of hex values would be allowed.

NOTE The IPv6 subnet ID, more formally called the *subnet router anycast address*, is reserved and should not be used as an IPv6 address for any host.

Assign Subnets to the Internetwork Topology

After an engineer lists all the possible subnet IDs (based on the subnet design), the next step is to choose which subnet ID to use for each link that needs an IPv6 subnet. Just like with IPv4, each VLAN, each serial link, each Ethernet WAN link, and many other data-link instances need an IPv6 subnet.

Figure 23-9 shows an example using Company 1 again. The figure uses the four subnets from Figure 23-8 that have check marks beside them. The check marks are just a reminder to not use those four subnets in other locations.

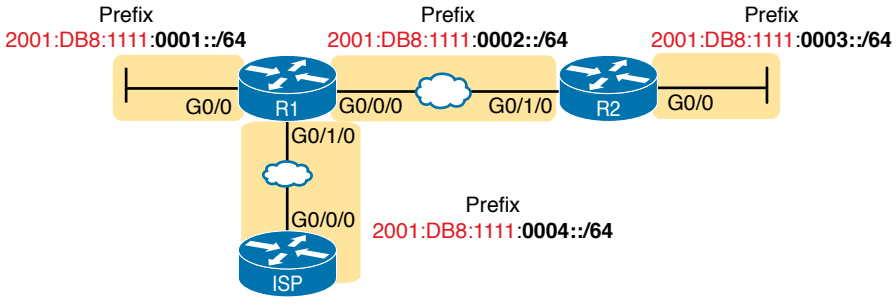


Figure 23-9 *Subnets in Company 1, with Global Routing Prefix of 2001:0DB8:1111::/48*

Assigning Addresses to Hosts in a Subnet

Now that the engineer has planned which IPv6 subnet will be used in each location, the individual IPv6 addressing can be planned and implemented. Each address must be unique, in that no other host interface uses the same IPv6 address. Also, the hosts cannot use the subnet ID itself.

The process of assigning IPv6 addresses to interfaces works similarly to IPv4. Addresses can be configured statically, along with the prefix length, default router, and Domain Name System (DNS) IPv6 addresses. Alternatively, hosts can learn these same settings dynamically, using either Dynamic Host Configuration Protocol (DHCP) or a built-in IPv6 mechanism called Stateless Address Autoconfiguration (SLAAC).

For example, Figure 23-10 shows some static IP addresses that could be chosen for the router interfaces based on the subnet choices shown in Figure 23-9. In each case, the router interfaces use an interface ID that is a relatively low number, easily remembered.

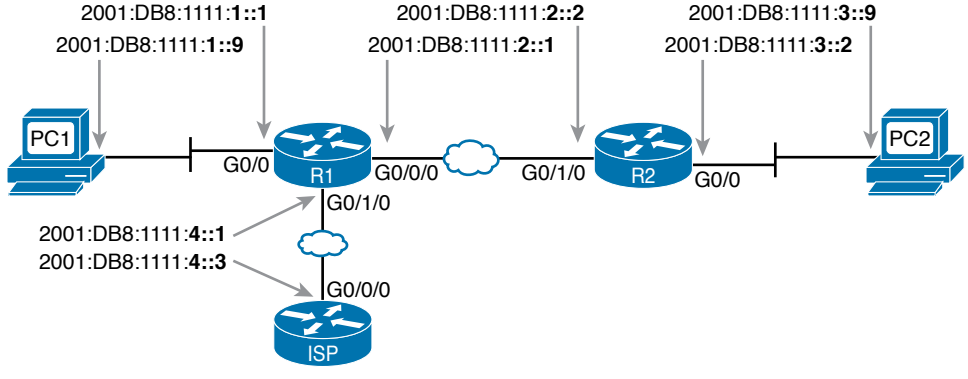


Figure 23-10 *Example Static IPv6 Addresses Based on the Subnet Design of Figure 23-9*

This chapter puts off the details of how to configure the IPv6 addresses until Chapter 24, "Implementing IPv6 Addressing on Routers."

Unique Local Unicast Addresses

Unique local unicast addresses act as private IPv6 addresses. These addresses have many similarities with global unicast addresses, particularly in how to subnet. The biggest difference lies in the literal number (unique local addresses begin with hex FD) and with the administrative process: the unique local prefixes are not registered with any numbering authority and can be used by multiple organizations.

Although the network engineer creates unique local addresses without any registration or assignment process, the addresses still need to follow some rules, as follows:

Key Topic

- Use FD as the first two hex digits.
- Choose a unique 40-bit global ID.
- Append the global ID to FD to create a 48-bit prefix, used as the prefix for all your addresses.
- Use the next 16 bits as a subnet field.
- Note that the structure leaves a convenient 64-bit interface ID field.

Figure 23-11 shows the format of these unique local unicast addresses.

Key Topic

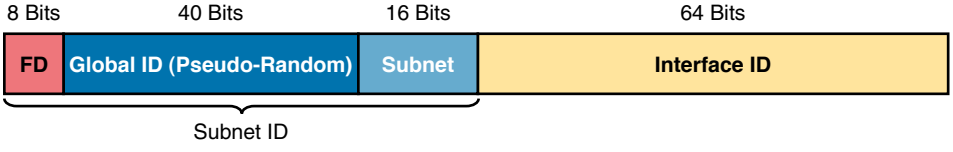


Figure 23-11 *IPv6 Unique Local Unicast Address Format*

NOTE Just to be completely exact, IANA actually reserves prefix FC00::/7, and not FD00::/8, for these addresses. FC00::/7 includes all addresses that begin with hex FC and FD. However, an RFC (4193) requires the eighth bit of these addresses to be set to 1, which means that in practice today, the unique local addresses all begin with their first two digits as FD.

Subnetting with Unique Local IPv6 Addresses

Subnetting using unique local addresses works just like subnetting with global unicast addresses with a 48-bit global routing prefix. The only difference is that with global unicasts, you start by asking for a global routing prefix to be assigned to your company, and that global routing prefix might or might not have a /48 prefix length. With unique local, you create that prefix locally, and the prefix begins with /48, with the first 8 bits set and the next 40 bits randomly chosen.

The process can be as simple as choosing a 40-bit value as your global ID. These 40 bits require 10 hex digits, so you can even avoid thinking in binary and just make up a unique 10-hex-digit value and add hex FD to the front. For example, imagine you chose a 10-hex-digit value of hex 00 0001 0001, prepend a hex FD, making the entire prefix be FD00:0001:0001::/48, or FD00:1:1::/48 when abbreviated.

To create subnets, just as you did in the earlier examples with a 48-bit global routing prefix, treat the entire fourth quartet as a subnet field, as shown in Figure 23-11.

Figure 23-12 shows an example subnetting plan using unique local addresses. The example repeats the same topology shown earlier in Figure 23-9; that figure showed subnetting with a global unicast prefix. This example uses the exact same numbers for the fourth quartet's subnet field, simply replacing the 48-bit global unicast prefix with this new local unique prefix of FD00:1:1.

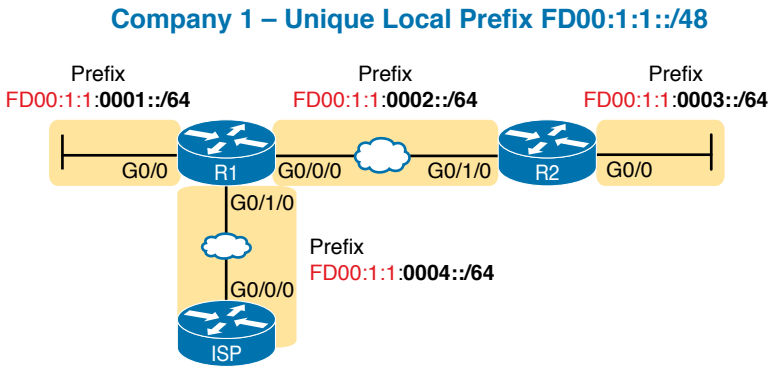


Figure 23-12 *Subnetting Using Unique Local Addresses*

The Need for Globally Unique Local Addresses

The example in Figure 23-12 shows an easy-to-remember prefix of FD00:1:1::/48. Clearly, I made up the easy-to-remember global ID in this example. What global ID would you choose for your company? Would you pick a number that you could not abbreviate and make it shorter? If you had to pick the IPv6 prefix for your unique local addresses from the options in the following list, which would you pick for your company?

- `FDE9:81BE:A059::/48`
- `FDF0:E1D2:C3B4::/48`
- `FD00:1:1::/48`

Given freedom to choose, most people would pick an easy-to-remember, short-to-type prefix, like FD00:1:1::/48. And in a lab or other small network used for testing, making up an easy-to-use number is reasonable. However, for use in real corporate networks, you should not just make up any global ID you like; you should try to follow the unique local address rules that strive to help make your addresses unique in the universe—even without registering a prefix with an ISP or RIR.

RFC 4193 defines unique local addresses, and that RFC stresses the importance of choosing your global ID in a way to make it statistically unlikely to be used by other companies. What is the result of unique global IDs at every company? Making all these unique local addresses unique across the globe. So, if you do plan on using unique local addresses in a real network, plan on using the random number generator logic listed in RFC 4193 to create your prefix.

One of the big reasons to attempt to use a unique prefix, rather than everyone using the same easy-to-remember prefixes, is to be ready for the day that your company merges with

or buys another company. Today, with IPv4, a high percentage of companies use private IPv4 network 10.0.0.0. When they merge their networks, the fact that both use network 10.0.0.0 makes the network merger more painful than if the companies had used different private IPv4 networks. With IPv6 unique local addresses, if both companies did the right thing and randomly chose a prefix, they will most likely be using completely different prefixes, making the merger much simpler. However, companies that take the seemingly easy way out and choose an easy-to-remember prefix like FD00:1:1 greatly increase their risk of requiring extra effort when merging with another company that also chose to use that same prefix.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 23-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 23-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory table		Website

Review All the Key Topics

Key Topic

Table 23-4 Key Topics for Chapter 23

Key Topic Element	Description	Page Number
List	Two types of IPv6 unicast addresses	542
Table 23-2	Values of the initial hex digits of IPv6 addresses, and the address type implied by each	545
Figure 23-6	Subnetting concepts for IPv6 global unicast addresses	547
List	Rules for how to find all IPv6 subnet IDs, given the global routing prefix, and prefix length used for all subnets	548
List	Rules for building unique local unicast addresses	551
Figure 23-11	Subnetting concepts for IPv6 unique local addresses	551

Key Terms You Should Know

global unicast address, global routing prefix, unique local address, subnet ID (prefix ID), subnet router anycast address

CHAPTER 24

Implementing IPv6 Addressing on Routers

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.9 Compare and contrast IPv6 address types

1.9.a Global unicast

1.9.b Unique local

1.9.c Link local

1.9.d Anycast

1.9.e Multicast

1.9.f Modified EUI 64

With IPv4 addressing, some devices, like servers and routers, typically use static predefined IPv4 addresses. End-user devices do not mind if their address changes from time to time, and they typically learn an IPv4 address dynamically using DHCP. IPv6 uses the same approach, with servers, routers, and other devices in the control of the IT group often using predefined IPv6 addresses, and with end-user devices using dynamically learned IPv6 addresses.

This chapter focuses on IPv6 address configuration on routers. The chapter begins with the more obvious IPv6 addressing configuration, with features that mirror IPv4 features, showing how to configure interfaces with IPv6 addresses and view that configuration with **show** commands. The second half of the chapter introduces new IPv6 addressing concepts, showing some other addresses used by routers when doing different tasks.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 24-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Implementing Unicast IPv6 Addresses on Routers	1–3
Special Addresses Used by Routers	4–5

1. Router R1 has an interface named Gigabit Ethernet 0/1, whose MAC address has been set to 0200.0001.000A. Which of the following commands, added in R1's Gigabit Ethernet 0/1 configuration mode, gives this router's G0/1 interface a unicast IPv6 address of 2001:1:1:1:1:200:1:A, with a /64 prefix length?
   a. **ipv6 address 2001:1:1:1:1:200:1:A/64**
   b. **ipv6 address 2001:1:1:1:1:200:1:A/64 eui-64**
   c. **ipv6 address 2001:1:1:1:1:200:1:A /64 eui-64**
   d. **ipv6 address 2001:1:1:1:1:200:1:A /64**
   e. None of the other answers are correct.
2. Router R1 has an interface named Gigabit Ethernet 0/1, whose MAC address has been set to 5055.4444.3333. This interface has been configured with the **ipv6 address 2000:1:1:1::/64 eui-64** subcommand. What unicast address will this interface use?
   a. 2000:1:1:1:52FF:FE55:4444:3333
   b. 2000:1:1:1:5255:44FF:FE44:3333
   c. 2000:1:1:1:5255:4444:33FF:FE33
   d. 2000:1:1:1:200:FF:FE00:0
3. Router R1 currently supports IPv4, routing packets in and out all its interfaces. R1's configuration needs to be migrated to support dual-stack operation, routing both IPv4 and IPv6. Which of the following tasks must be performed before the router can also support routing IPv6 packets? (Choose two answers.)
   a. Enable IPv6 on each interface using an **ipv6 address** interface subcommand.
   b. Enable support for both versions with the **ip versions 4 6** global command.
   c. Additionally enable IPv6 routing using the **ipv6 unicast-routing** global command.
   d. Migrate to dual-stack routing using the **ip routing dual-stack** global command.
4. Router R1 has an interface named Gigabit Ethernet 0/1, whose MAC address has been set to 0200.0001.000A. The interface is then configured with the **ipv6 address 2001:1:1:1:200:FF:FE01:B/64** interface subcommand; no other **ipv6 address** commands are configured on the interface. Which of the following answers lists the link-local address used on the interface?
   a. FE80::FF:FE01:A
   b. FE80::FF:FE01:B
   c. FE80::200:FF:FE01:A
   d. FE80::200:FF:FE01:B

5. Which of the following multicast addresses is defined as the address for sending packets to only the IPv6 routers on the local link?

   a. FF02::1
   b. FF02::2
   c. FF02::5
   d. FF02::A

Foundation Topics

Implementing Unicast IPv6 Addresses on Routers

Every company bases its enterprise network on one or more protocol models, or protocol stacks. In the earlier days of networking, enterprise networks used one or more protocol stacks from different vendors, as shown on the left of Figure 24-1. Over time, companies added TCP/IP (based on IPv4) to the mix. Eventually, companies migrated fully to TCP/IP as the only protocol stack in use.

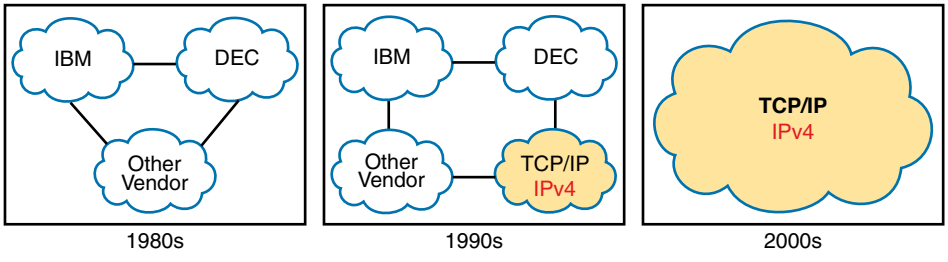


Figure 24-1 *Migration of Enterprise Networks to Use TCP/IP Stack Only, IPv4*

The emergence of IPv6 requires that IPv6 be implemented in end-user hosts, servers, routers, and other devices. However, corporations cannot just migrate all devices from IPv4 to IPv6 over one weekend. Instead, what will likely occur is some kind of long-term migration and coexistence, in which for a large number of years, most corporate networks again use multiple protocol stacks—one based on IPv4 and one based on IPv6.

Eventually, over time, we might all see the day when enterprise networks run only IPv6, without any IPv4 remaining, but that day might take awhile. Figure 24-2 shows the progression, just to make the point, but who knows how long it will take?

One way to add IPv6 support to an established IPv4-based enterprise internetwork is to implement a *dual-stack* strategy. To do so, the routers can be configured to route IPv6 packets, with IPv6 addresses on their interfaces, with a similar model to how routers support IPv4. Then hosts can implement IPv6 when ready, running both IPv4 and IPv6 (dual stacks). The first major section of this chapter shows how to configure and verify unicast IPv6 addresses on routers.

Answers to the "Do I Know This Already?" quiz:

1 A **2** B **3** A, C **4** A **5** B

Key Topic

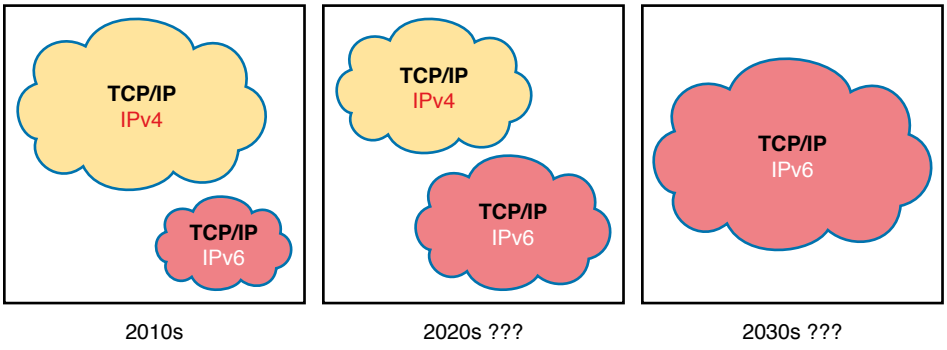


Figure 24-2 *Possible Path Through Dual-Stack (IPv4 and IPv6) over a Long Period*

Static Unicast Address Configuration

Cisco routers give us two options for static configuration of IPv6 addresses. In one case, you configure the full 128-bit address, while in the other, you configure a 64-bit prefix and let the router derive the second half of the address (the interface ID). The next few pages show how to configure both options and how the router chooses the second half of the IPv6 address.

Configuring the Full 128-Bit Address

To statically configure the full 128-bit unicast address—either global unicast or unique local—the router needs an **ipv6 address** *address/prefix-length* interface subcommand on each interface. The address can be an abbreviated IPv6 address or the full 32-digit hex address. The command includes the prefix length value, at the end, with no space between the address and prefix length.

The configuration of the router interface IPv6 address really is that simple. Figure 24-3, along with Examples 24-1 and 24-2, shows a basic example. The figure shows the global unicast IPv6 address used by two different routers, on two interfaces each. As usual, all subnets use a /64 prefix length.

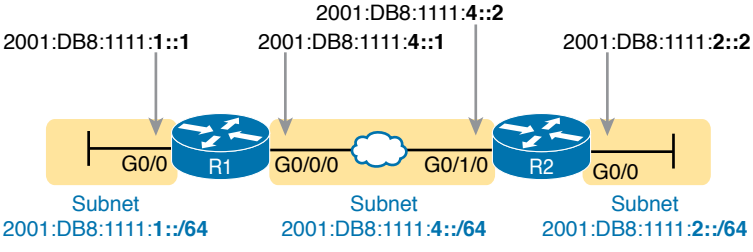


Figure 24-3 *Sample 128-bit IPv6 Addresses to Be Configured on Cisco Router Interfaces*

Example 24-1 *Configuring Static IPv6 Addresses on R1*

```
ipv6 unicast-routing
!
interface GigabitEthernet0/0
 ipv6 address 2001:DB8:1111:1::1/64
!
interface GigabitEthernet0/0/0
 ipv6 address 2001:0db8:1111:0004:0000:0000:0000:0001/64
```

Example 24-2 *Configuring Static IPv6 Addresses on R2*

```
ipv6 unicast-routing
!
interface GigabitEthernet0/0
 ipv6 address 2001:DB8:1111:2::2/64
!
interface GigabitEthernet0/1/0
 ipv6 address 2001:db8:1111:4::2/64
```

NOTE The configuration on R1 in Example 24-1 uses both abbreviated and unabbreviated addresses, and both lowercase and uppercase hex digits, showing that all are allowed. Router **show** commands list the abbreviated value with uppercase hex digits.

Enabling IPv6 Routing

While the configurations shown in Examples 24-1 and 24-2 focus on the IPv6 address configuration, they also include an important but often overlooked step when configuring IPv6 on Cisco routers: IPv6 routing needs to be enabled. On Cisco routers, IPv4 routing is enabled by default, but IPv6 routing is not enabled by default. The solution takes only a single command—**ipv6 unicast-routing**—which enables IPv6 routing on the router.

A router must enable IPv6 globally (**ipv6 unicast-routing**) and enable IPv6 on the interface (**ipv6 address**) before the router will attempt to route IPv6 packets in and out an interface. If you omit the **ipv6 unicast-routing** command but configure interface IPv6 addresses, the router will not route any received IPv6 packets, but the router will act as an IPv6 host. If you include the **ipv6 unicast-routing** command but omit all the interface IPv6 addresses, the router will be ready to route IPv6 packets but have no interfaces that have IPv6 enabled, effectively disabling IPv6 routing.

Verifying the IPv6 Address Configuration

IPv6 uses many **show** commands that mimic the syntax of IPv4 **show** commands. For example:

- The **show ipv6 interface brief** command gives you interface IPv6 address info, but not prefix length info, similar to the IPv4 **show ip interface brief** command.
- The **show ipv6 interface** command gives the details of IPv6 interface settings, much like the **show ip interface** command does for IPv4.

The one notable difference in the most common commands is that the **show interfaces** command still lists the IPv4 address and mask but tells us nothing about IPv6. So, to see IPv6 interface addresses, use commands that begin with **show ipv6**. Example 24-3 lists a few samples from Router R1, with the explanations following.

Example 24-3 *Verifying Static IPv6 Addresses on Router R1*

```
! The first interface is in subnet 1
R1# show ipv6 interface GigabitEthernet 0/0
GigabitEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1:AAFF:FE00:1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:1111:1::1, subnet is 2001:DB8:1111:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds (using 30000)
  ND advertised reachable time is 0 (unspecified)
  ND advertised retransmit interval is 0 (unspecified)
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.

R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::1:AAFF:FE00:1
    2001:DB8:1111:1::1
GigabitEthernet0/1      [administratively down/down]
    unassigned
GigabitEthernet0/0/0    [up/up]
    FE80::32F7:DFF:FE29:8568
    2001:DB8:1111:4::1
GigabitEthernet0/1/0    [administratively down/down]
    unassigned
```

First, focus on the output of the two **show ipv6 interface** commands at the top of the example, which lists interface G0/0, showing output about that interface only. Note that the output lists the configured IPv6 address and prefix length, as well as the IPv6 subnet (2001:DB8:1111:1::/64), which the router calculated based on the IPv6 address.

The end of the example lists the output of the **show ipv6 interface brief** command. Similar to the IPv4-focused **show ip interface brief** command, this command lists IPv6 addresses, but not the prefix length or prefixes. This command also lists all interfaces on the router, whether or not IPv6 is enabled on the interfaces. For example, in this case, the only two interfaces on R1 that have an IPv6 address are G0/0 and G0/0/0, as configured earlier in Example 24-1.

Beyond the IPv6 addresses on the interfaces, the router also adds IPv6 connected routes to the IPv6 routing table off each interface. Just as with IPv4, the router keeps these connected routes in the IPv6 routing table only when the interface is in a working (up/up) state. But if the interface has an IPv6 unicast address configured, and the interface is working, the router adds the connected routes. Example 24-4 shows the connected IPv6 on Router R1 from Figure 24-3.

Example 24-4 *Displaying Connected IPv6 Routes on Router R1*

```
R1# show ipv6 route connected
IPv6 Routing Table - default - 5 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       RL - RPL, O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1
       OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       la - LISP alt, lr - LISP site-registrations, ld - LISP dyn-eid
       lA - LISP away, a - Application
C   2001:DB8:1111:1::/64 [0/0]
     via GigabitEthernet0/0, directly connected
C   2001:DB8:1111:4::/64 [0/0]
     via GigabitEthernet0/0/0, directly connected
```

Generating a Unique Interface ID Using Modified EUI-64

IPv6 follows the same general model as IPv4 regarding which types of devices typically use static, predefined addresses and which use dynamically learned address. For example, routers inside an enterprise use static IPv4 addresses, while end-user devices typically learn their IPv4 address using DHCP. With IPv6, routers also typically use static IPv6 addresses, while user devices use DHCP or Stateless Address Auto Configuration (SLAAC) to dynamically learn their IPv6 address.

Even though engineers typically choose to use stable and predictable IPv6 interface addresses, IOS supports two different methods to configure a stable address. One method uses the **ipv6 address** command to define the entire 128-bit address, as shown in Examples 24-1 and 24-2. The other method uses this same **ipv6 address** command, but the command configures only the 64-bit IPv6 prefix for the interface and lets the router automatically generate a unique interface ID.

This second method uses rules called *modified EUI-64* (extended unique identifier). Often, in the context of IPv6 addressing, people refer to modified EUI-64 as just EUI-64; there is no other term or concept about EUI-64 that you need to know for IPv6. The configuration that uses EUI-64 includes a keyword to tell the router to use EUI-64 rules, along with the 64-bit prefix. The router then uses EUI-64 rules to create the interface ID part of the address, as follows:

Key Topic

1. Split the 6-byte (12-hex-digit) MAC address in two halves (6 hex digits each).
2. Insert FFFE in between the two, making the interface ID now have a total of 16 hex digits (64 bits).
3. Invert the seventh bit of the interface ID.

Figure 24-4 shows the major pieces of how the address is formed.

Key Topic

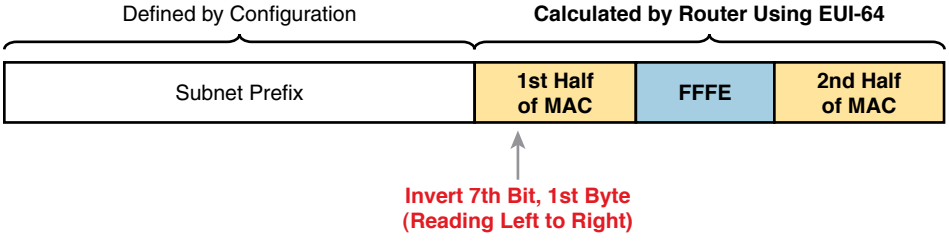


Figure 24-4 *IPv6 Address Format with Interface ID and EUI-64*

NOTE You can find a video about the EUI-64 process on the companion website, in the Chapter Review section for this chapter.

Although this process might seem a bit convoluted, it works. Also, with a little practice, you can look at an IPv6 address and quickly notice the FFFE in the middle of the interface ID and then easily find the two halves of the corresponding interface's MAC address. But you need to be ready to do the same math, in this case to predict the EUI-64 formatted IPv6 address on an interface.

For example, if you ignore the final step of inverting the seventh bit, the rest of the steps just require that you move the pieces around. Figure 24-5 shows two examples, just so you see the process.

Key Topic

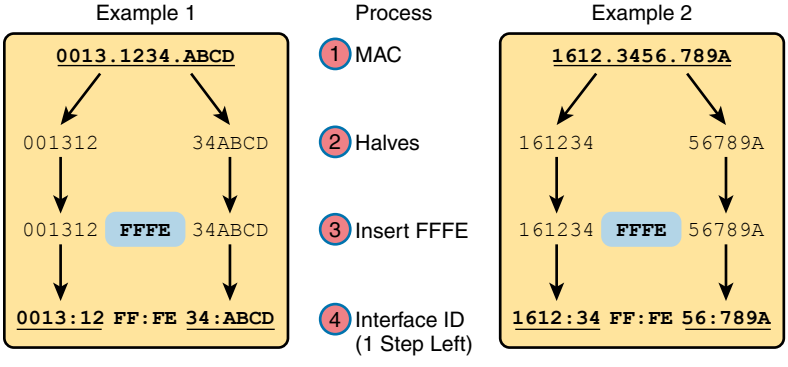


Figure 24-5 *Two Examples of Most of the EUI-64 Interface ID Process*

Both examples follow the same process. Each starts with the MAC address, breaking it into two halves (Step 2). The third step inserts FFFE in the middle, and the fourth step inserts a colon every four hex digits, keeping with IPv6 conventions.

While the examples in Figure 24-5 show most of the steps, they omit the final step. The final step requires that you convert the first byte (first two hex digits) from hex to binary, invert the seventh of the 8 bits, and convert the bits back to hex. Inverting a bit means that if the bit is a 0, make it a 1; if it is a 1, make it a 0. Most of the time, with IPv6 addresses, the original bit will be 0 and will be inverted to a 1.

For example, Figure 24-6 completes the two examples from Figure 24-5, focusing only on the first two hex digits. The examples show each pair of hex digits (Step 1) and the binary equivalent (Step 2). Step 3 shows a copy of those same 8 bits, except the seventh bit is inverted; the example on the left inverts from 0 to 1, and the example on the right inverts from 1 to 0. Finally, the bits are converted back to hex at Step 4.

Key Topic

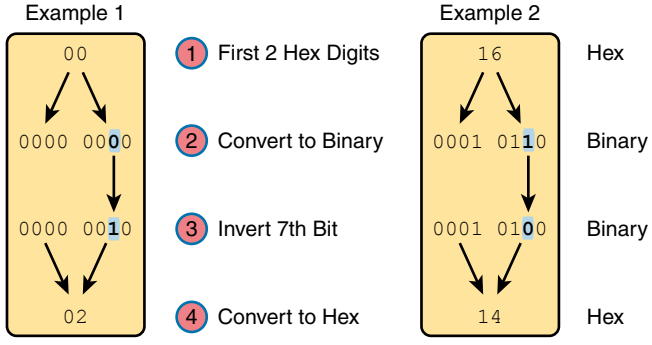


Figure 24-6 *Inverting the Seventh Bit of an EUI-64 Interface ID Field*

NOTE If you do not remember how to do hex-to-binary conversions, take a few moments to review the process. If you memorize the 16 hex values for digits 0 through F, with the corresponding binary values, the conversion can be easy. If you do not have those handy in your memory, take a few moments to look at Table A-2 in Appendix A, "Numeric Reference Tables."

For those of you who prefer the decimal shortcuts, with a little memorization you can do the bit-flip math without doing any hex-binary conversions. First, note that the process to invert the seventh bit, when working with a hexadecimal IPv6 address, flips the third of 4 bits in a single hex digit. With only 16 single hex digits, you could memorize what each hex digit becomes if its third bit is inverted, and you can easily memorize those values with a visual process.

If you want to try to memorize the values, it helps to work through the following process a few times, so grab a piece of scratch paper. Then write the 16 single hex digits as shown on the left side of Figure 24-7. That is, write them in eight rows of two numbers each, with the spacing as directed in the figure.

Next, start at the top of the lists and draw arrow lines between two numbers in the same column on the top left (0 and 2). Then move down the left-side column, connecting the next two digits (4 and 6) with an arrow line, then 8 and A, and then C and E. Repeat the process on the right, re-creating the right side of Figure 24-7.

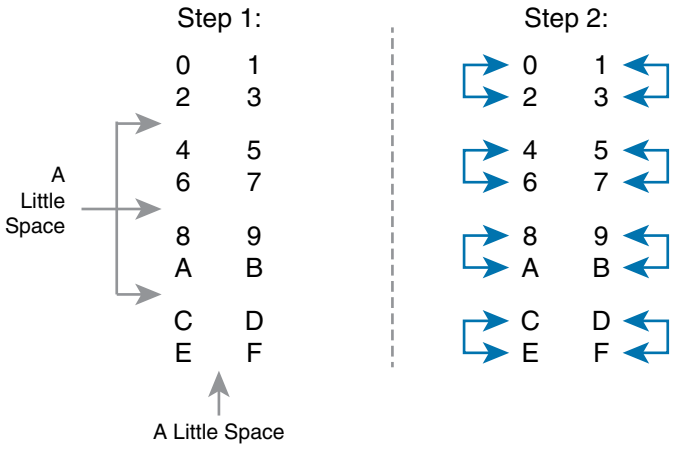


Figure 24-7 *A Mnemonic Device to Help Memorize Bit Inversion Shortcut*

The figure you drew (and the right side of Figure 24-7) shows the hex digits which, when you invert their third bit, convert to the other. That is, 0 converts to 2; 2 converts to 0; 1 converts to 3; 3 converts to 1; 4 converts to 6; 6 converts to 4; and so on. So, on the exam, if you can remember the pattern to redraw Figure 24-7, you could avoid doing binary/hexadecimal conversion. Use whichever approach makes you more comfortable.

As usual, the best way to get comfortable with forming these EUI-64 interface IDs is to calculate some yourself. Table 24-2 lists some practice problems, with an IPv6 64-bit prefix in the first column and the MAC address in the second column. Your job is to calculate the full (unabbreviated) IPv6 address using EUI-64 rules. The answers are at the end of the chapter, in the section "Answers to Earlier Practice Problems."

Table 24-2 IPv6 EUI-64 Address Creation Practice

Prefix	MAC Address	Unabbreviated IPv6 Address
2001:DB8:1:1::/64	0013.ABAB.1001	
2001:DB8:1:1::/64	AA13.ABAB.1001	
2001:DB8:1:1::/64	000C.BEEF.CAFE	
2001:DB8:1:1::/64	B80C.BEEF.CAFE	
2001:DB8:FE:FE::/64	0C0C.ABAC.CABA	
2001:DB8:FE:FE::/64	0A0C.ABAC.CABA	

Configuring a router interface to use the EUI-64 format uses the **ipv6 address** *address/prefix-length* **eui-64** interface subcommand. The **eui-64** keyword tells the router to find the interface MAC address and do the EUI-64 conversion math to find the interface ID.

Example 24-5 shows a revised configuration on Router R1, as compared to the earlier Example 24-1. In this case, R1 uses EUI-64 formatting for its IPv6 addresses.

Example 24-5 *Configuring R1's IPv6 Interfaces Using EUI-64*

```
ipv6 unicast-routing
!
! The ipv6 address command now lists a prefix, not the full address
interface GigabitEthernet0/0
 mac-address 0201.aa00.0001
 ipv6 address 2001:DB8:1111:1::/64 eui-64
!
interface GigabitEthernet0/0/0
 ipv6 address 2001:DB8:1111:4::/64 eui-64

R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::1:AAFF:FE00:1
    2001:DB8:1111:1:1:AAFF:FE00:1
GigabitEthernet0/1      [administratively down/down]
    unassigned
GigabitEthernet0/0/0    [up/up]
    FE80::32F7:DFF:FE29:8568
    2001:DB8:1111:4:32F7:DFF:FE29:8568
GigabitEthernet0/0/1    [administratively down/down]
    unassigned
```

The example uses only Ethernet interfaces, all of which have a universal MAC address to use to create their EUI-64 interface IDs. However, in this case, the configuration includes the **mac-address** command under R1's G0/0 interface, which causes IOS to use the configured MAC address instead of the universal (burned-in) MAC address. Interface G0/0/0 defaults to use its universal MAC address. Following that math:

G0/0 – MAC `0201.AA00.0001` – Interface ID `0001.AAFF.FE00.0001`

G0/0 – MAC `30F7.0D29.8568` – Interface ID `32F7.0DFF.FE29.8568`

Also, be aware that for interfaces that do not have a MAC address, like serial interfaces, the router uses the MAC of the lowest-numbered router interface that does have a MAC.

NOTE When you use EUI-64, the address value in the **ipv6 address** command should be the prefix, not the full 128-bit IPv6 address. However, if you mistakenly type the full address and still use the **eui-64** keyword, IOS accepts the command and converts the address to the matching prefix before putting the command into the running config file. For example, IOS converts **ipv6 address 2000:1:1:1::1/64 eui-64** to **ipv6 address 2000:1:1:1::/64 eui-64**.

Dynamic Unicast Address Configuration

In most cases, network engineers will configure the IPv6 addresses of router interfaces so that the addresses do not change until the engineer changes the router configuration. However, routers can be configured to use dynamically learned IPv6 addresses. These can be

useful for routers connecting to the Internet through some types of Internet access technologies, like DSL and cable modems.

Cisco routers support two ways for the router interface to dynamically learn an IPv6 address to use:

- Stateful DHCP
- Stateless Address Autoconfiguration (SLAAC)

Both methods use the familiar **ipv6 address** command. Of course, neither option configures the actual IPv6 address; instead, the commands configure a keyword that tells the router which method to use to learn its IPv6 address. Example 24-6 shows the configuration, with one interface using stateful DHCP and one using SLAAC.

Example 24-6 *Router Configuration to Learn IPv6 Addresses with DHCP and SLAAC*

```
! This interface uses DHCP to learn its IPv6 address
interface FastEthernet0/0
 ipv6 address dhcp
!
! This interface uses SLAAC to learn its IPv6 address
interface FastEthernet0/1
 ipv6 address autoconfig
```

Special Addresses Used by Routers

IPv6 configuration on a router begins with the simple steps discussed in the first part of this chapter. After you configure the **ipv6 unicast-routing** global configuration command, to enable the function of IPv6 routing, the addition of a unicast IPv6 address on an interface causes the router to do the following:

Key Topic

- Gives the interface a unicast IPv6 address
- Enables the routing of IPv6 packets in/out that interface
- Defines the IPv6 prefix (subnet) that exists off that interface
- Tells the router to add a connected IPv6 route for that prefix, to the IPv6 routing table, when that interface is up/up

NOTE In fact, if you pause and look at the list again, the same ideas happen for IPv4 when you configure an IPv4 address on a router interface.

While all the IPv6 features in this list work much like similar features in IPv4, IPv6 also has a number of additional functions not seen in IPv4. Often, these additional functions use other IPv6 addresses, many of which are multicast addresses. This second major section of the chapter examines the additional IPv6 addresses seen on routers, with a brief description of how they are used.

Link-Local Addresses

IPv6 uses link-local addresses as a special kind of unicast IPv6 address. These addresses are not used for normal IPv6 packet flows that contain data for applications. Instead, these addresses are used by some overhead protocols and for routing. This next topic first looks at how IPv6 uses link-local addresses and then how routers create link-local addresses.

Link-Local Address Concepts

IPv6 defines rules so that packets sent to any link-local address should not be forwarded by any router to another subnet. As a result, several IPv6 protocols make use of link-local addresses when the protocol's messages need to stay within the local LAN. For example, Neighbor Discovery Protocol (NDP), which replaces the functions of IPv4's ARP, uses link-local addresses.

Routers also use link-local addresses as the next-hop IP addresses in IPv6 routes, as shown in Figure 24-8. IPv6 hosts also use a default router (default gateway) concept, like IPv4, but instead of the router address being in the same subnet, hosts refer to the router's link-local address. The **show ipv6 route** command lists the link-local address of the neighboring router, rather than the global unicast or unique local unicast address.

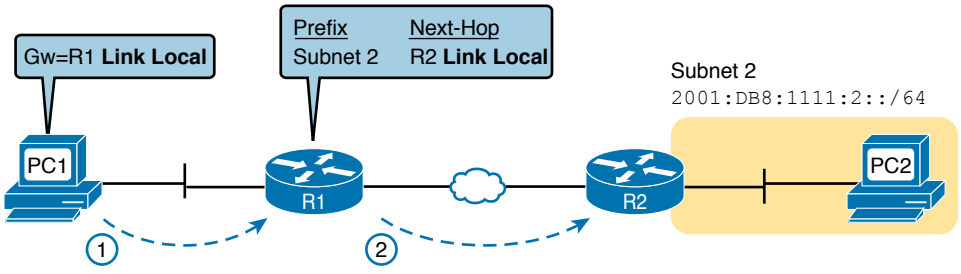


Figure 24-8 *IPv6 Using Link-Local Addresses as the Next-Hop Address*

Following are some key facts about link-local addresses:

Key Topic

Unicast (not multicast): Link-local addresses represent a single host, and packets sent to a link-local address should be processed by only that one IPv6 host.

Forwarding scope is the local link only: Packets sent to a link-local address do not leave the local data link because routers do not forward packets with link-local destination addresses.

Automatically generated: Every IPv6 host interface (and router interface) can create its own link-local address automatically, solving some initialization problems for hosts before they learn a dynamically learned global unicast address.

Common uses: Link-local addresses are used for some overhead protocols that stay local to one subnet and as the next-hop address for IPv6 routes.

Creating Link-Local Addresses on Routers

IPv6 hosts and routers can calculate their own link-local address, for each interface, using some basic rules. First, all link-local addresses start with the same prefix, as shown on the left side of Figure 24-9. By definition, the first 10 bits must match prefix FE80::/10, meaning

that the first three hex digits will be either FE8, FE9, FEA, or FEB. However, when following the RFC, the next 54 bits should be binary 0, so the link-local address should always start with FE80:0000:0000:0000 as the first four unabbreviated quartets.

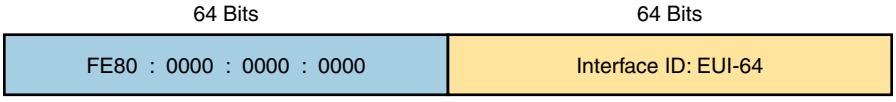


Figure 24-9 *Link-Local Address Format*

The second half of the link-local address, in practice, can be formed using EUI-64 rules, can be randomly generated, or even configured. Cisco routers use the EUI-64 format to create the interface ID (see the earlier section "Generating a Unique Interface ID Using Modified EUI-64"). As a result, a router's complete link-local address should be unique because the MAC address that feeds into the EUI-64 process should be unique.

Alternately, some OSs create their link-local addresses by randomly generating the interface ID. For example, Microsoft OSs use a somewhat random process to choose the interface ID and change it over time in an attempt to prevent some forms of attacks.

IOS creates a link-local address for any interface that has configured at least one other unicast address using the **ipv6 address** command (global unicast or unique local). To see the link-local address, just use the usual commands that also list the unicast IPv6 address: **show ipv6 interface** and **show ipv6 interface brief**. Note that Example 24-7 shows an example from Router R1 just after it was configured as shown in Example 24-5 (with the **eui-64** keyword on the **ipv6 address** commands).

Example 24-7 *Comparing Link-Local Addresses with EUI-Generated Unicast Addresses*

```
R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::1:AAFF:FE00:1
    2001:DB8:1111:1:1:AAFF:FE00:1
GigabitEthernet0/1      [administratively down/down]
    unassigned
GigabitEthernet0/0/0    [up/up]
    FE80::32F7:DFF:FE29:8568
    2001:DB8:1111:4:32F7:DFF:FE29:8568
GigabitEthernet0/0/1    [administratively down/down]
    unassigned
```


First, examine the two pairs of highlighted entries in the example. For each of the two interfaces that have a global unicast address (G0/0 and G0/0/0), the output lists the global unicast, which happens to begin with 2001 in this case. At the same time, the output also lists the link-local address for each interface, beginning with FE80.

Next, focus on the two addresses listed under interface G0/0. If you look closely at the second half of the two addresses listed for interface G0/0, you will see that both addresses have the same interface ID value. The global unicast address was configured in this case with the

ipv6 address 2001:DB8:1111:1::/64 eui-64 command, so the router used EUI-64 logic to form both the global unicast address and the link-local address. The interface MAC address in this case is 0201.AA00.0001, so the router calculates an interface ID portion of both addresses as 0001:AAFF:FE00:0001 (unabbreviated). After abbreviation, Router R1's link-local address on interface G0/0 becomes FE80::AAFF:FE00:1.

IOS can either automatically create the link-local address, or it can be configured. IOS chooses the link-local address for the interface based on the following rules:

- If configured, the router uses the value in the **ipv6 address** *address* **link-local** interface subcommand. Note that the configured link-local address must be from the correct address range for link-local addresses; that is, an address from prefix FE80::/10. In other words, the address must begin with FE8, FE9, FEA, or FEB.
- If not configured, the IOS calculates the link-local address using EUI-64 rules, as discussed and demonstrated in and around Example 24-7. The calculation uses EUI-64 rules even if the interface unicast address does not use EUI-64.

Routing IPv6 with Only Link-Local Addresses on an Interface

This chapter has shown four variations on the **ipv6 address** command so far. To review:

ipv6 address *address/prefix-length*: Static configuration of a specific address

ipv6 address *prefix/prefix-length* **eui-64:** Static configuration of a specific prefix and prefix length, with the router calculating the interface ID using EUI-64 rules

ipv6 address dhcp: Dynamic learning on the address and prefix length using DHCP

ipv6 address autoconfig: Dynamic learning of the prefix and prefix length, with the router calculating the interface ID using EUI-64 rules (SLAAC)

This next short topic completes the list with the following command:

ipv6 enable: Enables IPv6 processing and adds a link-local address, but adds no other unicast IPv6 addresses.

The purpose of the **ipv6 enable** command will not make sense until you realize that some links, particularly WAN links, do not need a global unicast address. Using the backdrop of Figure 24-10, think about the destination of packets sent by hosts like PC1 and PC2. When PC1 sends PC2 an IPv6 packet, the packet holds PC1's and PC2's IPv6 addresses and never contains the WAN link's IPv6 addresses. PC1 and PC2 may need to know the routers' LAN IPv6 addresses, to use as their default gateway, but the hosts do not need to know the routers' WAN interface addresses.

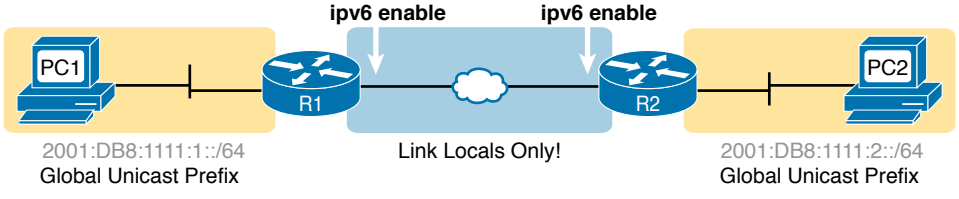


Figure 24-10 *Typical Use of the* **ipv6 enable** *Command*

Additionally, the routers do not need to have global unicast (or unique local) addresses on the WAN links for routing to work. IPv6 routing protocols use link-local addresses as the next-hop address when dynamically building IPv6 routes. Additionally, static routes, as discussed in Chapter 25, "Implementing IPv6 Routing," can use link-local addresses for the next-hop address.

In short, creating a WAN link with no global unicast (or unique local) addresses works. As a result, you would not even need to assign an IPv6 subnet to each WAN link. Then to configure the WAN interfaces, use the **ipv6 enable** command, enabling IPv6 and giving each interface a generated link-local IPv6 address.

To use the command, just configure the **ipv6 enable** command on the interfaces on both ends of the WAN link.

IPv6 Multicast Addresses

IPv6 uses multicast IPv6 addresses for several purposes. Like IPv4, IPv6 includes a range of multicast addresses that can be used by multicast applications, with many of the same fundamental concepts as IPv4 multicasts. For instance, IANA defines the range FF30::/12 (all IPv6 addresses that begin with FF3) as the range of addresses to be used for some types of multicast applications.

Additionally, different IPv6 RFCs reserve multicast addresses for specific purposes. For instance, OSPFv3 uses FF02::5 and FF02::6 as the all-OSPF-routers and all-DR-Routers multicast addresses, respectively, similar to how OSPFv2 uses IPv4 addresses 224.0.0.5 and 224.0.0.6 for the equivalent purposes.

This next section focuses on IPv6 multicast addresses reserved for use with different protocols. The first, link-local multicast addresses, are multicast addresses useful for communicating over a single link. The other type is a special overhead multicast address calculated for each host, called the solicited-node multicast address.

Reserved Multicast Addresses

Stop for a moment and think about some of the control plane protocols discussed throughout this book so far. Some of those IPv4 control plane protocols used IPv4 broadcasts, meaning that the packet destination address was either 255.255.255.255 (the address for all hosts in the local LAN) or the subnet broadcast address (the address for all hosts in that specific subnet). Those broadcast packets were then sent as Ethernet broadcast frames, destined to the Ethernet broadcast address of FFFF.FFFF.FFFF.

While useful, the IPv4 approach of IPv4 broadcast and LAN broadcast requires every host in the VLAN to process the broadcast frame, even if only one other device needed to think about the message. Also, each host has to process the frame, then packet, read the type of message, and so on, before ignoring the task. For example, an IPv4 ARP Request—an IPv4 and LAN broadcast—requires a host to process the Ethernet, IP, and ARP details of the message before deciding whether to reply or not.

IPv6, instead of using Layer 3 and Layer 2 broadcasts, instead uses Layer 3 multicast addresses, which in turn cause Ethernet frames to use Ethernet multicast addresses. As a result:

- All the hosts that should receive the message receive the message, which is necessary for the protocols to work. However…
- …Hosts that do not need to process the message can make that choice with much less processing as compared to IPv4.

For instance, OSPFv3 uses IPv6 multicast addresses FF02::5 and FF02::6. In a subnet, the OSPFv3 routers will listen for packets sent to those addresses. However, all the endpoint hosts do not use OSPFv3 and should ignore those OSPFv3 messages. If a host receives a packet with FF02::5 as the destination IPv6 address, the host can ignore the packet because the host knows it does not care about packets sent to that multicast address. That check takes much less time than the equivalent checks with IPv4.

Table 24-3 lists the most common reserved IPv6 multicast addresses.

Table 24-3 Key IPv6 Local-Scope Multicast Addresses

Short Name	Multicast Address	Meaning	IPv4 Equivalent
All-nodes	FF02::1	All-nodes (all interfaces that use IPv6 that are on the link)	224.0.0.1
All-routers	FF02::2	All-routers (all IPv6 router interfaces on the link)	224.0.0.2
All-OSPF, All-OSPF-DR	FF02::5, FF02::6	All OSPF routers and all OSPF-designated routers, respectively	224.0.0.5, 224.0.0.6
RIPng Routers	FF02::9	All RIPng routers	224.0.0.9
EIGRPv6 Routers	FF02::A	All routers using EIGRP for IPv6 (EIGRPv6)	224.0.0.10
DHCP Relay Agent	FF02::1:2	All routers acting as a DHCPv6 relay agent	None

NOTE An Internet search of "IPv6 Multicast Address Space Registry" will show the IANA page that lists all the reserved values and the RFC that defines the use of each address.

Example 24-8 repeats the output of the **show ipv6 interface** command to show the multicast addresses used by Router R1 on its G0/0 interface. In this case, the highlighted lines show the all-nodes address (FF02::1), all-routers (FF02::2), and two for OSPFv3 (FF02::5 and FF02::6). Note that the IPv6 multicast addresses that the router interface is listening for and processing are listed under the heading "Joined group address(es):" at the top of the highlighted section of the output.

Example 24-8 *Verifying Static IPv6 Addresses on Router R1*

```
R1# show ipv6 interface GigabitEthernet 0/0
GigabitEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:1111:1::1, subnet is 2001:DB8:1111:1::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::5
    FF02::6
    FF02::1:FF00:1
 ! Lines omitted for brevity
```

Multicast Address Scopes

IPv6 RFC 4291 defines IPv6 addressing including the ideas of IPv6 address scope. Each scope defines a different set of rules about whether routers should or should not forward a packet, and how far routers should forward packets, based on those scopes.

For instance, you read earlier in this chapter about the link-local address on an interface—a unicast IPv6 address—but with a link-local scope. The scope definition called "link-local" dictates that packets sent to a link-local unicast address should remain on the link and not be forwarded by any router.

Most of the scope discussion in RFC 4291 applies to multicast addresses, using the term *multicast scope*. Per that RFC, the fourth digit of the multicast address identifies the scope, as noted in Table 24-4.

Key Topic

Table 24-4 IPv6 Multicast Scope Terms

Scope Name	First Quartet	Scope Defined by...	Meaning
Interface-Local	FF01	Derived by Device	Packet remains within the device. Useful for internally sending packets to services running on that same host.
Link-Local	FF02	Derived by Device	Host that creates the packet can send it onto the link, but no routers forward the packet.
Site-Local	FF05	Configuration on Routers	Intended to be more than Link-Local, so routers forward, but must be less than Organization-Local; generally meant to limit packets so they do not cross WAN links.
Organization-Local	FF08	Configuration on Routers	Intended to be broad, probably for an entire company or organization. Must be broader than Site-Local.
Global	FF0E	No Boundaries	No boundaries.

Breaking down the concepts a little further, packets sent to a multicast address with a link-local scope should stay on the local link, that is, the local subnet. Hosts know they can process a link-local packet if received, as do routers. However, routers know to not route the packet to other subnets because of the scope. Packets with an organization-local scope should be routed inside the organization but not out to the Internet or over a link to another company. (Note that routers can predict the boundaries of some scopes, like link-local, but they need configuration to know the boundaries of other scopes, for instance, organization-local.)

Comparing a few of the scopes in terms of where the packets can flow, the higher the value in the fourth hex digit, the further away from the sending host the scope allows the packet to be forwarded. Table 24-4 shows that progression top to bottom, while Figure 24-11 shows an example with three scopes: link-local, site-local, and organization-local. In the figure, site-local messages do not cross the WAN, and organization-local messages do not leave the organization over the link to the Internet.

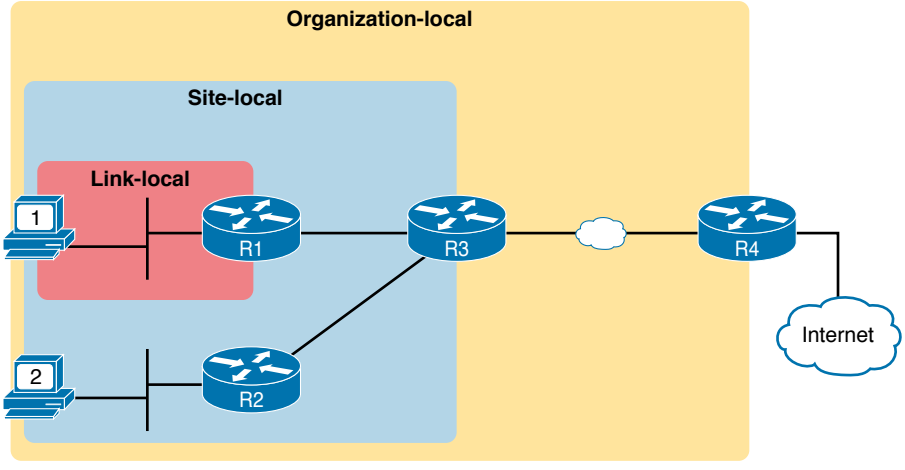


Figure 24-11 *IPv6 Multicast Scopes*

Finally, the term *link-local* has a couple of common uses in IPv6 and can be confusing as a result. The following descriptions should clarify the different uses of the term:

Key Topic

Link-local address: An IPv6 address that begins FE80. This serves as a unicast address for an interface to which devices apply a link-local scope. Devices often create their own link-local addresses using EUI-64 rules. A more complete term for comparison would be *link-local unicast address*.

Link-local multicast address: An IPv6 address that begins with FF02. This serves as a reserved multicast address to which devices apply a link-local scope.

Link-local scope: A reference to the scope itself, rather than an address. This scope defines that routers should not forward packets sent to an address in this scope.

Solicited-Node Multicast Addresses

IPv6 Neighbor Discovery Protocol (NDP) replaces IPv4 ARP, as discussed in Chapter 25. NDP improves the MAC-discovery process by sending IPv6 multicast packets that can be processed by the correct host but discarded with less processing by the rest of the hosts in the subnet. The process uses the solicited-node multicast address associated with the unicast IPv6 address.

Figure 24-12 shows how to determine the solicited node multicast address associated with a unicast address. Start with the predefined /104 prefix (26 hex digits) shown in Figure 24-12. In other words, all the solicited-node multicast addresses begin with the abbreviated FF02::1:FF. In the last 24 bits (6 hex digits), copy the last 6 hex digits of the unicast address into the solicited-node address.

Key Topic

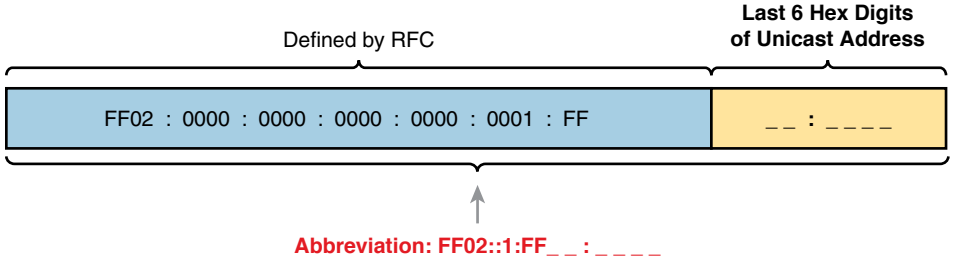


Figure 24-12 *Solicited-Node Multicast Address Format*

Note that a host or router calculates a matching solicited node multicast address for every unicast address on an interface. Example 24-9 shows an example, in which the router interface has a unicast address of 2001:DB8:1111:1::1/64, and a link-local address of FE80::AA:AAAA. As a result, the interface has two solicited node multicast addresses, shown at the end of the output.

Example 24-9 *Verifying Static IPv6 Addresses on Router R1*

```
R1# show ipv6 interface GigabitEthernet 0/0
GigabitEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::AA:AAAA
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:1111:1::1, subnet is 2001:DB8:1111:1::/64 [TEN]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::5
    FF02::1:FF00:1
    FF02::1:FFAA:AAAA
 ! Lines omitted for brevity
```

Note that in this case, R1's global unicast address ends with 00:0001 (unabbreviated), resulting in an unabbreviated solicited node multicast address of FF02:0000:0000:0000:0000:0001:FF00:00001. This value begins with the 26-hex-digit prefix shown in Figure 24-12, followed by 00:0001. The solicited node multicast address corresponding to link-local address FE80::AA:AAAA ends in AA:AAAA and is shown in the last line of the example.

Miscellaneous IPv6 Addresses

Together, this chapter and the preceding chapter have introduced most of the IPv6 addressing concepts included in this book. This short topic mentions a few remaining IPv6 addressing ideas and summarizes the topics for easy study.

First, all IPv6 hosts can use two additional special addresses:

- The unknown (unspecified) IPv6 address, ::, or all 0s
- The loopback IPv6 address, ::1, or 127 binary 0s with a single 1

A host can use the unknown address (::) when its own IPv6 address is not yet known or when the host wonders if its own IPv6 address might have problems. For example, hosts use the unknown address during the early stages of dynamically discovering their IPv6 address. When a host does not yet know what IPv6 address to use, it can use the :: address as its source IPv6 address.

The IPv6 loopback address gives each IPv6 host a way to test its own protocol stack. Just like the IPv4 127.0.0.1 loopback address, packets sent to ::1 do not leave the host but are instead simply delivered down the stack to IPv6 and back up the stack to the application on the local host.

Anycast Addresses

Imagine that routers collectively need to implement some service. Rather than have one router supply that service, that service works best when implemented on several routers. But the hosts that use the service need to contact only the nearest such service, and the network wants to hide all these details from the hosts. Hosts can send just one packet to an IPv6 address, and the routers will forward the packet to the nearest router that supports that service by virtue of supporting that destination IPv6 address.

IPv6 anycast addresses provide that exact function. The *any* part of the name refers to the fact that any instances of the service can be used. Figure 24-13 shows this big concept, with two major steps:

Step 1. Two routers configure the exact same IPv6 address, designated as an anycast address, to support some service.

Step 2. In the future, when any router receives a packet for that anycast address, the other routers simply route the packet to the nearest router that supports the address.

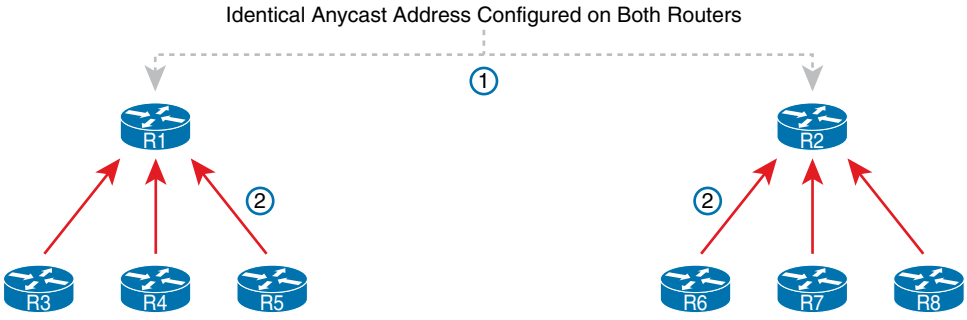


Figure 24-13 *IPv6 Anycast Addresses*

To make this anycast process work, the routers implementing the anycast address must be configured and then advertise a route for the anycast address. The addresses do not come from a special reserved range of addresses; instead, they are from the unicast address range. Often, the address is configured with a /128 prefix so that the routers advertise a host route for that one anycast address. At that point, the routing protocol advertises the route just like any other IPv6 route; the other routers cannot tell the difference.

Example 24-10 shows a sample configuration on a router. Note that the actual address (2001:1:1:2::99) looks like any other unicast address; the value can be chosen like any other IPv6 unicast addresses. However, note the different **anycast** keyword on the **ipv6 address** command, telling the local router that the address has a special purpose as an anycast address. Finally, note that the **show ipv6 interface** command does identify the address as an anycast address, but the **show ipv6 interface brief** command does not.

Example 24-10 *Configuring and Verifying IPv6 Anycast Addresses*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# ipv6 address 2001:1:1:1::1/64
R1(config-if)# ipv6 address 2001:1:1:2::99/128 anycast
R1(config-if)# ^Z
R1#
R1# show ipv6 interface g0/0
GigabitEthernet0/0 is up, line protocol is up
    IPv6 is enabled, link-local address is FE80::11FF:FE11:1111
    No Virtual link-local address(es):
    Global unicast address(es):
       2001:1:1:1::1, subnet is 2001:1:1:1::/64
       2001:1:1:2::99, subnet is 2001:1:1:2::99/128 [ANY]
 ! Lines omitted for brevity
R1# show ipv6 interface brief g0/0
GigabitEthernet0/0 [up/up]
     FE80::11FF:FE11:1111
     2001:1:1:1::1
     2001:1:1:2::99
```

NOTE The *subnet router anycast address* is one special anycast address in each subnet. It is reserved for use by routers as a way to send a packet to any router on the subnet. The address's value in each subnet is the same number as the subnet ID; that is, the address has the same prefix value as the other addresses and all binary 0s in the interface ID.

IPv6 Addressing Configuration Summary

This chapter completes the discussion of various IPv6 address types, while showing how to enable IPv6 on interfaces. Many implementations will use the **ipv6 address** command on each router LAN interface, and either that same command or the **ipv6 enable** command on the WAN interfaces. For exam prep, Table 24-5 summarizes the various commands and the automatically generated IPv6 addresses in one place for review and study.

Key Topic

Table 24-5 Summary of IPv6 Address Types and the Commands That Create Them

Type	Prefix/Address Notes	Enabled with What Interface Subcommand
Global unicast	Many prefixes	**ipv6 address** *address/prefix-length* **ipv6 address** *prefix/prefix-length* **eui-64**
Unique Local	FD00::/8	**ipv6 address** *prefix/prefix-length* **eui-64**
Link local	FE80::/10	**ipv6 address** *address* **link-local** Autogenerated by all **ipv6 address** commands Autogenerated by the **ipv6 enable** command
All hosts multicast	FF02::1	Autogenerated by all **ipv6 address** commands
All routers multicast	FF02::2	Autogenerated by all **ipv6 address** commands
Routing protocol multicasts	Various	Added to the interface when the corresponding routing protocol is enabled on the interface
Solicited-node multicast	FF02::1:FF /104	Autogenerated by all **ipv6 address** commands

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 24-6 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 24-6 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Review memory tables		Website
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 24-7 Key Topics for Chapter 24

Key Topic Element	Description	Page Number
Figure 24-2	Conceptual drawing about the need for dual stacks for the foreseeable future	557
List	Rules for creating an IPv6 address using EUI-64 rules	561
Figure 24-4	IPv6 EUI-64 Address Format and Rules	561
Figure 24-5	Conceptual drawing of how to create an IPv6 address using EUI-64 rules	561
Figure 24-6	Example of performing the bit inversion when using EUI-64	562
List	Functions IOS enables when an IPv6 is configured on a working interface	565
List	Key facts about IPv6 link-local addresses	566
Table 24-4	Link-local scope terms and meanings	571
List	Comparisons of the use of the term *link-local*	572
Figure 24-12	Conceptual drawing of how to make a solicited-node multicast address	573
List	Other special IPv6 addresses	574
Table 24-5	IPv6 address summary with the commands that enable each address type	576

Key Terms You Should Know

anycast address, dual stacks, EUI-64, link-local address, link-local scope, link-local multicast address, site-local scope, organization-local scope, interface-local scope, IPv6 address scope, solicited-node multicast address, all-nodes multicast address, all-routers multicast address, subnet-router anycast address

Additional Practice for This Chapter's Processes

For additional practice with IPv6 abbreviations, you may do the same set of practice problems using your choice of tools:

For additional practice with calculating IPv6 address using EUI-64 rules and finding the solicited-node multicast address based on a unicast address, use the exercises in Appendix H, "Practice for Chapter 24: Implementing IPv6 Addressing on Routers." You have two options to use:

PDF: Navigate to the companion website and open the PDF for Appendix H.

Application: Navigate to the companion website and open the application "Practice Exercise: EUI-64 and Solicited Node Multicast Problems"

Additionally, you can create your own problems using any real router or simulator: Get into the router CLI, into configuration mode, and configure the **mac-address** *address* and **ipv6 address** *prefix*/**64 eui-64** command. Then predict the IPv6 unicast address, link-local address, and solicited-node multicast address; finally, check your predictions against the **show ipv6 interface** command.

Command References

Tables 24-8 and 24-9 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 24-8 Chapter 24 Configuration Command Reference

Command	Description
ipv6 unicast-routing	Global command that enables IPv6 routing on the router.
ipv6 address *ipv6-address*/*prefix-length* [**eui-64**]	Interface subcommand that manually configures either the entire interface IP address or a /64 prefix with the router building the EUI-64 format interface ID automatically.
ipv6 address *ipv6-address*/*prefix-length* [**anycast**]	Interface subcommand that manually configures an address to be used as an anycast address.
ipv6 enable	Command that enables IPv6 on an interface and generates a link-local address.
ipv6 address dhcp	Interface subcommand that enables IPv6 on an interface, causes the router to use DHCP client processes to try to lease an IPv6 address, and creates a link-local address for the interface.

Table 24-9 Chapter 24 EXEC Command Reference

Command	Description
show ipv6 route [connected] [local]	Lists IPv6 routes, or just the connected routes, or just the local routes.
show ipv6 interface [*type number*]	Lists IPv6 settings on an interface, including link-local and other unicast IP addresses (or for the listed interface).
show ipv6 interface brief [*type number*]	Lists interface status and IPv6 addresses for each interface (or for the listed interface).

Answers to Earlier Practice Problems

Table 24-2, earlier in this chapter, listed several practice problems in which you needed to calculate the IPv6 address based on EUI-64 rules. Table 24-10 lists the answers to those problems.

Table 24-10 Answers to IPv6 EUI-64 Address Creation Practice

Prefix	MAC Address	Unabbreviated IPv6 Address
2001:DB8:1:1::/64	0013.ABAB.1001	2001:DB8:1:1:0213:ABFF:FEAB:1001
2001:DB8:1:1::/64	AA13.ABAB.1001	2001:DB8:1:1:A813:ABFF:FEAB:1001
2001:DB8:1:1::/64	000C.BEEF.CAFE	2001:DB8:1:1:020C:BEFF:FEEF:CAFE
2001:DB8:1:1::/64	B80C.BEEF.CAFE	2001:DB8:1:1:BA0C:BEFF:FEEF:CAFE
2001:DB8:FE:FE::/64	0C0C.ABAC.CABA	2001:DB8:FE:FE:0E0C:ABFF:FEAC:CABA
2001:DB8:FE:FE::/64	0A0C.ABAC.CABA	2001:DB8:FE:FE:080C:ABFF:FEAC:CABA

CHAPTER 25

Implementing IPv6 Routing

3.0 IP Connectivity

3.3 Configure and verify IPv4 and IPv6 static routing

3.3.a Default route

3.3.b Network route

3.3.c Host route

3.3.d Floating static

This last chapter in Part VII of the book completes the materials about IPv6 by examining three major topics. The first section examines IPv6 connected and local routes, similar to IPv4, showing how a router adds both connected and local routes based on each interface IPv6 address. The second major section of this chapter then looks at how to configure static IPv6 routes by typing in commands, in this case using the **ipv6 route** command instead of IPv4's **ip route** command. The final major section examines the Neighbor Discovery Protocol (NDP).

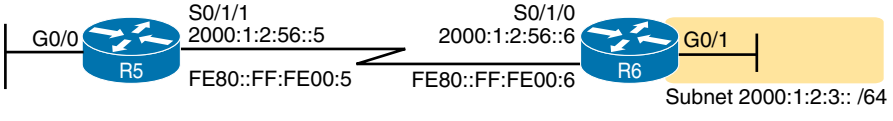
"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 25-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Connected and Local IPv6 Routes	1–2
Static IPv6 Routes	3–6
The Neighbor Discovery Protocol	7–8

Refer to the following figure for questions 1, 3, and 4.



1. A router has been configured with the **ipv6 address 2000:1:2:3::1/64** command on its G0/1 interface as shown in the figure. The router creates a link-local address of FE80::FF:FE00:1 as well. The interface is working. Which of the following routes will the router add to its IPv6 routing table? (Choose two answers.)
   a. A route for 2000:1:2:3::/64
   b. A route for FE80::FF:FE00:1/64
   c. A route for 2000:1:2:3::1/128
   d. A route for FE80::FF:FE00:1/128
2. A router has been configured with the **ipv6 address 3111:1:1:1::1/64** command on its G0/1 interface and **ipv6 address 3222:2:2:2::1/64** on its G0/2 interface. Both interfaces are working. Which of the following routes would you expect to see in the output of the **show ipv6 route connected** command? (Choose two answers.)
   a. A route for 3111:1:1:1::/64
   b. A route for 3111:1:1:1::1/64
   c. A route for 3222:2:2:2::/64
   d. A route for 3222:2:2:2::2/128
3. An engineer needs to add a static IPv6 route for prefix 2000:1:2:3::/64 to Router R5's configuration, in the figure shown with question 1. Which of the following answers shows a valid static IPv6 route for that subnet, on Router R5?
   a. **ipv6 route 2000:1:2:3::/64 S0/1/1**
   b. **ipv6 route 2000:1:2:3::/64 S0/1/0**
   c. **ip route 2000:1:2:3::/64 S0/1/1**
   d. **ip route 2000:1:2:3::/64 S0/1/0**
4. An engineer needs to add a static IPv6 route for prefix 2000:1:2:3::/64 to Router R5 in the figure shown with question 1. Which of the following answers shows a valid static IPv6 route for that subnet on Router R5?
   a. **ipv6 route 2000:1:2:3::/64 2000:1:2:56::5**
   b. **ipv6 route 2000:1:2:3::/64 2000:1:2:56::6**
   c. **ipv6 route 2000:1:2:3::/64 FE80::FF:FE00:5**
   d. **ipv6 route 2000:1:2:3::/64 FE80::FF:FE00:6**

5. An engineer types the command **ipv6 route 2001:DB8:8:8::/64 2001:DB8:9:9::9 129** in configuration mode of Router R1 and presses **Enter**. Later, a **show ipv6 route** command does not list any route for subnet 2001:DB8:8:8::/64. Which of the following could have caused the route to not be in the IPv6 routing table?

   a. The command should be using a next-hop link-local address instead of a global unicast.

   b. The command is missing an outgoing interface parameter, so IOS rejected the **ipv6 route** command.

   c. The router has no routes that match 2001:DB8:9:9::9.

   d. A route for 2001:DB8:8:8::/64 with administrative distance 110 already exists.

6. The command output shows two routes from the longer output of the **show ipv6 route** command. Which answers are true about the output? (Choose two answers.)

```
R1# show ipv6 route static
! Legend omitted for brevity
S 2001:DB8:2:2::/64 [1/0]
 via 2001:DB8:4:4::4
S ::/0 [1/0]
 via Serial0/0/1, directly connected
```

   a. The route to ::/0 is added because of an **ipv6 route** global command.

   b. The administrative distance of the route to 2001:DB8:2:2::/64 is 1.

   c. The route to ::/0 is added because of an **ipv6 address** interface subcommand.

   d. The route to 2001:DB8:2:2::/64 is added because of an IPv6 routing protocol.

7. PC1, PC2, and Router R1 all connect to the same VLAN and IPv6 subnet. PC1 wants to send its first IPv6 packet to PC2. What protocol or message will PC1 use to discover the MAC address to which PC1 should send the Ethernet frame that encapsulates this IPv6 packet?

   a. ARP

   b. NDP NS

   c. NDP RS

   d. SLAAC

8. Which of the following pieces of information does a router supply in an NDP Router Advertisement (RA) message? (Choose two answers.)

   a. Router IPv6 address

   b. Host name of the router

   c. IPv6 prefix(es) on the link

   d. IPv6 address of DHCP server

Foundation Topics

Connected and Local IPv6 Routes

A Cisco router adds IPv6 routes to its IPv6 routing table for several reasons. Many of you could predict those reasons at this point in your reading, in part because the logic mirrors the logic routers use for IPv4. Specifically, a router adds IPv6 routes based on the following:

Key Topic

- The configuration of IPv6 addresses on working interfaces (connected and local routes)
- The direct configuration of a static route (static routes)
- The configuration of a routing protocol, like OSPFv3, on routers that share the same data link (dynamic routes)

The first two sections of this chapter examine the first of these two topics, with discussions of IPv6 routing protocols now residing in the CCNP Enterprise exams.

Rules for Connected and Local Routes

Routers add and remove connected routes and local routes, based on the interface configuration and the interface state. First, the router looks for any configured unicast addresses on any interfaces by looking for the **ipv6 address** command. Then, if the interface is working—if the interface has a "line status is up, protocol status is up" notice in the output of the **show interfaces** command—the router adds both a connected and local route.

NOTE Routers do not create connected or local IPv6 routes for link-local addresses.

The connected and local routes themselves follow the same general logic as with IPv4. The connected route represents the subnet connected to the interface, whereas the local route is a host route for only the specific IPv6 address configured on the interface.

As an example, consider a router, with a working interface, configured with the **ipv6 address 2000:1:1:1::1/64** command. The router will calculate the subnet ID based on this address and prefix length, and it will place a connected route for that subnet (2000:1:1:1::/64) into the routing table. The router also takes the listed IPv6 address and creates a host route for that address, with a /128 prefix length. (With IPv4, host routes have a /32 prefix length, while IPv6 uses a /128 prefix length, meaning "exactly this one address.")

The following list summarizes the rules about how routers create routes based on the configuration of an interface IPv6 unicast address, for easier review and study:

Key Topic

1. Routers create IPv6 routes based on each unicast IPv6 address on an interface, as configured with the **ipv6 address** command, as follows:
   - **A.** The router creates a route for the subnet (a connected route).
   - **B.** The router creates a host route (/128 prefix length) for the router IPv6 address (a local route).
2. Routers do not create routes based on the link-local addresses associated with the interface.
3. Routers remove the connected and local routes for an interface if the interface fails, and they re-add these routes when the interface is again in a working (up/up) state.

Example of Connected IPv6 Routes

While the concept of connected and local IPv6 routes works much like IPv4 routes, seeing a few examples can certainly help. To show some sample routes, Figure 25-1 gives the details of one sample internetwork used in this chapter. The figure shows the IPv6 subnet IDs. The upcoming examples focus on the connected and local routes on Router R1.

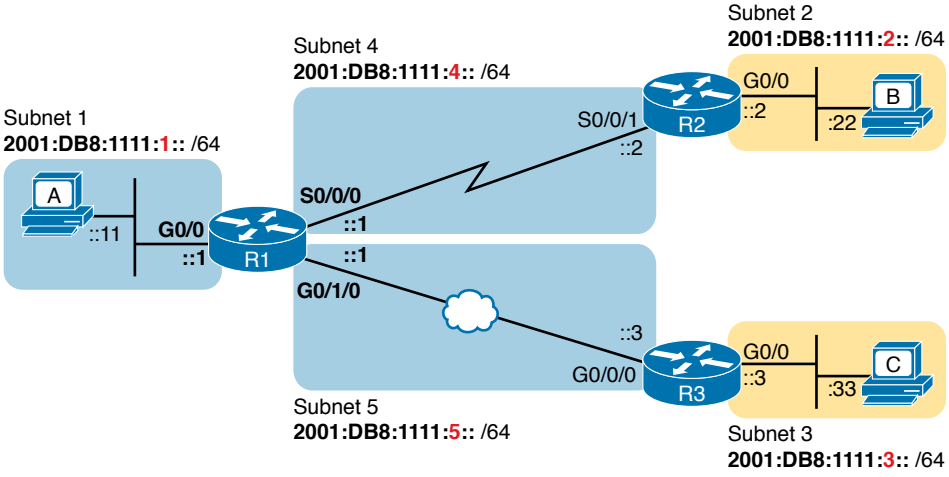


Figure 25-1 *Sample Network Used to Show Connected and Local Routes*

To clarify the notes in Figure 25-1, note that the figure shows IPv6 prefixes (subnets), with a shorthand notation for the interface IPv6 addresses. The figure shows only the abbreviated interface ID portion of each interface address near each interface. For example, R1's G0/0 interface address would begin with subnet ID value 2001:DB8:1111:1, added to ::1, for 2001:DB8:1111:1::1.

Now on to the example of connected routes. To begin, consider the configuration of Router R1 from Figure 25-1, as shown in Example 25-1. The excerpt from the **show running-config** command on R1 shows three interfaces, all of which are working. Also note that no static route or routing protocol configuration exists.

Example 25-1 *IPv6 Addressing Configuration on Router R1*

```
ipv6 unicast-routing
!
interface GigabitEthernet0/0
 ipv6 address 2001:DB8:1111:1::1/64
!
interface Serial0/0/0
 ipv6 address 2001:db8:1111:4::1/64
!
interface GigabitEthernet0/1/0
 ipv6 address 2001:db8:1111:5::1/64
```

Answers to the "Do I Know This Already?" quiz:

1 A, C **2** A, C **3** A **4** B **5** C **6** A, B **7** B **8** A, C

Based on Figure 25-1 and Example 25-1, R1 should have three connected IPv6 routes, as highlighted in Example 25-2.

Example 25-2 *Routes on Router R1 Before Adding Static Routes or Routing Protocols*

```
R1# show ipv6 route
IPv6 Routing Table - default - 7 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       RL - RPL, O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1
       OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       la - LISP alt, lr - LISP site-registrations, ld - LISP dyn-eid
       lA - LISP away, a - Application
C   2001:DB8:1111:1::/64 [0/0]
     via GigabitEthernet0/0, directly connected
L   2001:DB8:1111:1::1/128 [0/0]
     via GigabitEthernet0/0, receive
C   2001:DB8:1111:4::/64 [0/0]
     via Serial0/0/0, directly connected
L   2001:DB8:1111:4::1/128 [0/0]
     via GigabitEthernet0/0/0, receive
C   2001:DB8:1111:5::/64 [0/0]
     via GigabitEthernet0/1/0, directly connected
L   2001:DB8:1111:5::1/128 [0/0]
     via GigabitEthernet0/1/0, receive
L   FF00::/8 [0/0]
     via Null0, receive
```

All three highlighted routes show the same basic kinds of information, so for discussion, focus on the first pair of highlighted lines, which detail the connected route for subnet 2001:DB8:1111:1::/64. The first pair of highlighted lines state: The route is a "directly connected" route; the interface ID is GigabitEthernet0/0; and the prefix/length is 2001:DB8:1111:1::/64. At the far left, the code letter "C" identifies the route as a connected route (per the legend above). Also note that the numbers in brackets mirror the same ideas as IPv4's **show ip route** command: The first number represents the administrative distance, and the second is the metric.

Examples of Local IPv6 Routes

Continuing this same example, three local routes should exist on R1 for the same three interfaces as the connected routes. Indeed, that is the case, with one extra local route for other purposes. Example 25-3 shows only the local routes, as listed by the **show ipv6 route local** command, with highlights of one particular local route for discussion.

Example 25-3 *Local IPv6 Routes on Router R1*

```
R1# show ipv6 route local
! Legend omitted for brevity

L   2001:DB8:1111:1::1/128 [0/0]
     via GigabitEthernet0/0, receive
L   2001:DB8:1111:4::1/128 [0/0]
     via Serial0/0/0, receive
L   2001:DB8:1111:5::1/128 [0/0]
     via GigabitEthernet0/1/0, receive
L   FF00::/8 [0/0]
     via Null0, receive
```

For the highlighted local route, look for a couple of quick facts. First, look back to R1's configuration in Example 25-1, and note R1's IPv6 address on its G0/0 interface. This local route lists the exact same address. Also note the /128 prefix length, meaning this route matches packets sent to that address (2001:DB8:1111:1::1), and only that address.

NOTE While the **show ipv6 route local** command shows all local IPv6 routes, the **show ipv6 route connected** command shows all connected routes.

Static IPv6 Routes

While routers automatically add connected and local routes based on the interface configuration, static routes require direct configuration with the **ipv6 route** command. Simply put, someone configures the command, and the router places the details from the command into a route in the IPv6 routing table.

The **ipv6 route** command follows the same general logic as does IPv4's **ip route** command, as discussed in Chapter 16, "Configuring IPv4 Addressing and Static Routes." For IPv4, the **ip route** command starts by listing the subnet ID and mask, so for IPv6, the **ipv6 route** command begins with the prefix and prefix length. Then the respective commands list the directions of how this router should forward packets toward that destination subnet or prefix by listing the outgoing interface or the address of the next-hop router.

Figure 25-2 shows the concepts behind a single **ipv6 route** command, demonstrating the concepts behind a static route on Router R1 for the subnet on the right (subnet 2, or 2001:DB8:1111:2::/64). A static route on R1, for this subnet, will begin with **ipv6 route 2001:DB8:1111:2::/64**, followed by either the outgoing interface (S0/0/0) or the next-hop IPv6 address, or both.

Now that you understand the big ideas with IPv6 static routes, the next few pages walk you through a series of examples. In particular, the examples look at configuring static routes with an outgoing interface, then with a next-hop global unicast address, and then with a next-hop link-local address. This section ends with a discussion of static IPv6 default routes.

Key Topic

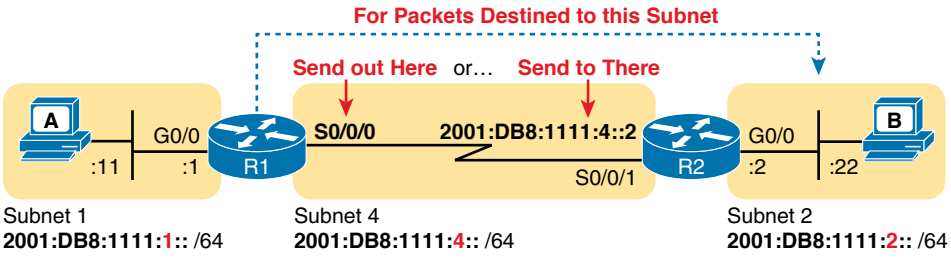


Figure 25-2 *Logic Behind IPv6 Static Route Commands (IPv6 Route)*

Static Routes Using the Outgoing Interface

This first IPv6 static route example uses the outgoing interface option. As a reminder, for both IPv4 and IPv6 static routes, when the command references an interface, the interface is a local interface. That is, it is an interface on the router where the command is added. In this case, as shown in Figure 25-2, R1's **ipv6 route** command would use interface S0/0/0, as shown in Example 25-4.

Example 25-4 *Static IPv6 Routes on Router R1*

```
! Static route on router R1
R1(config)# ipv6 route 2001:db8:1111:2::/64 S0/0/0
```

While Example 25-4 shows the correct syntax of the route, if using static routes throughout this internetwork, more static routes are needed. For example, to support traffic between hosts A and B, R1 is now prepared. Host A will forward all its IPv6 packets to its default router (R1), and R1 can now route those packets out S0/0/0 to R2 next. However, Router R2 does not yet have a route back to host A's subnet, subnet 1 (2001:DB8:1111:1::/64), so a complete solution requires more routes.

Example 25-5 solves this problem by giving Router R2 a static route for subnet 1 (2001:DB8:1111:1::/64). After this route is added, hosts A and B should be able to ping each other.

Example 25-5 *Static IPv6 Routes on Router R2*

```
! Static route on router R2
R2(config)# ipv6 route 2001:db8:1111:1::/64 s0/0/1
```

Many options exist for verifying the existence of the static route and testing whether hosts can use the route. **ping** and **traceroute** can test connectivity. From the router command line, the **show ipv6 route** command will list all the IPv6 routes. The shorter output of the **show ipv6 route static** command, which lists only static routes, could also be used; Example 25-6 shows that output, with the legend omitted.

Example 25-6 *Verification of Static Routes Only on R1*

```
R1# show ipv6 route static
! Legend omitted for brevity
S   2001:DB8:1111:2::/64 [1/0]
     via Serial0/0/0, directly connected
```

This command lists many facts about the one static route on R1. First, the code "S" in the left column does identify the route as a static route. (However, the later phrase "directly connected" might mislead you to think this is a connected route; trust the "S" code.) Note that the prefix (2001:DB8:1111:2::/64) matches the configuration (in Example 25-4), as does the outgoing interface (S0/0/0).

While this command lists basic information about each static route, it does not state whether this route would be used when forwarding packets to a particular destination. For example, if host A sent an IPv6 packet to host B (2001:DB8:1111:2::22), would R1 use this static route? As it turns out, R1 would use that route, as confirmed by the **show ipv6 route 2001:DB8:1111:2::22** command. This command asks the router to list the route that the router would use when forwarding packets to that particular address. Example 25-7 shows an example.

Example 25-7 *Displaying the Route R1 Uses to Forward to Host B*

```
R1# show ipv6 route 2001:db8:1111:2::22
Routing entry for 2001:DB8:1111:2::/64
  Known via "static", distance 1, metric 0
  Route count is 1/1, share count 0
  Routing paths:
    directly connected via Serial0/0/0
      Last updated 00:01:29 ago
```

Static Routes Using Next-Hop IPv6 Address

The previous example used a serial WAN link on purpose. With a point-to-point WAN link, the **ipv6 route** command can use the outgoing interface style of configuration Static IPv6 routes that refer to a next-hop address have two options: the unicast address on the neighboring router (global unicast or unique local) or the link-local address of that same neighboring router. Figure 25-3 spells out those two options with an updated version of Figure 25-2, this time showing Router R2's global unicast as well as R2's link-local address.

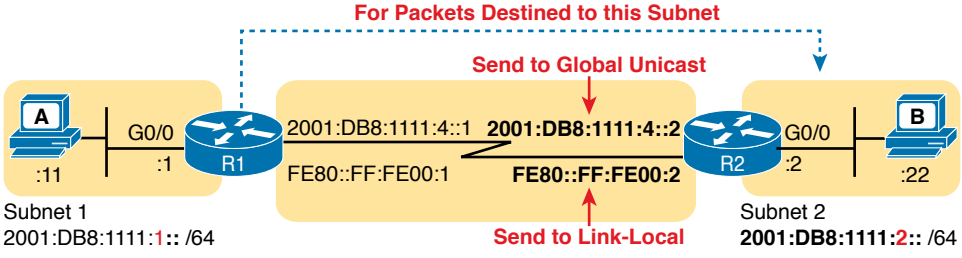


Figure 25-3 *Using Unicast or Link-Local as the Next-Hop Address for Static Routes*

The next few pages walk you through examples, first with a global unicast as a next-hop and then with a link-local as a next-hop.

Example Static Route with a Global Unicast Next-Hop Address

This example uses the internetwork shown in Figure 25-3, but with the earlier static routes removed. That is, both routers have only connected and local routes to begin the example.

In Example 25-8, both R1 and R2 add static routes that refer to the neighbor's global unicast address. R1 adds a route for subnet 2 (on the right), while R2 adds a route for subnet 1 (on the left). Note that the example shows routes in both directions so that the two hosts can send packets to each other.

Example 25-8 *Static IPv6 Routes Using Global Unicast Addresses*

```
! The first command is on router R1, listing R2's global unicast address
R1(config)# ipv6 route 2001:db8:1111:2::/64 2001:DB8:1111:4::2
```

```
! The next command is on router R2, listing R1's global unicast address
R2(config)# ipv6 route 2001:db8:1111:1::/64 2001:db8:1111:4::1
```

The **ipv6 route** command itself is relatively straightforward. Focus on R1's route, which matches the logic shown in Figure 25-3. The command lists subnet 2 (2001:DB8:1111:2::/64). It then lists R2's global unicast address (ending in 4::2).

The verification commands on R1, as shown in Example 25-9, list the usual information. Example 25-9 shows two commands, first listing R1's only static route (the one configured in Example 25-8). The end of the example lists the **show ipv6 route 2001:DB8:1111:2::22** command, which lists the route R1 uses when forwarding packets to Host B, proving that R1 uses this new static route when forwarding packets to that host.

Example 25-9 *Verification of Static Routes to a Next-Hop Global Unicast Address*

```
R1# show ipv6 route static
! Legend omitted for brevity
S   2001:DB8:1111:2::/64 [1/0]
     via 2001:DB8:1111:4::2

R1# show ipv6 route 2001:db8:1111:2::22/64
Routing entry for 2001:DB8:1111:2::/64
  Known via "static", distance 1, metric 0
  Backup from "ospf 1 [110]"
  Route count is 1/1, share count 0
  Routing paths:
    2001:DB8:1111:4::2
      Last updated 00:07:43 ago
```


Example Static Route with a Link-Local Next-Hop Address

Static routes that refer to a neighbor's link-local address work a little like both of the preceding two styles of static routes. First, the **ipv6 route** command refers to a next-hop address,

namely a link-local address. However, the command must also refer to the router's local outgoing interface. Why both? The **ipv6 route** command cannot simply refer to a link-local next-hop address by itself because the link-local address does not, by itself, tell the local router which outgoing interface to use.

Interestingly, when the **ipv6 route** command refers to a global unicast next-hop address, the router can deduce the outgoing interface. For example, the earlier example on R1, as shown in Example 25-8, shows R1 with a static IPv6 route with a next-hop IPv6 address of 2001:DB8:1111:4::2. R1 can look at its IPv6 routing table, see its connected route that includes this 2001:DB8:1111:4::2 address, and see a connected route off R1's S0/0/0. As a result, with a next-hop global unicast address, R1 can deduce the correct outgoing interface (R1's S0/0/0).

With a link-local next-hop address, a router cannot work through this same logic, so the outgoing interface must also be configured. Example 25-10 shows the configuration of static routes on R1 and R2, replacements for the two routes previously configured in Example 25-8.

Example 25-10 *Static IPv6 Routes Using Link-Local Neighbor Addresses*

```
! The first command is on router R1, listing R2's link-local address
R1(config)# ipv6 route 2001:db8:1111:2::/64 S0/0/0 FE80::FF:FE00:2
```

```
! The next command is on router R2, listing R1's link-local address
R2(config)# ipv6 route 2001:db8:1111:1::/64 S0/0/1 FE80::FF:FE00:1
```

Example 25-11 verifies the configuration in Example 25-10 by repeating the **show ipv6 route static** and **show ipv6 route 2001:DB8:1111:2::22** commands used in Example 25-9. Note that the output from both commands differs slightly in regard to the forwarding details. Because the new commands list both the next-hop address and outgoing interface, the **show** commands also list both the next-hop (link-local) address and the outgoing interface. If you refer back to Example 25-9, you will see only a next-hop address listed.

Example 25-11 *Verification of Static Routes to a Next-Hop Link-Local Address*

```
R1# show ipv6 route static
! Legend omitted for brevity

S   2001:DB8:1111:2::/64 [1/0]
      via FE80::FF:FE00:2, Serial0/0/0

R1# show ipv6 route 2001:db8:1111:2::22
Routing entry for 2001:DB8:1111:2::/64
  Known via "static", distance 1, metric 0
  Backup from "ospf 1 [110]"
  Route count is 1/1, share count 0
  Routing paths:
    FE80::FF:FE00:2, Serial0/0/0
      Last updated 00:08:10 ago
```

Static Routes over Ethernet Links

You might have wondered why the chapter shows examples with a serial link, knowing that most networks use fewer and fewer serial links today. Using serial links in the examples avoids one complication when defining static routes that use Ethernet interfaces (LAN or WAN). The next example discusses the issues and shows configuration options for static routes when the outgoing interface is an Ethernet interface.

To configure a static route that uses an Ethernet interface, the **ipv6 route** command's forwarding parameters should always include a next-hop IPv6 address. IOS allows you to configure the **ipv6 route** command using only the outgoing-interface parameter, without listing a next-hop address. The router will accept the command; however, if that outgoing interface happens to be an Ethernet interface, the router cannot successfully forward IPv6 packets using the route.

To configure the **ipv6 route** correctly when directing packets out an Ethernet interface, the configuration should use one of these styles:

- Refer to the next-hop global unicast address (or unique local address) only
- Refer to both the outgoing interface and next-hop global unicast address (or unique local address)
- Refer to both the outgoing interface and next-hop link-local address

Example 25-12 shows a sample configuration from routers R1 and R3 in Figure 25-4. The top part of the figure shows the details for R1's route to the subnet on the right side of the figure, with the details labeled with an "A." The bottom half shows the details for R3's route to the LAN subnet on the left of the figure, labeled with a "B."

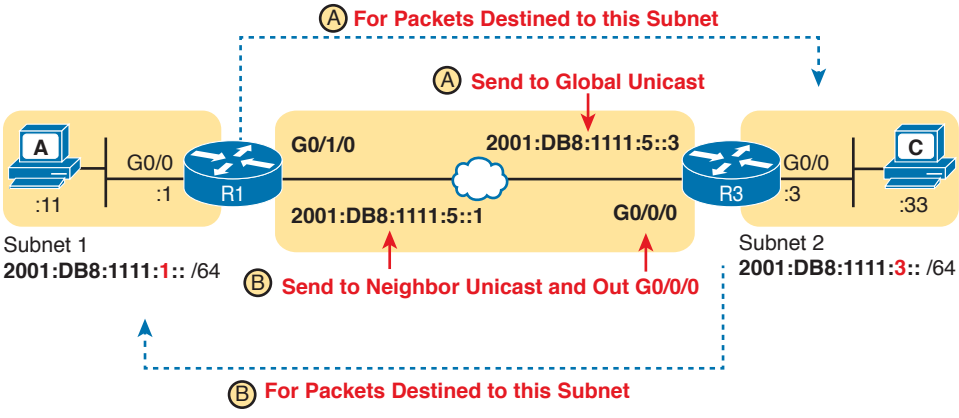


Figure 25-4 *Network Details for IPv6 Static Routes on an Ethernet Interface*

Example 25-12 *Static IPv6 Routes with an Ethernet WAN Interface*

```
! The first command is on router R1, listing R3's global unicast address
R1(config)# ipv6 route 2001:db8:1111:3::/64 2001:db8:1111:5::3
```

```
! The next command is on router R3, listing R1's link-local address
R3(config)# ipv6 route 2001:db8:1111:1::/64 G0/0/0 2001:db8:1111:5::1
```

Static Default Routes

IPv6 supports a default route concept, similar to IPv4. The default route tells the router what to do with an IPv6 packet when the packet matches no other IPv6 route. The logic is pretty basic:

- With no default route, the router discards the IPv6 packet.
- With a default route, the router forwards the IPv6 packet based on the default route.

Default routes can be particularly useful in a couple of network design cases. For example, with an enterprise network design that uses a single router at each branch office, with one WAN link to each branch, the branch routers have only one possible path over which to forward packets. In a large network, when using a routing protocol, the branch router could learn thousands of routes—all of which point back toward the core of the network over that one WAN link.

Branch routers could use default routes instead of a routing protocol. The branch router would forward all traffic to the core of the network. Figure 25-5 shows just such an example, with two sample branch routers on the right and a core site router on the left.

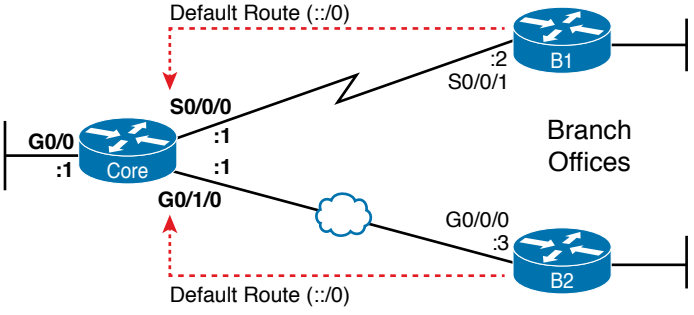


Figure 25-5 *Using Static Default Routes at Branches to Forward Back to the Core*

To configure a static default route, use the same rules already discussed in this section of the chapter, but use a specific value to note the route as a default route: ::/0. Taken literally, the double colon (::) is the IPv6 abbreviation for all 0s, and the /0 means the prefix length is 0. This idea mirrors the IPv4 convention to refer to the default route as 0.0.0.0/0. Otherwise, just configure the **ipv6 route** command as normal.

Example 25-13 shows one such sample static default route on Router B1 from Figure 25-5. This example uses the outgoing interface option.

Example 25-13 *Static Default Route for Branch Router B1*

```
!Forward out B1's S0/0/1 local interface...
B1(config)# ipv6 route ::/0 S0/0/1
```

With IPv6, the router displays the default a little more cleanly than with IPv4. The **show ipv6 route** command simply includes the route in the output of the command, along with the other routes. Example 25-14 shows an example, with "::/0" listed to denote this route as the default route.

Example 25-14 *Router B1's Static Default Route (Using Outgoing Interface)*

```
B1# show ipv6 route static
IPv6 Routing Table - default - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, R - RIP, I1 - ISIS L1, I2 - ISIS L2
       IA - ISIS interarea, IS - ISIS summary, D - EIGRP, EX - EIGRP external
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   ::/0 [1/0]
     via Serial0/0/1, directly connected
```

Static IPv6 Host Routes

Both IPv4 and IPv6 allow the definition of static host routes—that is, a route to a single host IP address. With IPv4, those routes use a /32 mask, which identifies a single IPv4 address in the **ip route** command; with IPv6, a /128 mask identifies that single host in the **ipv6 route** command.

A host route follows the same rules as a route for any other IPv6 subnet. For instance, if you refer back to Figure 25-3, host B sits on the right side of the figure. Earlier examples showed R1's static routes for the subnet in which host B resides—for example, the routes for Router R1 in Examples 25-8 and 25-10. To create a host route on R1, referring to host B's specific IPv6 address, just change those commands to refer to host B's entire IPv6 address (2001:DB8:1111:2::22), with prefix length /128.

Example 25-15 shows two sample host routes on Router R1. Both define a host route to host B's IPv6 address as seen in Figure 25-3. One route uses Router R2's link-local address as the next-hop address, and one route uses R2's global unicast address as the next-hop address.

Example 25-15 *Static Host IPv6 Routes on R1, for Host B*

```
! The first command lists host B's address, prefix length /128,
! with R2's link-local address as next-hop, with an outgoing interface.
R1(config)# ipv6 route 2001:db8:1111:2::22/128 S0/0/0 FE80::FF:FE00:2
R1(config)#
! The next command also lists host B's address, prefix length /128,
! but with R2's global unicast address as next-hop, and no outgoing interface.
R1(config)# ipv6 route 2001:db8:1111:2::22/128 2001:DB8:1111:4::2
```

Floating Static IPv6 Routes

Next, consider the case in which a static route competes with other static routes or routes learned by a routing protocol. For example, consider the topology shown in Figure 25-6, which shows a branch office with two WAN links: one very fast Gigabit Ethernet link and one rather slow (but cheap) T1. In this design, the network uses OSPFv3 to learn IPv6 routes over the primary link, learning a route for subnet 2001:DB8:1111:7::/64. R1 also defines a

static route over the backup link to that exact same subnet, so R1 must choose whether to use the static route or the OSPF-learned route.

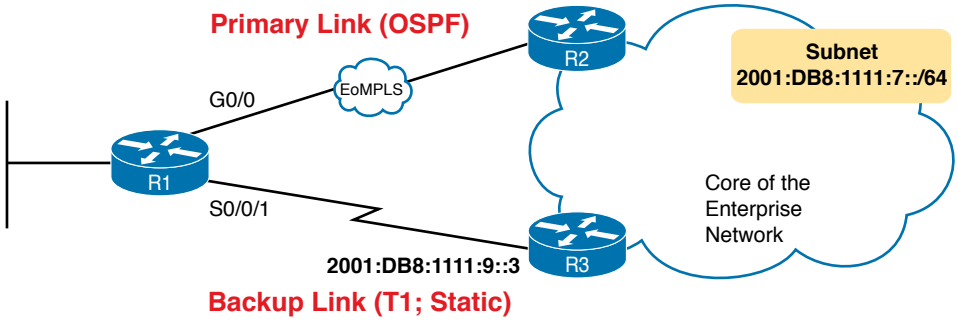


Figure 25-6 *Using a Floating Static Route to Key Subnet 2001:DB8:1111:7::/64*

IOS considers static routes better than OSPF-learned routes by default due to administrative distance. IOS uses the same administrative distance concept and default values for IPv6 as it does for IPv4. As a result, a static IPv6 route over the lower path would be given an administrative distance of 1, and an OSPFv3-learned route over the top path would be given an administrative distance of 110. R1 would use the lower path to reach subnet 2001:DB8:1111:7::/64 in this case, which is not the intended design. Instead, the engineer prefers to use the OSPF-learned routes over the much-faster primary link and use the static route over the backup link only as needed when the primary link fails.

To instead prefer the OSPF routes, the configuration would need to change the administrative distance settings and use what many networkers call a floating static route. Like an IPv4 floating static route, an IPv6 *floating static* route floats or moves into and out of the IPv6 routing table depending on whether the better (lower) administrative distance route learned by the routing protocol happens to exist currently. Basically, the router ignores the static route during times when the better routing protocol route is known.

To implement an IPv6 floating static route, just override the default administrative distance on the static route, making the value larger than the default administrative distance of the routing protocol. For example, the **ipv6 route 2001:db8:1111:7::/64 2001:db8:1111:9::3 130** command on R1 would do exactly that, setting the static route's administrative distance to 130. As long as the primary link (G0/0) stays up, and OSPFv3 on R1 learns a route for **2001:db8:1111:7**::/64 with OSPF's default administrative distance of 110, R1 ignores the static route whose administrative distance is explicitly configured as 130.

Finally, note that both the **show ipv6 route** and **show ipv6 route 2001:db8:1111:7::/64** commands list the administrative distance. Example 25-16 shows a sample matching this most recent example. Note that in this case, the static route is in use in the IPv6 routing table.

Example 25-16 *Displaying the Administrative Distance of the Static Route*

```
R1# show ipv6 route static
! Legend omitted for brevity
S   2001:db8:1111:7::/64 [130/0]
      via 2001:db8:1111:9::3

R1# show ipv6 route 2001:db8:1111:7::/64
Routing entry for 2001:db8:1111:7::/64
  Known via "static", distance 130, metric 0
  Route count is 1/1, share count 0
  Routing paths:
    2001:db8:1111:9::3
      Last updated 00:00:58 ago
```

Table 25-2 lists some of the default administrative distance values used with IPv6.

Table 25-2 IOS Defaults for Administrative Distance

Route Source	Administrative Distance
Connected routes	0
Static routes	1
NDP	2
EIGRP	90
OSPF	110
RIP	120
Unknown or unbelievable	255

Troubleshooting Static IPv6 Routes

IPv6 static routes have the same potential issues and mistakes as do static IPv4 routes, as discussed in Chapter 16. However, IPv6 static routes do have a few small differences. This last part of the static route content in the chapter looks at troubleshooting IPv6 static routes, reviewing many of the same troubleshooting rules applied to IPv4 static routes, while focusing on the details specific to IPv6.

This topic breaks static route troubleshooting into two perspectives: cases in which the route is in the routing table but is incorrect, and cases in which the route is not in the routing table.

Troubleshooting Incorrect Static Routes That Appear in the IPv6 Routing Table

A static route is only as good as the input typed into the **ipv6 route** command. IOS checks the syntax of the command, of course. However, IOS cannot tell if you choose the incorrect outgoing interface, incorrect next-hop address, or incorrect prefix/prefix-length in a static route. If the parameters pass the syntax checks, IOS places the **ipv6 route** command into the

running-config file. Then, if no other problem exists (as discussed at the next heading), IOS puts the route into the IP routing table—even though the route may not work because of the poorly chosen parameters.

For instance, an exam question might show a figure with Router R1 having an address of 2001:1:1:1::1 and neighboring Router R2 with an address of 2001:1:1:1::2. If R1 lists a static route with the command **ipv6 route 3333::/64 2001:1:1:1::1**, the command would be accepted by IOS with correct syntax, but it would not be effective as a route. Note that the command lists R1's address as the next-hop address, and R1 cannot use its own IPv6 address as a next-hop address. IOS does not prevent the configuration of the command, however; it allows the command and adds the route to the IPv6 routing table, but the route cannot possibly forward packets correctly.

When you see an exam question that has static routes, and you see them in the output of **show ipv6 route**, remember that the routes may have incorrect parameters. Check for these types of mistakes:

Key Topic

Step 1. Prefix/Length: Does the **ipv6 route** command reference the correct subnet ID (prefix) and mask (prefix length)?

Step 2. If using a next-hop IPv6 address that is a link-local address:

- **A.** Is the link-local address an address on the correct neighboring router? (It should be an address on another router on a shared link.)
- **B.** Does the **ipv6 route** command also refer to the correct outgoing interface on the local router?

Step 3. If using a next-hop IPv6 address that is a global unicast or unique local address, is the address the correct unicast address of the neighboring router?

Step 4. If referencing an outgoing interface, does the **ipv6 route** command reference the interface on the local router (that is, the same router where the static route is configured)?

This troubleshooting checklist works through the various cases in which IOS would accept the configuration of the static IPv6 route, but the route would not work because of the incorrect parameters in context. It helps to see a few examples. Figure 25-7 shows a sample network to use for the examples; all the examples focus on routes added to Router R1, for the subnet on the far right.

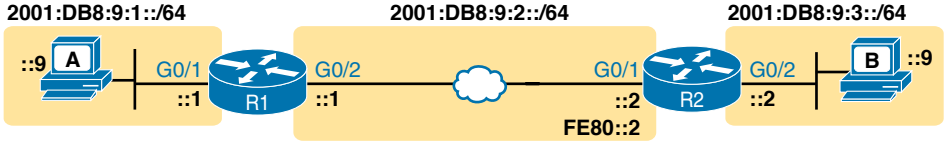


Figure 25-7 *Sample Topology for Incorrect IPv6 Route Examples*

Example 25-17 shows five **ipv6 route** commands. All have correct syntax, but all have one incorrect value; that is, the route will not work because of the types of problems in the

troubleshooting checklist. Look for the short comment at the end of each configuration command to see why each is incorrect.

Example 25-17 **ipv6 route** *Commands with Correct Syntax but Incorrect Ideas*

```
ipv6 route 2001:DB8:9:33::/64 2001:DB8:9:2::2    ! Step 1: Wrong prefix
ipv6 route 2001:DB8:9:3::/64 G0/2 FE80::AAA9     ! Step 2A: Wrong neighbor link local
ipv6 route 2001:DB8:9:3::/64 FE80::2             ! Step 2B: Missing outgoing interface
ipv6 route 2001:DB8:9:3::/64 2001:DB8:9:2::1     ! Step 3: Wrong neighbor address
ipv6 route 2001:DB8:9:3::/64 G0/1 FE80::2        ! Step 4: Wrong interface on R1
```

All these incorrect examples have correct syntax and would be added to R1's IPv6 routing table if configured on R1. However, all have flaws. Working through the examples in order:

Step 1. The prefix (2001:DB8:9:33::) has a typo in the fourth quartet (33 instead of 3).

Step 2A. The figure shows R2's G0/1 with link-local address FE80::2, but the command uses FE80::AAA9.

Step 2B. The command uses the correct link-local address on R2's address on the common link (FE80::2 per the figure), but it omits the outgoing interface of R1's G0/2 interface. (See the next example for more detail.)

Step 3. The figure shows the subnet in the center as 2001:DB8:9:2::/64, with R1 using the ::1 address and R2 using ::2. For the fourth command, R1's command should use R2's address 2001:DB8:9:2::2, but it uses R1's own 2001:DB8:9:2::1 address instead.

Step 4. As a command on R1, the outgoing interface references R1's own interfaces. R1's G0/1 is the interface on the left, whereas R1 should use its G0/2 interface on the right when forwarding packets to subnet 2001:DB8:9:3::/64.

The key takeaway for this section is to know that a route in the IPv6 routing table may be incorrect due to poor choices for the parameters. The parameters should always include the neighboring router's IPv6 addresses, but the local router's interface type/number, and in all cases, the correct prefix/length. The fact that a route is in the IPv6 routing table, particularly a static route, does not mean it is a correct route.

Note that of the five example commands in Example 25-17, IOS would accept all of them except the third one. IOS can notice the case of omitting the outgoing interface if the next-hop address is a link-local address. Example 25-18 shows a sample of the error message from IOS.

Example 25-18 *IOS Rejects the* **ipv6 route** *Command with Link-Local and No Outgoing Interface*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ipv6 route 2001:DB8:9:3::/64 FE80::2
% Interface has to be specified for a link-local nexthop
```

```
R1(config)# ^Z
R1#
R1# show running-config | include ipv6 route
R1#
```

The Static Route Does Not Appear in the IPv6 Routing Table

The preceding few pages focused on IPv6 static routes that show up in the IPv6 routing table but unfortunately have incorrect parameters. The next page looks at IPv6 routes that have correct parameters, but IOS does not place them into the IPv6 routing table.

When you add an **ipv6 route** command to the configuration, and the syntax is correct, IOS considers that route to be added to the IPv6 routing table. IOS makes the following checks before adding the route; note that IOS uses this same kind of logic for IPv4 static routes:

Key Topic

- For **ipv6 route** commands that list an outgoing interface, that interface must be in an up/up state.
- For **ipv6 route** commands that list a global unicast or unique local next-hop IP address (that is, not a link-local address), the local router must have a route to reach that next-hop address.
- If another IPv6 route exists for that exact same prefix/prefix-length, the static route must have a better (lower) administrative distance.

The Neighbor Discovery Protocol

Similar to ICMP for IPv4, IPv6 defines the ICMP protocol for IPv6 (ICMPv6). However, ICMPv6 reaches further than ICMPv4, pulling in functions done by other miscellaneous protocols in IPv4. For instance, with IPv4, ARP works as a separate protocol; with IPv6, the Neighbor Discovery Protocol (NDP), a part of ICMPv6, performs the same functions.

As it turns out, routers play a key role in several NDP protocol functions, so this final major section of the chapter explains a few of the functions of the NDP protocol (RFC 4861). Some of those NDP functions are

Key Topic

Neighbor MAC Discovery: An IPv6 LAN-based host will need to learn the MAC address of other hosts in the same subnet. NDP replaces IPv4's ARP, providing messages that replace the ARP Request and Reply messages.

Router Discovery: Hosts learn the IPv6 addresses of the available IPv6 routers in the same subnet.

SLAAC: When using Stateless Address Auto Configuration (SLAAC), the host uses NDP messages to learn the subnet (prefix) used on the link plus the prefix length.

DAD: Before using an IPv6 address, hosts use NDP to perform a Duplicate Address Detection (DAD) process, to ensure no other host uses the same IPv6 address before attempting to use it.

Discovering Neighbor Link Addresses with NDP NS and NA

NDP replaces IPv4 ARP using a pair of matched solicitation and advertisement messages: the *Neighbor Solicitation* (NS) and *Neighbor Advertisement* (NA) messages. Basically, the NS

acts like an IPv4 ARP request, asking the host with a particular unicast IPv6 address to send back a reply. The NA message acts like an IPv4 ARP Reply, listing that host's MAC address.

The process of sending the NS and NA messages follows the same general process with IPv4 ARP: the NS message asks for information, and the NA supplies the information, as summarized in this list:

Key Topic

Neighbor Solicitation (NS): This message asks the host with a particular IPv6 address (the target address) to reply with an NA message that lists its MAC address. The NS message is sent to the solicited-node multicast address associated with the target address, so the message is processed only by hosts whose last six hex digits match the address that is being queried.

Neighbor Advertisement (NA): This message lists the sender's IPv6 and MAC addresses. It can be sent in reply to an NS message, and if so, the packet is sent to the IPv6 unicast address of the host that sent the original NS message. A host can also send an unsolicited NA, announcing its IPv6 and MAC addresses, in which case the message is sent to the all-IPv6-hosts local-scope multicast address FF02::1.

NOTE With NDP, the word *neighbor* refers to the fact that the devices will be on the same data link—for example, the same VLAN.

Figure 25-8 shows an example of how a host (PC1) uses an NS message to learn the MAC address used by another host. The NS message lists a target IPv6 unicast address, with the implied question: "What is your link address?" The NA message, in this example sent back to the original host that asked the question, lists that link address.

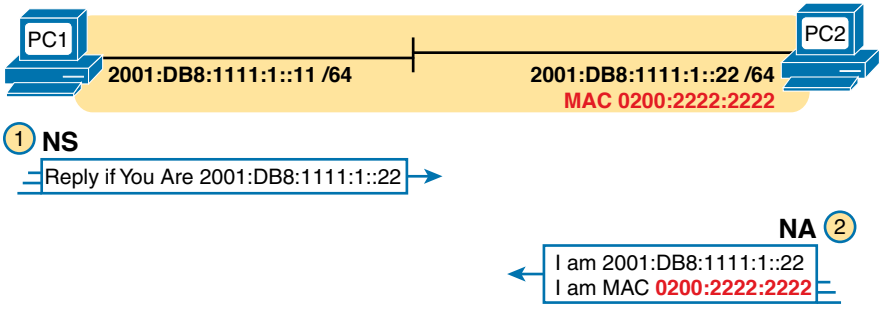


Figure 25-8 *Example NDP NS/NA Process to Find the Neighbor's Link Addresses*

At Step 1 of this particular example, PC1 sends the solicitation to find PC2's MAC address. PC1 first looks in its NDP neighbor table, the equivalent of the IPv4 ARP cache, and does not find the MAC address for IPv6 address 2001:DB8:1111:1::22. So, at Step 1, PC1 sends the NDP NS message to the matching solicited-node multicast address for 2001:DB8:1111:1::22 or FF02::1:FF00:22. Only IPv6 hosts whose address ends with 00:0022 will listen for this solicited-node multicast address. As a result, only a small subset of hosts on this link will process the received NDP NS message.

At Step 2, PC2 reacts to the received NS message. PC2 sends back an NA message in reply, listing PC2's MAC address. PC1 records PC2's MAC address in PC1's NDP neighbor table.

Example 25-19 shows an example of the IPv6 neighbor table on Router R3, as seen originally back in Figure 25-1. In this case, R3 has learned the MAC addresses of Router R1's WAN interface (G0/1/0)—both its global unicast address as well as the link-local address on that same interface.

Example 25-19 *IPv6 Neighbor Table on Router R3*

```
R3# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:1111:5::1                          0 0201.a010.0001  REACH Gi0/0/0
FE80::1:A0FF:FE10:1                         0 0201.a010.0001  REACH Gi0/0/0
```

NOTE To view a host's NDP neighbor table, use these commands: (Windows) **netsh interface ipv6 show neighbors**; (Linux) **ip -6 neighbor show**; (Mac OS) **ndp -an**.

Discovering Routers with NDP RS and RA

IPv4 hosts use the concept of an IPv4 default gateway or default router. When the host needs to send a packet to some IPv4 subnet other than the local subnet, the host sends the IPv4 packet to the default router, expecting the router to be able to route the packet to the destination. Note that hosts either statically set the IP address of their default gateway or learn it from a server called a Dynamic Host Configuration Protocol (DHCP) server.

IPv6 uses the same concept of a default gateway, but it improves the method for hosts to learn the identity of possible default gateways using NDP. NDP defines two messages that allow any host to discover all routers in the subnet:

Key Topic

Router Solicitation (RS): This message is sent to the "all-IPv6-routers" local-scope multicast address of FF02::2 so that the message asks all routers, on the local link only, to identify themselves.

Router Advertisement (RA): This message, sent by the router, lists many facts, including the link-local IPv6 address of the router. When sent in response to an RS message, it flows back to either the unicast address of the host that sent the RS or to the all-IPv6-hosts address FF02::1. Routers also send RA messages without being asked, sent to the all-IPv6-hosts local-scope multicast address of FF02::1.

For example, Figure 25-9 shows how host PC1 can learn R1's link-local address. The process is indeed simple, with PC1 first asking and R1 replying.

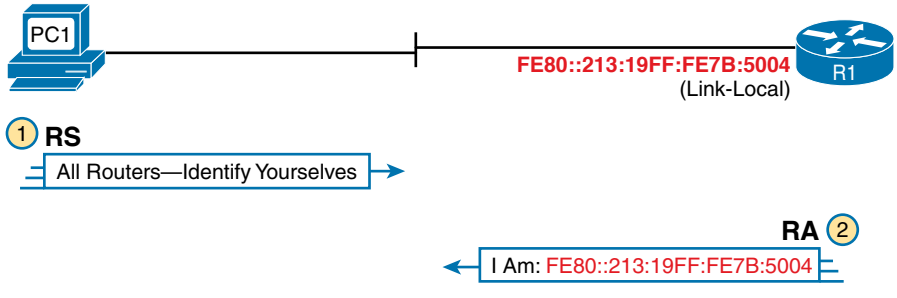


Figure 25-9 *Example NDP RS/RA Process to Find the Default Routers*

NOTE IPv6 allows multiple prefixes and multiple default routers to be listed in the RA message; Figure 25-9 just shows one of each for simplicity's sake.

IPv6 does not use broadcasts, but it does use multicasts. In this case, the RS message flows to the all-routers multicast address (FF02::2) so that all routers will receive the message. It has the same good effect as a broadcast with IPv4, without the negatives of a broadcast. In this case, only IPv6 routers will spend any CPU cycles processing the RS message, and IPv6 hosts will ignore the message. The RA message can flow either to the unicast IPv6 address of PC1 or to the all-nodes FF02::1 address.

Note that while Figure 25-9 shows how a host can ask to learn about any routers, routers also periodically send unsolicited RA messages, even without an incoming RS. When routers send these periodic RA messages, they basically advertise details about IPv6 on the link. In this case, the RA messages flow to the FF02::1 all-nodes IPv6 multicast address.

Using SLAAC with NDP RS and RA

Both IPv4 and IPv6 support the idea of dynamic address assignment for hosts via the Dynamic Host Configuration Protocol (DHCP). To find an address to use with DHCP, the DHCP client sends messages to a DHCP server, and the server assigns a currently unused address in the correct subnet for the endpoint host to use. The process relies on DHCP client functions in each device and a DHCP server configured and working in the network.

IPv6 supports an alternative method for IPv6 hosts to dynamically choose an unused IPv6 address to use—a process that does not require a server like a DHCP server. The process goes by the name *Stateless Address Autoconfiguration* (SLAAC). SLAAC uses a simple three-step process that begins by learning the prefix/length as shown in the figure. The steps are as follows:

1. Learn the IPv6 prefix used on the link, from any router, using NDP RS/RA messages.
2. Build an address from the prefix plus an interface ID, chosen either by using EUI-64 rules or as a random value.
3. Before using the address, first use DAD to make sure that no other host is already using the same address.

Figure 25-10 shows the structure of an IPv6 address created with SLACC using Steps 1 and 2 in the process, with the next topic detailing the third step (DAD).

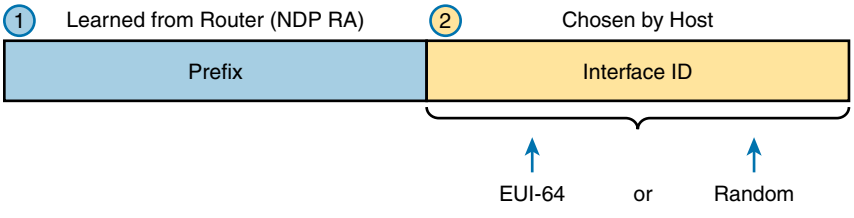


Figure 25-10 *Host IPv6 Address Formation Using SLAAC*

Discovering Duplicate Addresses Using NDP NS and NA

IPv6 uses the Duplicate Address Detection (DAD) process before using a unicast address to make sure that no other node on that link is already using the address. Hosts use DAD not only at the end of the SLAAC process, but also any time that a host interface initializes, no matter whether using SLAAC, DHCP, or static address configuration. When performing DAD, if another host already uses that address, the first host simply does not use the address until the problem is resolved.

The term *DAD* refers to the function, but the function uses NDP NS and NA messages. Basically, a host sends an NS message for its own IPv6 address. No other host should be using that address, so no other host should send an NDP NA in reply. However, if another host already uses that address, that host will reply with an NA, identifying a duplicate use of the address.

Figure 25-11 shows an example. PC1 initializes and does a DAD check, but PC2 happens to already be working and already be using the address. The figure shows the following steps:

1. PC1, before using address 2001:DB8:1111:1::11, must use DAD.
2. PC1 sends an NS message, listing the address PC1 now wants to use (2001:DB8:1111:1::11) as the target.
3. PC2 receives the NS, sees what PC2 already uses as its own address, and sends back an NA.
4. PC1, on receiving the NA message for its own IPv6 address, realizes a duplicate address exists.

Key Topic

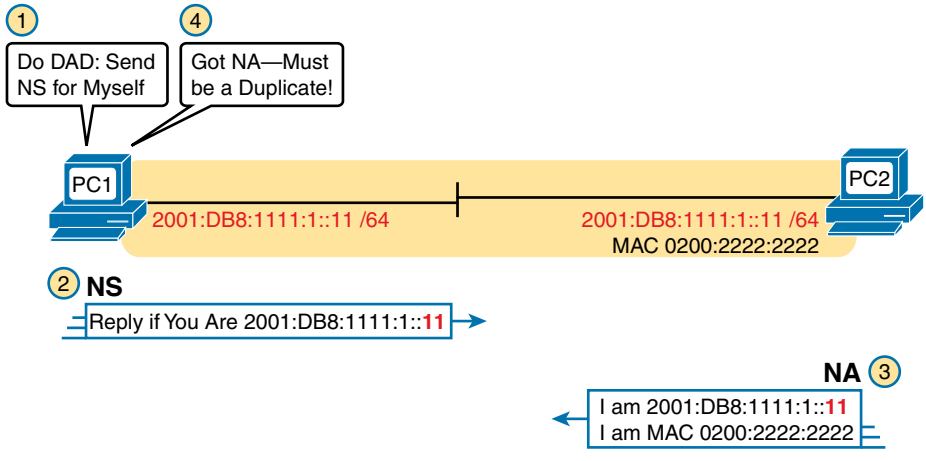


Figure 25-11 *Example Duplicate Address Detection (DAD) with NDP NS/NA*

Hosts do the DAD check for each of their unicast addresses, link-local addresses included, both when the address is first used and each time the host's interface comes up.

NDP Summary

This chapter explains some of the more important functions performed by NDP. NDP does more than what is listed in this chapter, and the protocol allows for addition of other functions, so NDP might continue to grow over time. For now, use Table 25-3 as a study reference for the four NDP features discussed here.

Key Topic

Table 25-3 NDP Function Summary

Function	Protocol Messages	Who Discovers Info	Who Supplies Info	Info Supplied
Router discovery	RS and RA	Any IPv6 host	Any IPv6 router	Link-local IPv6 address of router
Prefix/length discovery	RS and RA	Any IPv6 host	Any IPv6 router	Prefix(es) and associated prefix lengths used on local link
Neighbor discovery	NS and NA	Any IPv6 host	Any IPv6 host	Link-layer address (for example, MAC address) used by a neighbor
Duplicate Address Detection	NS and NA	Any IPv6 host	Any IPv6 host	Simple confirmation whether a unicast address is already in use

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 25-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 25-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Review memory tables		Book, website
Do labs		Blog

Review All the Key Topics

Key Topic

Table 25-5 Key Topics for Chapter 25

Key Topic Element	Description	Page Number
List	Methods by which a router can build IPv6 routes	583
List	Rules for IPv6 connected and local routes	583
Figure 25-2	IPv6 static route concepts	587
Checklist	Items to check on **ipv6 route** command that cause problems with IPv6 static routes	596
Checklist	Items to check other than the **ipv6 route** command that cause problems with IPv6 static routes	598
List	Four functions that use NDP messages	598
List	NDP NS and NA messages and meanings	599
List	NDP RS and RA messages and meanings	600
Figure 25-11	Example DAD check	602
Table 25-3	NDP Function summary table	603

Key Terms You Should Know

IPv6 host route, local route, IPv6 local route, IPv6 administrative distance, IPv6 multicast scope

Command References

Tables 25-6 and 25-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 25-6 Chapter 25 Configuration Command Reference

Command	Description
ipv6 route *prefix/length next-hop-address*	Global command to define an IPv6 static route to a next-hop router IPv6 address.
ipv6 route *prefix/length outgoing-interface*	Global command to define an IPv6 static route, with packets forwarded out the local router interface listed in the command.
ipv6 route *prefix/length outgoing-interface next-hop-address*	Global command to define an IPv6 static route, with both the next-hop address and local router outgoing interface listed.

Command	Description
ipv6 route ::/0 *{[next-hop-address] [outgoing-interface]}*	Global command to define a default IPv6 static route.
ipv6 address autoconfig [default]	Interface subcommand that tells the router to use SLAAC to find/build its own interface IPv6 address, and with the **default** parameter, to add a default route with a next hop of the router that responds with the RA message.

Table 25-7 Chapter 25 EXEC Command Reference

Command	Description
show ipv6 route [**connected** | **local** | **static**]	Lists routes in the routing table.
show ipv6 route *address*	Displays detailed information about the route this router uses to forward packets to the IPv6 address listed in the command.
show ipv6 neighbors	Lists the contents of the IPv6 neighbor table, which lists the MAC address associated with IPv6 addresses on common subnets.

Part VII Review

Keep track of your part review progress with the checklist in Table P7-1. Details on each task follow the table.

Table P7-1 Part VII Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Watch Videos		

Repeat All DIKTA Questions

For this task, use the PCPT software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or using the Key Topics application on the companion website.

Do Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog: Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book, as found at http://blog.certskills.com. Then navigate to the Hands-on Config labs.

Other: If using other lab tools, here are a few suggestions: Configure IPv6 addresses on interfaces, and before using any show commands, predict the connected and local routes that should be added to the IPv6 routing table, and predict the link-local (unicast) address and various multicast addresses you expect to see in the output of the **show ipv6 interfaces** command.

Watch Videos

Chapter 24 mentions that the companion website's section for Chapter 24 review includes a video about the EUI-64 address generation process, so consider using the video as a review.

This book began with an overview of the fundamentals of LANs, WANs, and IP routing. It then described Ethernet LANs (wired LANs) in some depth over the course of seven chapters. The book then meandered through many chapters exploring the many concepts of IPv4 and IPv6 addressing, routing, and how to implement those features in Cisco devices.

This final part of Volume 1 turns our attention back to the LAN, not to wired Ethernet LANs, but to IEEE 802.11 wireless LANs—in other words, Wi-Fi. The four chapters in this part of the book lay down the foundations of how wireless LANs work and then show how to implement wireless LANs using Cisco devices.

Building wireless LANs requires some thought because the endpoints that use the LAN do not sit in one place and connect via a known cable and known switch port. To explain those details, Chapter 26 begins with the basics of how a wireless client can connect to the wireless network through a wireless access point (AP). After you learn the foundations in Chapter 26, Chapter 27 takes an architectural view of wireless LANs to discuss how you might build a wireless LAN for an enterprise, which requires much different thinking than, for instance, building a wireless LAN for your home.

Chapter 28 completes the three concepts-focused wireless LAN chapters by working through the alphabet soup that is wireless LAN security. The fact that wireless LAN clients come and go means that the LAN may be under constant attack as an easy place for an attacker to gain access to the network, so wireless LANs must use effective security. Finally, Chapter 29 closes by showing how to configure an enterprise wireless LAN using Cisco APs and the Cisco Wireless LAN Controller (WLC) from the WLC's graphical interface.

Part VIII

Wireless LANs

CHAPTER 26

Fundamentals of Wireless Networks

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.d Access Points

1.11 Describe wireless principles

1.11.a Nonoverlapping Wi-Fi channels

1.11.b SSID

1.11.c RF

Wireless communication usually involves a data exchange between two devices. A wireless LAN goes even further; many devices can participate in sharing the medium for data exchanges. Wireless LANs must transmit a signal over radio frequencies (RF) to move data from one device to another. Transmitters and receivers can be fixed in consistent locations, or they can be mobile and free to move around. This chapter explains the topologies that can be used to control access to the wireless medium and provide data exchange between devices.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 26-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Comparing Wired and Wireless Networks	1
Wireless LAN Topologies	2–4
Other Wireless Topologies	5–6
Wireless Bands and Channels	7–8

1. Wired Ethernet and Wi-Fi are based on which two IEEE standards, respectively?
   a. 802.1, 802.3
   b. 802.3, 802.1
   c. 802.3, 802.11
   d. 802.11, 802.3
2. Devices using a wireless LAN must operate in which one of the following modes?
   a. Round-robin access
   b. Half duplex
   c. Full duplex
   d. None of these answers
3. An access point is set up to offer wireless coverage in an office. Which one of the following is the correct 802.11 term for the resulting standalone network?
   a. BSA
   b. BSD
   c. BSS
   d. IBSS
4. Which one of the following is used to uniquely identify an AP and the basic service set it maintains with its associated wireless clients?
   a. SSID
   b. BSSID
   c. Ethernet MAC address
   d. Radio MAC address
5. Which one of the following can be used to provide wireless connectivity to a nonwireless device?
   a. Wireless repeater
   b. Workgroup bridge
   c. Transparent bridge
   d. Adaptive bridge
6. Which one of the following is not needed in a Cisco outdoor mesh network?
   a. A BSS function
   b. Ethernet cabling to each AP
   c. A workgroup bridge
   d. A backhaul network

7. Which of the following are frequency bands commonly used for Wi-Fi?
   a. 2.4 KHz
   b. 2.4 MHz
   c. 5 MHz
   d. 2.4 GHz
   e. 5 GHz
8. Which of the following are considered to be nonoverlapping channels?
   a. Channels 1, 2, and 3 in the 2.4-GHz band
   b. Channels 1, 5, and 10 in the 2.4-GHz band
   c. Channels 1, 6, and 11 in the 2.4-GHz band
   d. Channels 40, 44, and 48 in the 5-GHz band

Foundation Topics

Comparing Wired and Wireless Networks

In a wired network, any two devices that need to communicate with each other must be connected by a wire. (That was obvious!) The "wire" might contain strands of metal or fiber-optic material that run continuously from one end to the other. Data that passes over the wire is bounded by the physical properties of the wire. In fact, the IEEE 802.3 set of standards defines strict guidelines for the Ethernet wire itself, in addition to how devices may connect, send, and receive data over the wire.

Wired connections have been engineered with tight constraints and have few variables that might prevent successful communication. Even the type and size of the wire strands, the number of twists the strands must make around each other over a distance, and the maximum length of the wire must adhere to the standard.

Therefore, a wired network is essentially a bounded medium; data must travel over whatever path the wire or cable takes between two devices. If the cable goes around a corner or lies in a coil, the electrical signals used to carry the data must also go around a corner or around a coil. Because only two devices may connect to a wire, only those two devices may send or transmit data. Even better: the two devices may transmit data to each other simultaneously because they each have a private, direct path to each other.

Wired networks also have some shortcomings. When a device is connected by a wire, it cannot move around very easily or very far. Before a device can connect to a wired network, it must have a connector that is compatible with the one on the end of the wire. As devices get smaller and more mobile, it just is not practical to connect them to a wire.

As its name implies, a wireless network removes the need to be tethered to a wire or cable. Convenience and mobility become paramount, enabling users to move around at will while staying connected to the network. A user can (and often does) bring along many different wireless devices that can all connect to the network easily and seamlessly.

Wireless data must travel through free space, without the constraints and protection of a wire. In the free space environment, many variables can affect the data and its delivery. To minimize the variables, wireless engineering efforts must focus on two things:

- Wireless devices must adhere to a common standard (IEEE 802.11).
- Wireless coverage must exist in the area where devices are expected to use it.

As you study for the CCNA 200-301 exam, keep in mind that the exam is geared more toward a functional view of wireless technology. More detailed topics like RF characteristics, antenna performance, and so on are reserved for the Implementing Cisco Enterprise Network Core Technologies ENCOR 300-401 exam.

Wireless LAN Topologies

Wireless communication takes place over free space through the use of radio frequency (RF) signals. The theory behind RF signals can be complex, and is described further in the "RF Overview" section in this chapter. For now, just assume that one device, the transmitter, sends RF signals to another device, the receiver. As Figure 26-1 shows, the transmitter can contact the receiver at any and all times, as long as both devices are tuned to the same frequency (or channel) and use the same scheme to carry the data between them. That all sounds simple, except that it is not really practical.

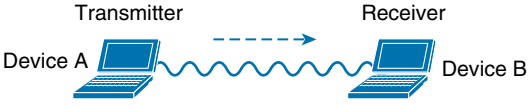


Figure 26-1 *Unidirectional Communication*

To fully leverage wireless communication, data should travel in *both* directions, as shown in Figure 26-2. Sometimes Device A needs to send data to Device B, while Device B would like to take a turn to send at other times.

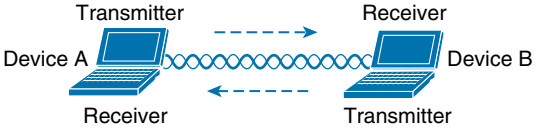


Figure 26-2 *Bidirectional Communication*

Because the two devices are using the same channel, two phrases in the preceding sentence become vitally important: *take a turn* and *send at other times*. With wireless communication, if multiple signals are received at the same time, they can interfere with each other. The likelihood of interference increases as the number of wireless devices grows. For example, Figure 26-3 shows four devices tuned to the same channel and what might happen if some or all of them transmit at the same time.

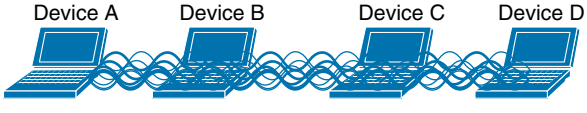


Figure 26-3 *Interference from Simultaneous Transmissions*

All this talk about waiting turns and avoiding interference might remind you of a traditional (nonswitched) Ethernet LAN, where multiple hosts can connect to a shared media and share a common bandwidth. To use the media effectively, all the hosts must operate in half-duplex mode so that they try to avoid colliding with other transmissions already in progress. The side effect is that no host can transmit and receive at the same time on a shared medium.

A wireless LAN is similar. Because multiple hosts can share the same channel, they also share the "airtime" or access to that channel at any given time. Therefore, to keep everything clean, only one device should transmit at any given time. To contend for use of the channel, devices based on the 802.11 standard have to determine whether the channel is clear and available before transmitting anything.

NOTE IEEE 802.11 WLANs are always half duplex because transmissions between stations use the same frequency or channel. Only one station can transmit at any time; otherwise, collisions occur. To achieve full-duplex mode, one station's transmission would have to occur on one frequency while it receives over a different frequency—much like full-duplex Ethernet links work. Although this is certainly possible and practical, the 802.11 standard does not permit full-duplex operation. Some amendments to the standard do provide a means for multiple devices to transmit on the same channel at the same time, but this is beyond the scope of this book.

At the most basic level, there is no inherent organization to a wireless medium or any inherent control over the number of devices that can transmit and receive frames. Any device that has a wireless network adapter can power up at any time and try to communicate. At a minimum, a wireless network should have a way to make sure that every device using a channel can support a common set of parameters. Beyond that, there should be a way to control which devices (and users) are allowed to use the wireless medium and the methods that are used to secure the wireless transmissions.

Basic Service Set

The solution is to make every wireless service area a closed group of mobile devices that forms around a fixed device; before a device can participate, it must advertise its capabilities and then be granted permission to join. The 802.11 standard calls this a *basic service set* (BSS). At the heart of every BSS is a wireless *access point* (AP), as shown in Figure 26-4. The AP operates in *infrastructure mode*, which means it offers the services that are necessary to form the infrastructure of a wireless network. The AP also establishes its BSS over a single wireless channel. The AP and the members of the BSS must all use the same channel to communicate properly.

Because the operation of a BSS hinges on the AP, the BSS is bounded by the area where the AP's signal is usable. This is known as the *basic service area* (BSA) or *cell*. In Figure 26-4, the cell is shown as a simple shaded circular area that centers around the AP itself. Cells can

---

Answers to the "Do I Know This Already?" quiz:

1 C **2** B **3** C **4** B **5** B **6** B **7** D, E **8** C, D

have other shapes too, depending on the antenna that is connected to the AP and on the physical surroundings that might affect the AP's signals.

The AP serves as a single point of contact for every device that wants to use the BSS. It advertises the existence of the BSS so that devices can find it and try to join. To do that, the AP uses a unique BSS identifier (BSSID) that is based on the AP's own radio MAC address.

NOTE Recall that wired Ethernet devices each have a unique MAC address to send frames from a source to a destination over a Layer 2 network. Wireless devices must also have unique MAC addresses to send wireless frames at Layer 2 over the air.

Key Topic

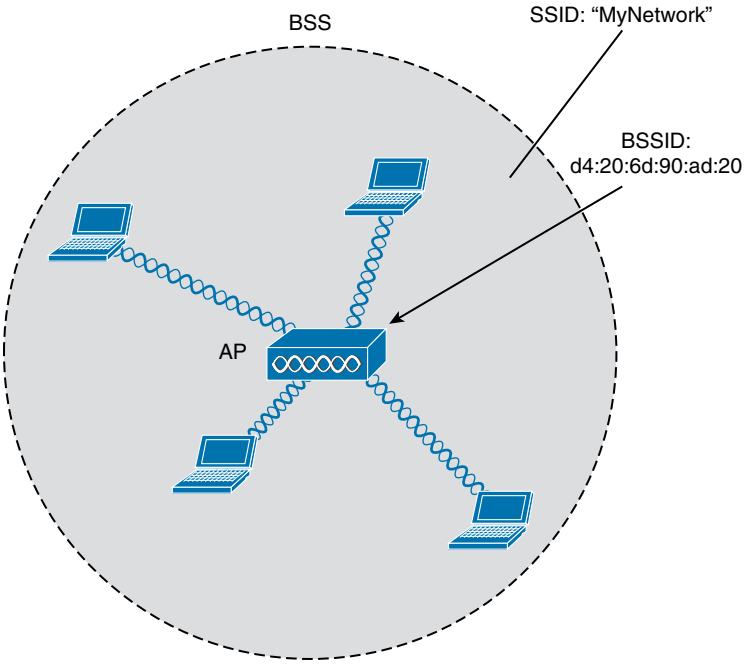


Figure 26-4 *802.11 Basic Service Set*

In addition, the AP advertises the wireless network with a Service Set Identifier (SSID), which is a text string containing a logical name. Think of the BSSID as a machine-readable name tag that uniquely identifies the BSS ambassador (the AP), and the SSID as a nonunique, human-readable name tag that identifies the wireless service.

Membership with the BSS is called an *association*. A wireless device must send an association request to the AP and the AP must either grant or deny the request. Once associated, a device becomes a client, or an 802.11 *station* (STA), of the BSS. What then? As long as a wireless client remains associated with a BSS, most communications to and from the client must pass *through* the AP, as indicated in Figure 26-5. By using the BSSID as a source or destination address, data frames can be relayed to or from the AP.

You might be wondering why all client traffic has to traverse the AP at all. Why can two clients not simply transmit data frames directly to each other and bypass the middleman? If clients

are allowed to communicate directly, then the whole idea of organizing and managing a BSS is moot. By sending data through the AP first, the BSS remains stable and under control.

NOTE Even though data frames are meant to pass through an AP, keep in mind that other devices in the same general area that are listening on the same channel can overhear the transmissions. After all, wireless frames are not contained within a wire that connects a device to an AP. Instead, the frames are freely available over the air to anyone that is within range to receive them. If the frames are unencrypted, then anyone may inspect their contents. Only the BSSID value contained within the frames indicates that the intended sender or recipient is the AP.

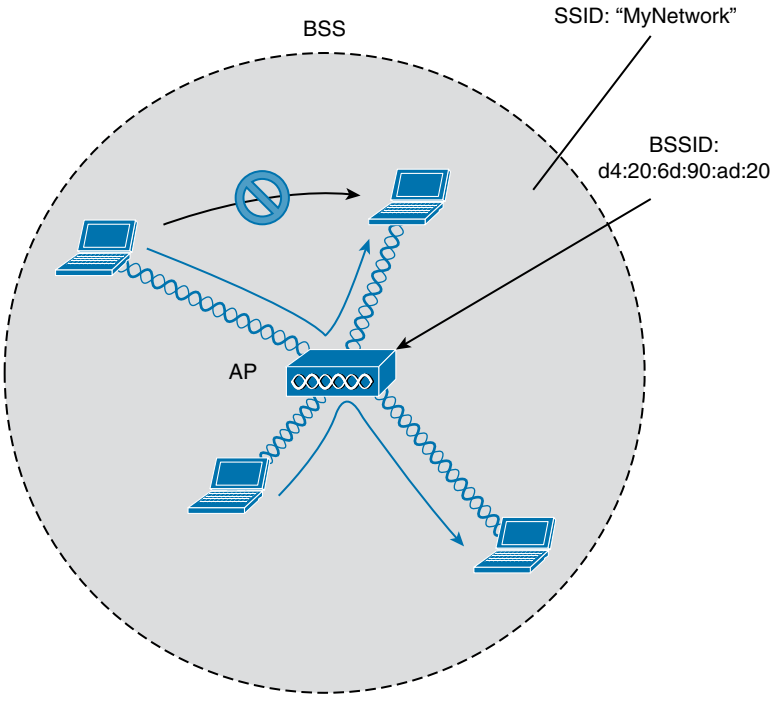


Figure 26-5 *Traffic Flows Within a BSS*

Distribution System

Notice that a BSS involves a single AP and no explicit connection into a regular Ethernet network. In that setting, the AP and its associated clients make up a standalone network. But the AP's role at the center of the BSS does not just stop with managing the BSS; sooner or later, wireless clients will need to communicate with other devices that are not members of the BSS. Fortunately, an AP can also uplink into an Ethernet network because it has both wireless and wired capabilities. The 802.11 standard refers to the upstream wired Ethernet as the *distribution system* (DS) for the wireless BSS, as shown in Figure 26-6.

You can think of an AP as a translational bridge, where frames from two dissimilar media (wireless and wired) are translated and then bridged at Layer 2. In simple terms, the AP is in charge of mapping a virtual local-area network (VLAN) to an SSID. In Figure 26-6, the AP

maps VLAN 10 to the wireless LAN using SSID "MyNetwork." Clients associated with the "MyNetwork" SSID will appear to be connected to VLAN 10.

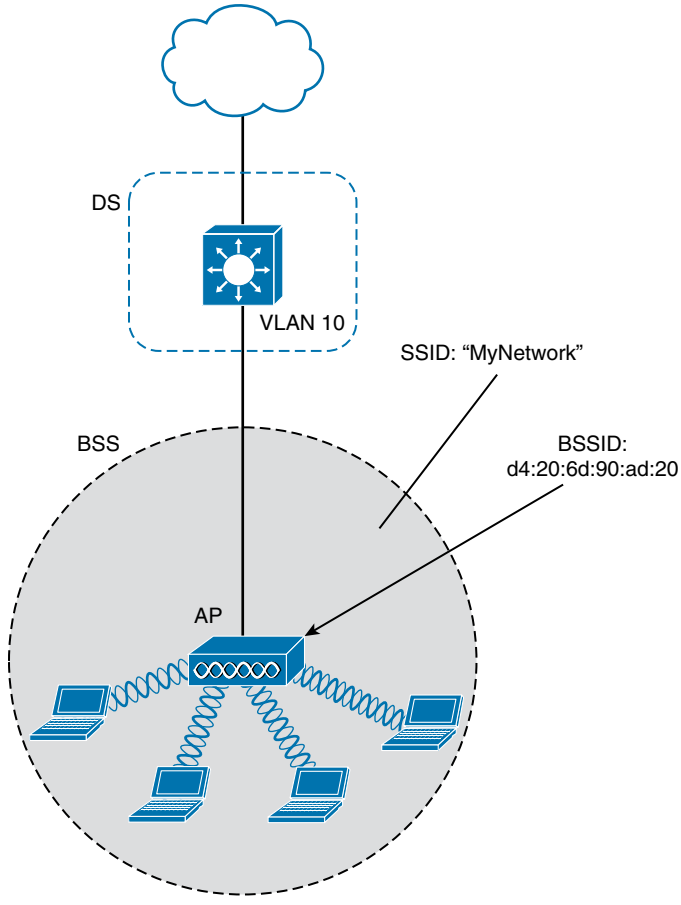


Figure 26-6 *Distribution System Supporting a BSS*

This concept can be extended so that multiple VLANs are mapped to multiple SSIDs. To do this, the AP must be connected to the switch by a trunk link that carries the VLANs. In Figure 26-7, VLANs 10, 20, and 30 are trunked to the AP over the DS. The AP uses the 802.1Q tag to map the VLAN numbers to the appropriate SSIDs. For example, VLAN 10 is mapped to SSID "MyNetwork," VLAN 20 is mapped to SSID "YourNetwork," and VLAN 30 to SSID "Guest."

In effect, when an AP uses multiple SSIDs, it is trunking VLANs over the air, and over the same channel, to wireless clients. The clients must use the appropriate SSID that has been mapped to the respective VLAN when the AP was configured. The AP then appears as multiple logical APs—one per BSS—with a unique BSSID for each. With Cisco APs, this is usually accomplished by incrementing the last digit of the radio's MAC address for each SSID.

Even though an AP can advertise and support multiple logical wireless networks, each of the SSIDs covers the same geographic area. The reason is that the AP uses the same transmitter, receiver, antennas, and channel for every SSID that it supports. Beware of one misconception though: multiple SSIDs can give an illusion of scale. Even though wireless clients can be

distributed across many SSIDs, all of those clients must share the same AP's hardware and must contend for airtime on the same channel.

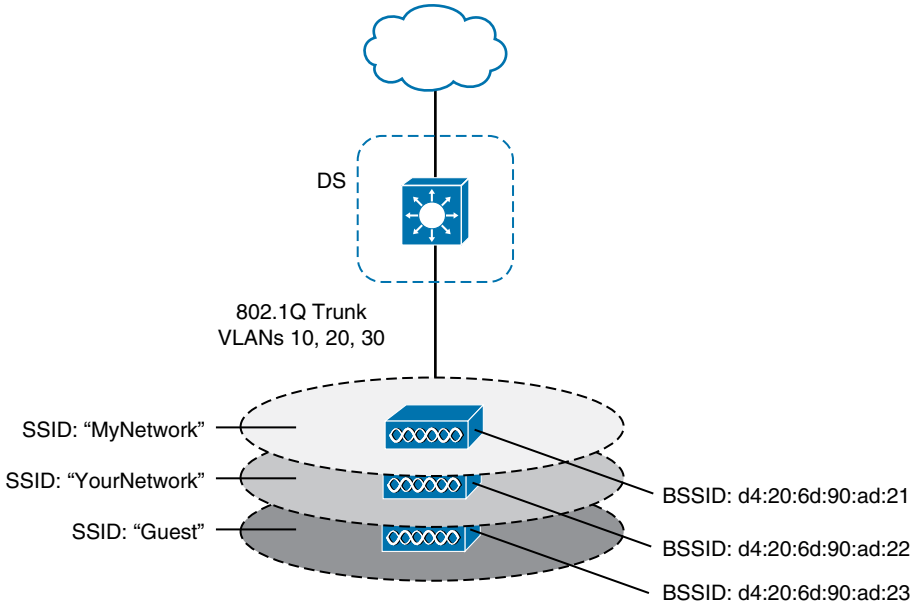


Figure 26-7 *Supporting Multiple SSIDs on One AP*

Extended Service Set

Normally, one AP cannot cover the entire area where clients might be located. For example, you might need wireless coverage throughout an entire floor of a business, hotel, hospital, or other large building. To cover more area than a single AP's cell can cover, you simply need to add more APs and spread them out geographically.

When APs are placed at different geographic locations, they can all be interconnected by a switched infrastructure. The 802.11 standard calls this an extended service set (ESS), as shown in Figure 26-8.

The idea is to make multiple APs cooperate so that the wireless service is consistent and seamless from the client's perspective. Ideally, any SSIDs that are defined on one AP should be defined on all the APs in an ESS; otherwise, it would be very cumbersome and inconvenient for a client to be reconfigured each time it moves into a different AP's cell.

Notice that each cell in Figure 26-8 has a unique BSSID, but both cells share one common SSID. Regardless of a client's location within the ESS, the SSID will remain the same but the client can always distinguish one AP from another.

In an ESS, a wireless client can associate with one AP while it is physically located near that AP. If the client later moves to a different location, it can associate with a different nearby AP automatically. Passing from one AP to another is called *roaming*. Keep in mind that each AP offers its own BSS on its own channel, to prevent interference between the APs. As a client device roams from one AP to another, it must scan the available channels to find a new AP (and BSS) to roam toward. In effect, the client is roaming from BSS to BSS, and from channel to channel.

Key Topic

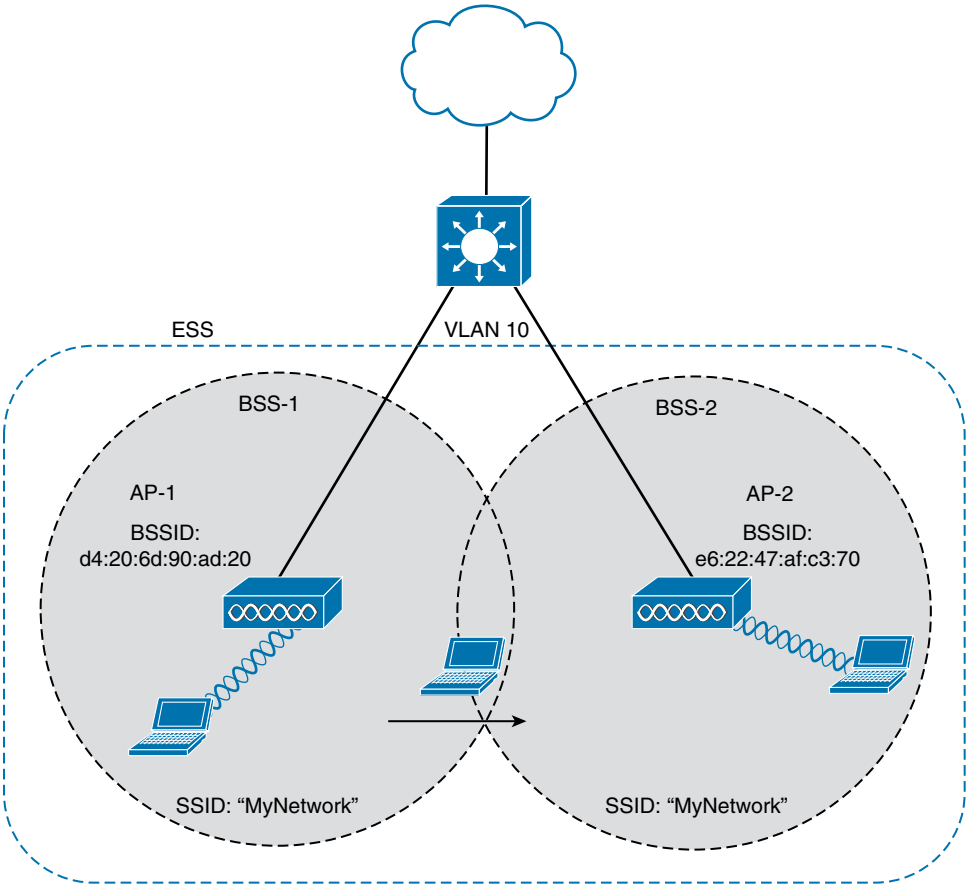


Figure 26-8 *Scaling Wireless Coverage with an 802.11 Extended Service Set*

Independent Basic Service Set

Usually a wireless network leverages APs for organization, control, and scalability. Sometimes that is not possible or convenient in an impromptu situation. For example, two people who want to exchange electronic documents at a meeting might not be able to find a BSS available or might want to avoid having to authenticate to a production network. In addition, many personal printers have the capability to print documents wirelessly, without relying on a regular BSS or AP.

The 802.11 standard allows two or more wireless clients to communicate directly with each other, with no other means of network connectivity. This is known as an *ad hoc* wireless network, or an *independent basic service set* (IBSS), as shown in Figure 26-9. For this to work, one of the devices must take the lead and begin advertising a network name and the necessary radio parameters, much like an AP would do. Any other device can then join as needed. IBSSs are meant to be organized in an impromptu, distributed fashion; therefore, they do not scale well beyond eight to ten devices.

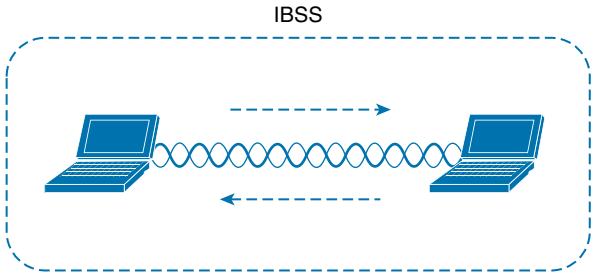


Figure 26-9 *802.11 Independent Basic Service Set*

Other Wireless Topologies

Wireless APs can be configured to operate in noninfrastructure modes when a normal BSS cannot provide the functionality that is needed. The following sections cover the most common modes.

Repeater

Normally, each AP in a wireless network has a wired connection back to the DS or switched infrastructure. To extend wireless coverage beyond a normal AP's cell footprint, additional APs and their wired connections can be added. In some scenarios, it is not possible to run a wired connection to a new AP because the cable distance is too great to support Ethernet communication.

In that case, you can add an additional AP that is configured for *repeater mode*. A wireless repeater takes the signal it receives and repeats or retransmits it in a new cell area around the repeater. The idea is to move the repeater out away from the AP so that it is still within range of both the AP and the distant client, as shown in Figure 26-10.

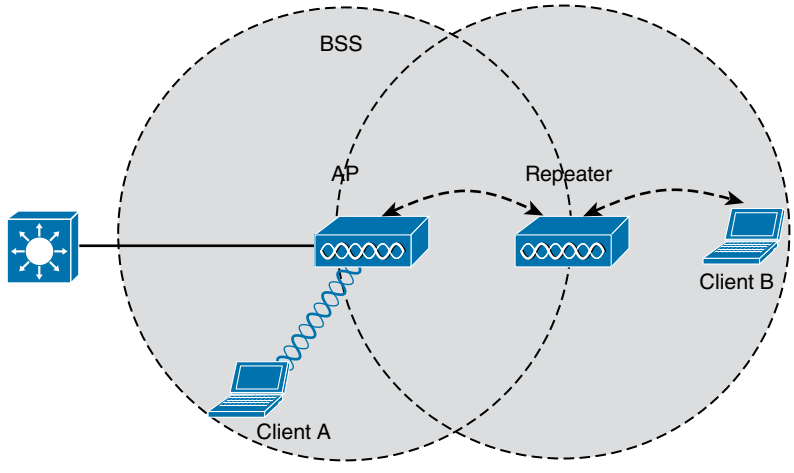


Figure 26-10 *Extending the Range of an AP with a Wireless Repeater*

If the repeater has a single transmitter and receiver, it must operate on the same channel that the AP is using. That can create the possibility that the AP's signal will be received and retransmitted by the repeater, only to be received again by the AP—halving the effective

throughput because the channel will be kept busy twice as long as before. As a remedy, some repeaters can use two transmitters and receivers to keep the original and repeated signals isolated on different channels. One transmitter and receiver pair is dedicated to signals in the AP's cell, while the other pair is dedicated to signals in the repeater's own cell.

Workgroup Bridge

Suppose you have a device that supports a wired Ethernet link but is not capable of having a wireless connection. For example, some mobile medical devices might be designed with only a wired connection. While it is possible to plug the device into an Ethernet connection when needed, a wireless connection would be much more practical. You can use a workgroup bridge (WGB) to connect the device's wired network adapter to a wireless network.

Rather than providing a BSS for wireless service, a WGB becomes a wireless client of a BSS. In effect, the WGB acts as an external wireless network adapter for a device that has none. In Figure 26-11, an AP provides a BSS; Client A is a regular wireless client, while Client B is associated with the AP through a WGB.

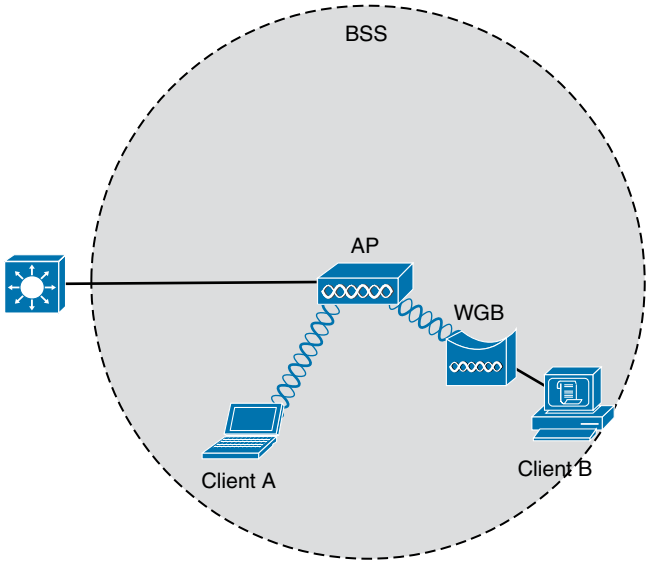


Figure 26-11 *Nonwireless Device Connecting Through a Workgroup Bridge*

You might encounter two types of workgroup bridges:

- **Universal workgroup bridge (uWGB):** A single wired device can be bridged to a wireless network.
- **Workgroup bridge (WGB):** A Cisco-proprietary implementation that allows multiple wired devices to be bridged to a wireless network.

Outdoor Bridge

An AP can be configured to act as a bridge to form a single wireless link from one LAN to another over a long distance. Outdoor bridged links are commonly used for connectivity between buildings or between cities.

If the LANs at two locations need to be bridged, a point-to-point bridged link can be used. One AP configured in bridge mode is needed on each end of the wireless link. Special purpose antennas are normally used with the bridges to focus their signals in one direction—toward the antenna of the AP at the far end of the link. This maximizes the link distance, as shown in Figure 26-12.

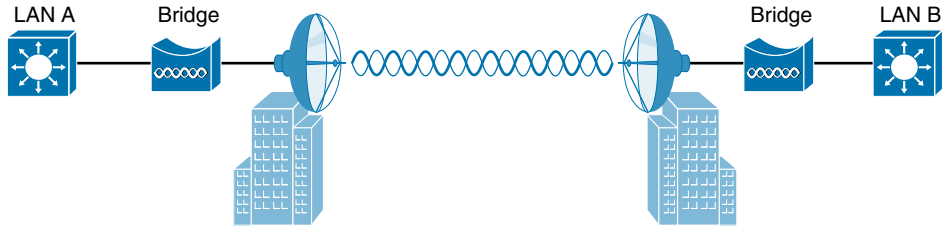


Figure 26-12 *Point-to-Point Outdoor Bridge*

Sometimes the LANs at multiple sites need to be bridged together. A point-to-multipoint bridged link allows a central site to be bridged to several other sites. The central site bridge is connected to an omnidirectional antenna, such that its signal is transmitted equally in all directions so that it can reach the other sites simultaneously. The bridges at each of the other sites can be connected to a directional antenna aimed at the central site. Figure 26-13 shows the point-to-multipoint scenario.

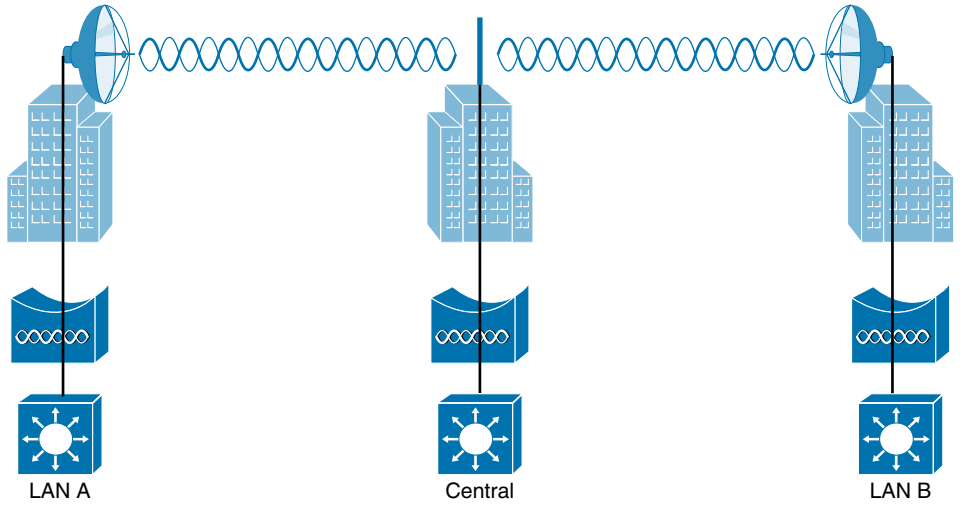


Figure 26-13 *Point-to-Multipoint Outdoor Bridge*

Mesh Network

To provide wireless coverage over a very large area, it is not always practical to run Ethernet cabling to every AP that would be needed. Instead, you could use multiple APs configured in mesh mode. In a mesh topology, wireless traffic is bridged from AP to AP, in a daisy-chain fashion, using another wireless channel.

Mesh APs can leverage dual radios—one using a channel in one range of frequencies and one a different range. Each mesh AP usually maintains a BSS on one channel, with which

wireless clients can associate. Client traffic is then usually bridged from AP to AP over other channels as a backhaul network. At the edge of the mesh network, the backhaul traffic is bridged to the wired LAN infrastructure. Figure 26-14 shows a typical mesh network. With Cisco APs, you can build a mesh network indoors or outdoors. The mesh network runs its own dynamic routing protocol to work out the best path for backhaul traffic to take across the mesh APs.

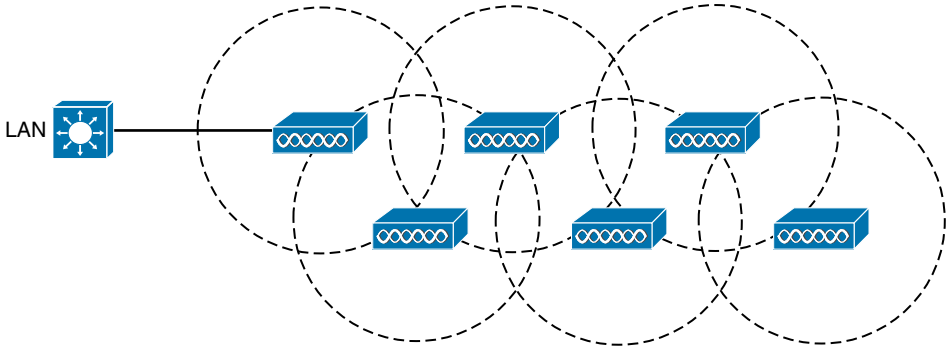


Figure 26-14 *Typical Wireless Mesh Network*

RF Overview

To send data across a wired link, an electrical signal is applied at one end and carried to the other end. The wire itself is continuous and conductive, so the signal can propagate rather easily. A wireless link has no physical strands of anything to carry the signal along.

How, then, can an electrical signal be sent across the air, or free space? Consider a simple analogy of two people standing far apart. One person wants to signal something to the other. They are connected by a long and somewhat loose rope; the rope represents free space. The sender at one end decides to lift his end of the rope high and hold it there so that the other end of the rope will also rise and notify the partner. After all, if the rope were a wire, he knows that he could apply a steady voltage at one end of the wire and it would appear at the other end. Figure 26-15 shows the end result; the rope falls back down after a tiny distance, and the receiver never notices a change.

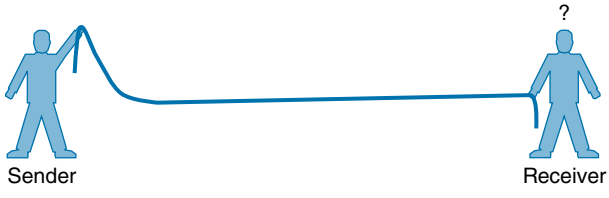


Figure 26-15 *Failed Attempt to Pass a Message Down a Rope*

The sender tries a different strategy. He cannot push the rope, but when he begins to wave it up and down in a steady, regular motion, a curious thing happens. A continuous wave pattern appears along the entire length of the rope, as shown in Figure 26-16. In fact, the waves (each representing one up and down cycle of the sender's arm) actually travel from the sender to the receiver.

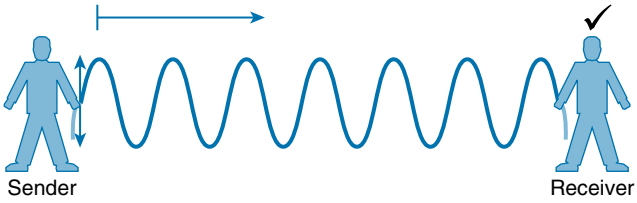


Figure 26-16 *Sending a Continuous Wave Down a Rope*

In free space, a similar principle occurs. The sender (a transmitter) can send an alternating current into a section of wire (an antenna), which sets up moving electric and magnetic fields that propagate out and away as traveling waves. The electric and magnetic fields travel along together and are always at right angles to each other, as shown in Figure 26-17. The signal must keep changing, or alternating, by cycling up and down, to keep the electric and magnetic fields cycling and pushing ever outward.

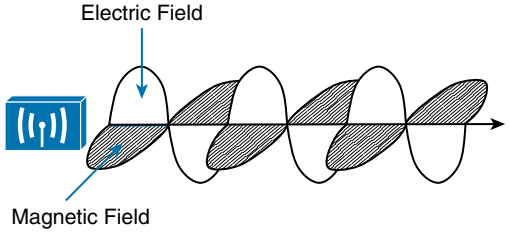


Figure 26-17 *Traveling Electric and Magnetic Waves*

Electromagnetic waves do not travel in a straight line. Instead, they travel by expanding in *all* directions away from the antenna. To get a visual image, think of dropping a pebble into a pond when the surface is still. Where it drops in, the pebble sets the water's surface into a cyclic motion. The waves that result begin small and expand outward, only to be replaced by new waves. In free space, the electromagnetic waves expand outward in all three dimensions.

Figure 26-18 shows a simple idealistic antenna that is a single point at the end of a wire. The waves produced expand outward in a spherical shape. The waves will eventually reach the receiver, in addition to many other locations in other directions.

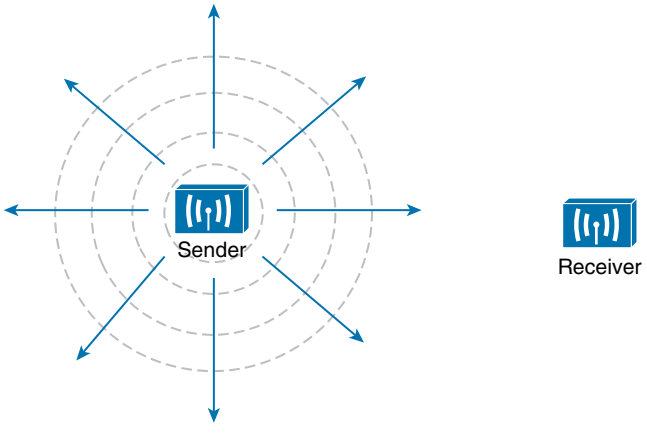


Figure 26-18 *Wave Propagation with an Idealistic Antenna*

At the receiving end of a wireless link, the process is reversed. As the electromagnetic waves reach the receiver's antenna, they induce an electrical signal. If everything works right, the received signal will be a reasonable copy of the original transmitted signal.

The electromagnetic waves involved in a wireless link can be measured and described in several ways. One fundamental property is the *frequency* of the wave, or the number of times the signal makes one complete up and down *cycle* in 1 second. Figure 26-19 shows how a cycle of a wave can be identified. A cycle can begin as the signal rises from the center line, falls through the center line, and rises again to meet the center line. A cycle can also be measured from the center of one peak to the center of the next peak. No matter where you start measuring a cycle, the signal must make a complete sequence back to its starting position where it is ready to repeat the same cyclic pattern.

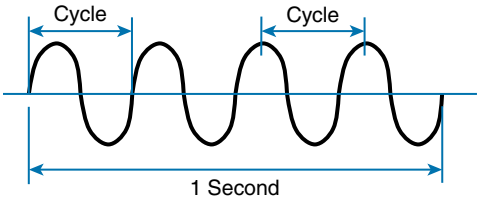


Figure 26-19 *Cycles Within a Wave*

In Figure 26-19, suppose that 1 second has elapsed, as shown. During that 1 second, the signal progressed through four complete cycles. Therefore, its frequency is 4 cycles/second or 4 hertz. A *hertz* (Hz) is the most commonly used frequency unit and is nothing other than one cycle per second.

Frequency can vary over a very wide range. As frequency increases by orders of magnitude, the numbers can become quite large. To keep things simple, the frequency unit name can be modified to denote an increasing number of zeros, as listed in Table 26-2.

Table 26-2 Frequency Unit Names

Unit	Abbreviation	Meaning
Hertz	Hz	Cycles per second
Kilohertz	kHz	1000 Hz
Megahertz	MHz	1,000,000 Hz
Gigahertz	GHz	1,000,000,000 Hz

Figure 26-20 shows a simple representation of the continuous frequency spectrum ranging from 0 Hz to 10^{22} (or 1 followed by 22 zeros) Hz. At the low end of the spectrum are frequencies that are too low to be heard by the human ear, followed by audible sounds. The highest range of frequencies contains light, followed by X, gamma, and cosmic rays.

Frequency (Hz)	Frequency Notation
10^{21}	
10^{20}	
10^{19}	
10^{18}	
10^{17}	
10^{16}	
10^{15}	
10^{14}	
10^{13}	
10^{12}	
10^{11}	100 GHz
10^{10}	10 GHz
10^{9}	1 GHz
10^{8}	100 MHz
10^{7}	10 MHz
10^{6}	1 MHz
10^{5}	100 kHz
10^{4}	10 kHz
10^{3}	1 kHz
10^{2}	100 Hz
10^{1}	10 Hz
0	0 Hz

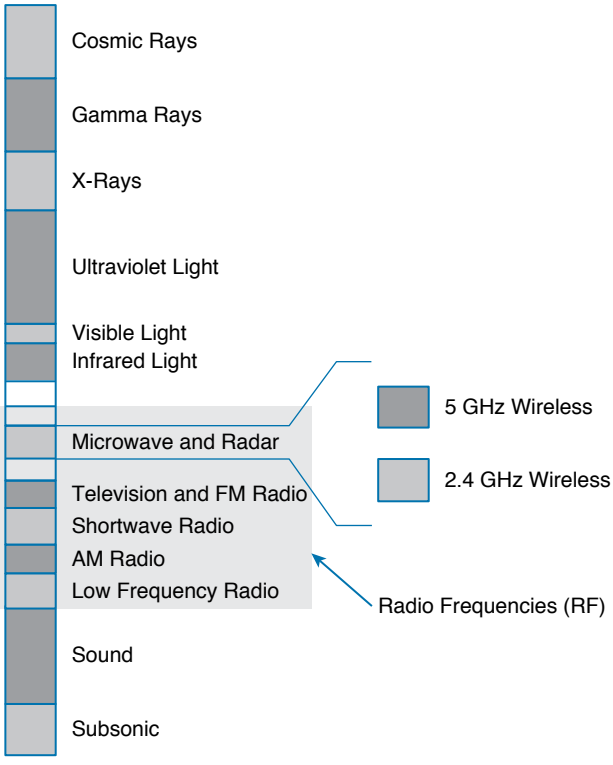


Figure 26-20 *Continuous Frequency Spectrum*

The frequency range from around 3 kHz to 300 GHz is commonly called *radio frequency* (RF). It includes many different types of radio communication, including low-frequency radio, AM radio, shortwave radio, television, FM radio, microwave, and radar. The microwave category also contains the two main frequency ranges that are used for wireless LAN communication: 2.4 and 5 GHz.

Wireless Bands and Channels

Because a range of frequencies might be used for the same purpose, it is customary to refer to the range as a *band* of frequencies. For example, the range from 530 kHz to around 1710 kHz is used by AM radio stations; therefore, it is commonly called the AM band or the AM broadcast band.

One of the two main frequency ranges used for wireless LAN communication lies between 2.400 and 2.4835 GHz. This is usually called the *2.4-GHz band*, even though it does not encompass the entire range between 2.4 and 2.5 GHz. It is much more convenient to refer to the band name instead of the specific range of frequencies included.

The other wireless LAN range is usually called the *5-GHz band* because it lies between 5.150 and 5.825 GHz. The 5-GHz band actually contains the following four separate and distinct bands:

5.150 to 5.250 GHz

5.250 to 5.350 GHz

5.470 to 5.725 GHz

5.725 to 5.825 GHz

TIP You might have noticed that most of the 5-GHz bands are contiguous except for a gap between 5.350 and 5.470. At the time of this writing, this gap exists and cannot be used for wireless LANs. However, some governmental agencies have moved to reclaim the frequencies and repurpose them for wireless LANs. Efforts are also underway to add 5.825 through 5.925 GHz.

It is interesting that the 5-GHz band can contain several smaller bands. Remember that the term *band* is simply a relative term that is used for convenience. Do not worry about memorizing the band names or exact frequency ranges; just be aware of the two main bands at 2.4 and 5 GHz.

A frequency band contains a continuous range of frequencies. If two devices require a single frequency for a wireless link between them, which frequency can they use? Beyond that, how many unique frequencies can be used within a band? To keep everything orderly and compatible, bands are usually divided into a number of distinct *channels*. Each channel is known by a channel number and is assigned to a specific frequency. As long as the channels are defined by a national or international standards body, they can be used consistently in all locations. Figures 26-21 and 26-22 show the channel layout for the 2.4 and 5 GHz bands, respectively.

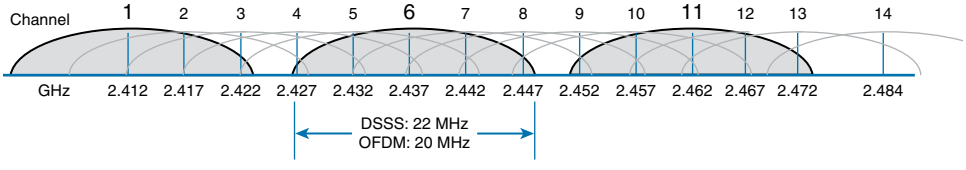


Figure 26-21 *Channel Layout in the 2.4-GHz Band*

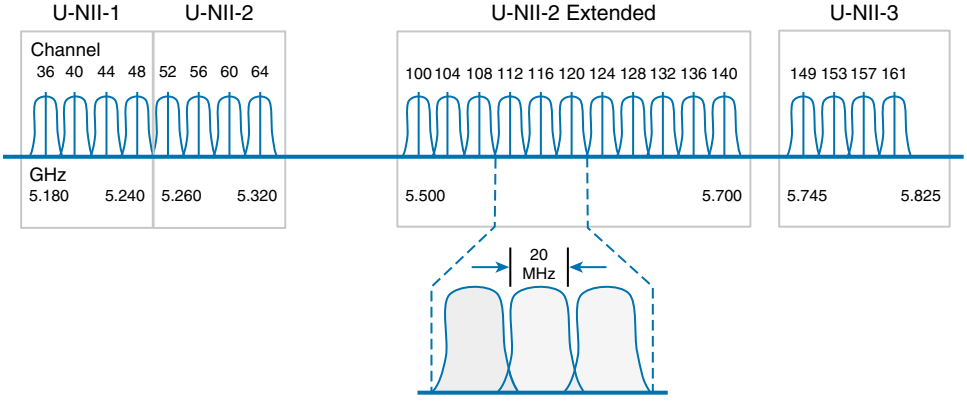


Figure 26-22 *Channel Layout in the 5-GHz Band*

You might assume that an AP can use any channel number without affecting any APs that use other channel numbers. In the 5-GHz band, this is the case because each channel is allocated a frequency range that does not encroach on or overlap the frequencies allocated for any other channel. In other words, the 5-GHz band consists of *nonoverlapping channels*.

Key Topic

The same is *not* true of the 2.4-GHz band. Each of its channels is much too wide to avoid overlapping the next lower or upper channel number. In fact, each channel covers the frequency range that is allocated to more than four consecutive channels! Notice the width of the channel spacing in Figure 26-21 as compared to the width of one of the shaded signals centered on channels 1, 6, and 11. The only way to avoid any overlap between adjacent channels is to configure APs to use only channels 1, 6, and 11. Even though there are 14 channels available to use, you should always strive for nonoverlapping channels in your network.

APs and Wireless Standards

It might be obvious that wireless devices and APs should all be capable of operating on the same band. For example, a 5-GHz wireless phone can communicate only with an AP that offers Wi-Fi service on 5-GHz channels. In addition, the devices and APs must also share a compatibility with the parts of the 802.11 standard they support.

As the IEEE 802.11 Wi-Fi standard evolves and develops, new amendments with new functionality get proposed. These amendments are known by "802.11" followed by a one- or two-letter suffix until they are accepted and rolled up into the next generation of the complete 802.11 standard. Even then, it is common to see the amendment suffixes still used to distinguish specific functions.

You should be aware of several amendments that define important characteristics such as data rates, methods used to transmit and receive data, and so on. For the CCNA 200-301 exam, you should know which band each of the amendments listed in Table 26-3 uses. The ENCOR 300-401 exam goes further into the data rates and modulation and coding schemes used by each.

Key Topic

Table 26-3 Basic Characteristics of Some IEEE 802.11 Amendments

Amendment	2.4 GHz	5 GHz	Max Data Rate	Notes
802.11-1997	Yes	No	2 Mbps	The original 802.11 standard ratified in 1997
802.11b	Yes	No	11 Mbps	Introduced in 1999
802.11g	Yes	No	54 Mbps	Introduced in 2003
802.11a	No	Yes	54 Mbps	Introduced in 1999
802.11n	Yes	Yes	600 Mbps	HT (high throughput), introduced in 2009
802.11ac	No	Yes	6.93 Gbps	VHT (very high throughput), introduced in 2013
802.11ax	Yes	Yes	4x 802.11ac	High Efficiency Wireless, Wi-Fi6; expected late 2019; will operate on other bands too, as they become available

The 802.11 amendments are not mutually exclusive. Wireless client devices and APs can be compatible with one or more amendments; however, a client and an AP can communicate only if they both support and agree to use the same amendment. When you look at the specifications for a wireless device, you may find supported amendments listed in a single string, separated by slashes. For example, a device that supports 802.11b/g will support both 802.11b and 802.11g. One that supports b/g/a/n/ac will support 802.11b, 802.11g, 802.11n, and 802.11ac. You should become familiar with Table 26-3 so that you can know which bands a device can use based on its 802.11 amendment support.

If a device can operate on both bands, how does it decide which band to use? APs can usually operate on both bands simultaneously to support any clients that might be present on each band. However, wireless clients typically associate with an AP on one band at a time, while scanning for potential APs on both bands. The band used to connect to an AP is chosen according to the operating system, wireless adapter driver, and other internal configuration. A wireless client can initiate an association with an AP on one band and then switch to the other band if the signal conditions are better there.

NOTE Cisco APs have dual radios (sets of transmitters and receivers) to support BSSs on one 2.4-GHz channel and other BSSs on one 5-GHz channel simultaneously. Some models also have two 5-GHz radios that can be configured to operate BSSs on two different channels at the same time, providing wireless coverage to higher densities of users that are located in the same vicinity.

You can configure a Cisco AP to operate on a specific channel number. As the number of APs grows, manual channel assignment can become a difficult task. Fortunately, Cisco wireless architectures can automatically and dynamically assign each AP to an appropriate channel. The architecture is covered in Chapter 27, "Analyzing Cisco Wireless Architectures," while dynamic channel assignment is covered on the ENCOR 300-401 exam.

In open space, RF signals propagate or reach further on the 2.4-GHz band than on the 5-GHz band. They also tend to penetrate indoor walls and objects easier at 2.4 GHz than 5 GHz. However, the 2.4-GHz band is commonly more crowded with wireless devices. Remember that only three nonoverlapping channels are available, so the chances of other neighboring APs using the same channels is greater. In contrast, the 5-GHz band has many more channels available to use, making channels less crowded and experiencing less interference.

Chapter Review

Review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 26-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 26-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Key Topic

Table 26-5 Key Topics for Chapter 26

Key Topic Element	Description	Page Number
Figure 26-4	Basic service set	615
Figure 26-7	Multiple SSIDs	618
Figure 26-8	Extended service set	619
Paragraph	Nonoverlapping channels and bands	628
Table 26-3	Basic Characteristics of Some 802.11 Amendments	628

Key Terms You Should Know

access point (AP), ad hoc network, Band, basic service set (BSS), Basic Service Set Identifier (BSSID), channel, cell, distribution system (DS), extended service set (ESS), independent basic service set (IBSS), infrastructure mode, mesh network, nonoverlapping channels, point-to-point bridge, repeater, roaming, Service Set Identifier (SSID), station (STA), workgroup bridge (WGB)

This page intentionally left blank

CHAPTER 27

Analyzing Cisco Wireless Architectures

This chapter covers the following exam topics:

2.0 Network Access

2.6 Compare Cisco Wireless Architectures and AP modes

In Chapter 26, "Fundamentals of Wireless Networks," you learned about how a single access point (AP) can provide a basic service set (BSS) for a cell area and how multiple APs can be connected to form an extended service set (ESS) for a larger network. In this chapter, you learn more about different approaches or architectures that allow APs to be networked together for an enterprise. You also learn how some architectures are more scalable than others and how to manage each type of wireless network architecture.

As you work through this chapter, think about how each architecture can be applied to specific environments—how easy it would be to manage, deploy, and troubleshoot the network, how the APs can be controlled, and how data would move through the network.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 27-1 "Do I Know This Already?" Section-to-Question Mapping

Foundation Topics Section	Questions
Autonomous AP Architectures	1
Cloud-based AP Architecture	2
Split-MAC Architectures	3–5
Comparing Wireless LAN Controller Deployments	6
Cisco AP Modes	7–8

1. Which one of the following terms best describes a Cisco wireless access point that operates in a standalone, independent manner?
   - **a.** Autonomous AP
   - **b.** Independent AP
   - **c.** Lightweight AP
   - **d.** Embedded AP

2. The Cisco Meraki cloud-based APs are most accurately described by which one of the following statements?
   a. Autonomous APs joined to a WLC
   b. Autonomous APs centrally managed
   c. Lightweight APs joined to a WLC
   d. Lightweight APs centrally managed
3. A lightweight access point is said to participate in which one of the following architectures?
   a. Light-MAC
   b. Tunnel-MAC
   c. Split-MAC
   d. Big-MAC
4. How does a lightweight access point communicate with a wireless LAN controller?
   a. Through an IPsec tunnel
   b. Through a CAPWAP tunnel
   c. Through a GRE tunnel
   d. Directly over Layer 2
5. Which one of the following is not needed for a lightweight AP in default local mode to be able to support three SSIDs that are bound to three VLANs?
   a. A trunk link carrying three VLANs
   b. An access link bound to a single VLAN
   c. A WLC connected to three VLANs
   d. A CAPWAP tunnel to a WLC
6. Which one of the following WLC deployment models would be best for a large enterprise with around 4000 lightweight APs?
   a. Cisco Mobility Express
   b. Embedded
   c. Unified
   d. Cloud-based
7. If a lightweight AP provides at least one BSS for wireless clients, which one of the following modes does it use?
   a. Local
   b. Normal
   c. Monitor
   d. Client

8. Regarding lightweight AP modes, which one of the following is true?

   a. An AP can operate in multiple modes at the same time.
   b. An AP only has one possible mode of operation.
   c. The Run mode is the default mode.
   d. The SE-Connect mode is used for spectrum analysis.

Foundation Topics

Autonomous AP Architecture

An access point's primary function is to bridge wireless data from the air to a normal wired network. An AP can accept "connections" from a number of wireless clients so that they become members of the LAN, as if the same clients were using wired connections.

APs act as the central point of access (hence the AP name), controlling client access to the wireless LAN. An *autonomous AP* is self-contained; it is equipped with both wired and wireless hardware so that the wireless client associations can be terminated onto a wired connection locally at the AP. The APs and their data connections must be distributed across the coverage area and across the network.

Autonomous APs offer one or more fully functional, standalone basic service sets (BSSs). They are also a natural extension of a switched network, connecting wireless service set identifiers (SSIDs) to wired virtual LANs (VLANs) at the access layer. Figure 27-1 shows the basic architecture; even though only four APs are shown across the bottom, a typical enterprise network could consist of hundreds or thousands of APs.

Key Topic

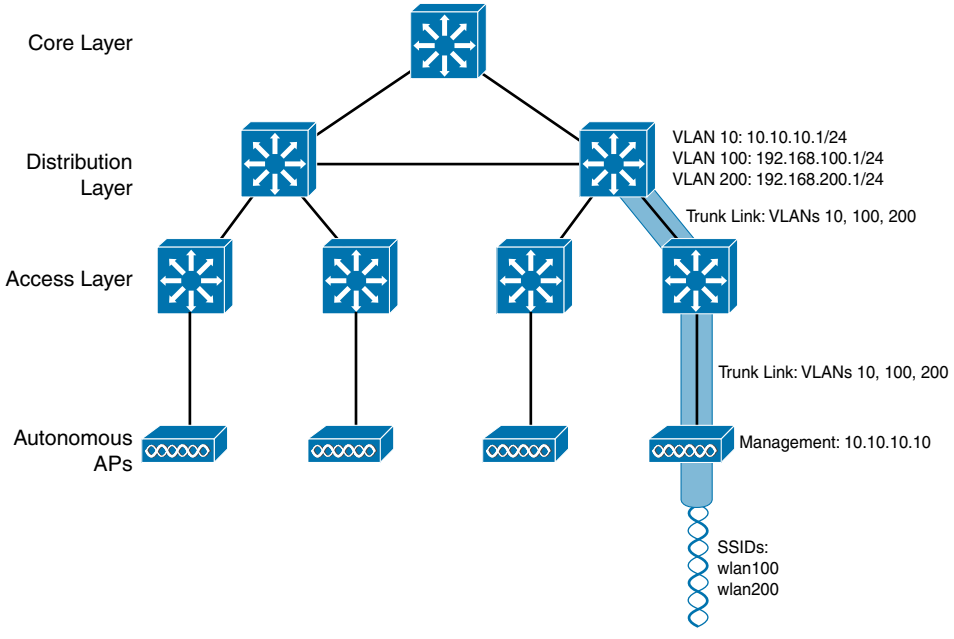


Figure 27-1 *Wireless Network Architecture with Autonomous APs*

What exactly does an autonomous AP need to become a part of the network? The wireless network in Figure 27-1 consists of two SSIDs: wlan100 and wlan200. These correspond to wired VLANs 100 and 200, respectively. As shown by the shaded links, the VLANs must be trunked from the distribution layer switch (where routing commonly takes place) to the access layer, where they are extended further over a trunk link to the AP.

An autonomous AP offers a short and simple path for data to travel between the wireless and wired networks. Data has to travel only through the AP to reach the network on the other side. Two wireless users that are associated to the same autonomous AP can reach each other through the AP without having to pass up into the wired network. As you work through the wireless architectures discussed in the rest of the chapter, notice the data path that is required for each.

An autonomous AP must also be configured with a management IP address (10.10.10.10 in Figure 27-1) so that you can remotely manage it. After all, you will want to configure SSIDs, VLANs, and many RF parameters like the channel and transmit power to be used. The management address is not normally part of any of the data VLANs, so a dedicated management VLAN (i.e., VLAN 10) must be added to the trunk links to reach the AP. Each AP must be configured and maintained individually unless you leverage a management platform such as Cisco Prime Infrastructure or Cisco DNA Center.

Because the data and management VLANs may need to reach every autonomous AP, the network configuration and efficiency can become cumbersome as the network scales. For example, you will likely want to offer the same SSID on many APs so that wireless clients can associate with that SSID in most any location or while roaming between any two APs. You might also want to extend the corresponding VLAN (and IP subnet) to each and every AP so that clients do not have to request a new IP address for each new association.

Because SSIDs and their VLANs must be extended at Layer 2, you should consider how they are extended throughout the switched network. The shaded links in Figure 27-2 show an example of a single VLAN's extent in the data plane. Working top to bottom, follow VLAN 100 as it reaches through the network. VLAN 100 is routed within the distribution layer and must be carried over trunk links to the access layer switches and then to each autonomous AP. In effect, VLAN 100 must extend end to end across the whole infrastructure—something that is usually considered to be a bad practice.

That might sound straightforward until you have to add a new VLAN and configure every switch and AP in your network to carry and support it. Even worse, suppose your network has redundant links between each layer of switches. The Spanning Tree Protocol (STP) running on each switch becomes a vital ingredient to prevent bridging loops from forming and corrupting the network. For these reasons, client roaming across autonomous APs is typically limited to the Layer 2 domain, or the extent of a single VLAN. As the wireless network expands, the infrastructure becomes more difficult to configure correctly and becomes less efficient.

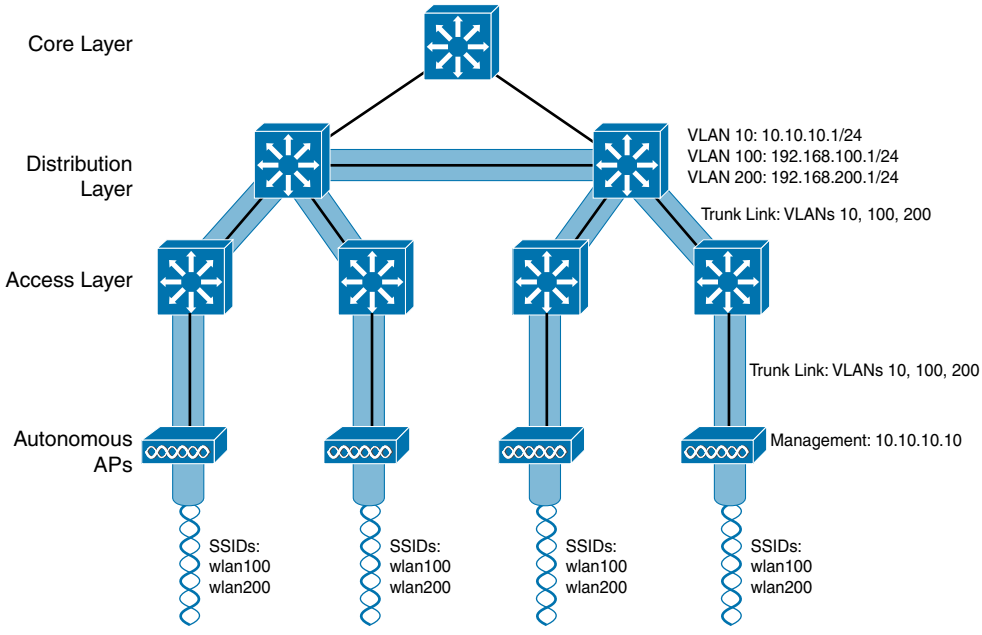


Figure 27-2 *Extent of a Data VLAN in a Network of Autonomous APs*

Cloud-based AP Architecture

Recall that an autonomous AP needs quite a bit of configuration and management. To help manage more and more autonomous APs as the wireless network grows, you could place an AP management platform such as Cisco Prime Infrastructure in a central location within the enterprise. The management platform would need to be purchased, configured, and maintained too.

A simpler approach is a *cloud-based AP* architecture, where the AP management function is pushed out of the enterprise and into the Internet cloud. Cisco Meraki is cloud-based and offers centralized management of wireless, switched, and security networks built from Meraki products. For example, through the cloud networking service, you can configure and manage APs, monitor wireless performance and activity, generate reports, and so on.

Cisco Meraki APs can be deployed automatically, once you register with the Meraki cloud. Each AP will contact the cloud when it powers up and will self-configure. From that point on, you can manage the AP through the Meraki cloud dashboard.

Figure 27-3 illustrates the basic cloud-based architecture. Notice that the network is arranged identically to that of the autonomous AP network. The reason is that the APs in a cloud-based network are all autonomous, too. The most visible difference is that all of the APs are managed, controlled, and monitored centrally from the cloud.

Answers to the "Do I Know This Already?" quiz:

1 A **2** B **3** C **4** B **5** A **6** C **7** A **8** D

Key Topic

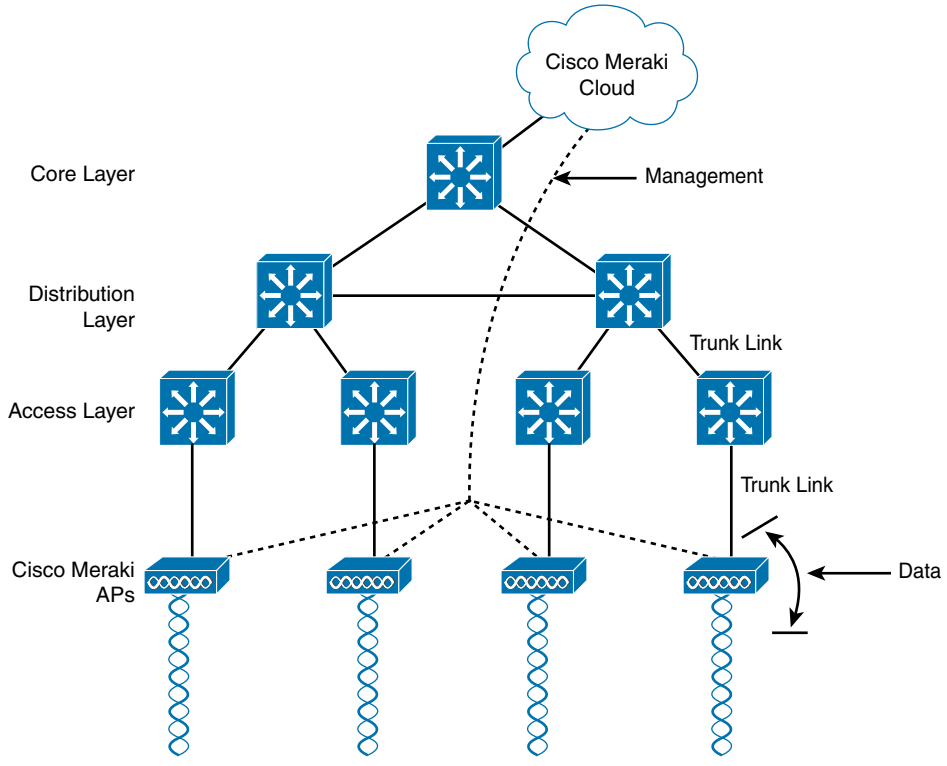


Figure 27-3 *Cisco Meraki Cloud-Based Wireless Network Architecture*

From the cloud, you can push out code upgrades and configuration changes to the APs in the enterprise. The Cisco Meraki cloud also adds the intelligence needed to automatically instruct each AP on which channel and transmit power level to use. It can also collect information from all of the APs about things such as RF interference, rogue or unexpected wireless devices that were overheard, and wireless usage statistics.

Finally, there are a couple of things you should observe about the cloud-based architecture. The data path from the wireless network to the wired network is very short; the autonomous AP links the two networks. Data to and from wireless clients does not have to travel up into the cloud and back; the cloud is used to bring management functions into the data plane.

Also, notice that the network in Figure 27-3 consists of two distinct paths—one for data traffic and another for management traffic, corresponding to the following two functions:

- **A control plane:** Traffic used to control, configure, manage, and monitor the AP itself
- **A data plane:** End-user traffic passing through the AP

This division will become important in the following sections as other types of architecture are discussed.

Split-MAC Architectures

Because autonomous APs are…well, autonomous, managing their RF operation can be quite difficult. As a network administrator, you are in charge of selecting and configuring the channel used by each AP and detecting and dealing with any rogue APs that might be interfering. You must also manage things such as the transmit power level to make sure that the wireless coverage is sufficient, it does not overlap too much, and there aren't any coverage holes—even when an AP's radio fails.

Managing wireless network security can also be difficult. Each autonomous AP handles its own security policies, with no central point of entry between the wireless and wired networks. That means there is no convenient place to monitor traffic for things such as intrusion detection and prevention, quality of service, bandwidth policing, and so on.

To overcome the limitations of distributed autonomous APs, many of the functions found within autonomous APs have to be shifted toward some central location. In Figure 27-4, most of the activities performed by an autonomous AP on the left are broken up into two groups—management functions on the top and real-time processes on the bottom.

Key Topic

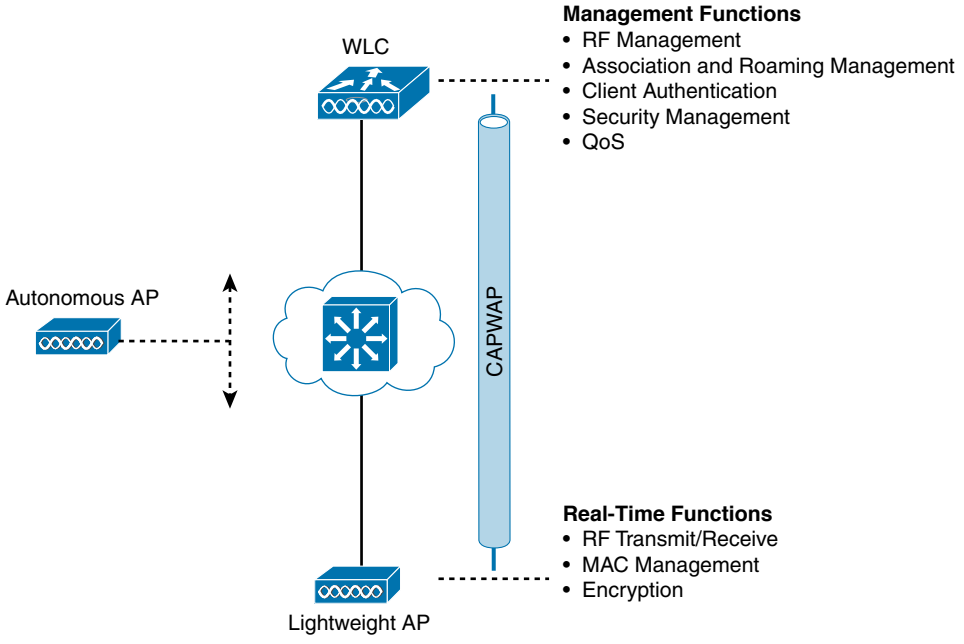


Figure 27-4 *Autonomous Versus Lightweight Access Point*

The real-time processes involve sending and receiving 802.11 frames, beacons, and probe messages. 802.11 data encryption is also handled in real time, on a per-packet basis. The AP must interact with wireless clients on some low level, known as the *Media Access Control* (MAC) layer. These functions must stay with the AP hardware, closest to the clients.

The management functions are not integral to handling frames over the RF channels, but are things that should be centrally administered. Therefore, those functions can be moved to a centrally located platform away from the AP.

When the functions of an autonomous AP are divided, the AP hardware is known as a *lightweight access point*, and performs only the real-time 802.11 operation. The lightweight AP gets its name because the code image and the local intelligence are stripped down, or lightweight, compared to the traditional autonomous AP.

The management functions are usually performed on a *wireless LAN controller* (WLC), which controls many lightweight APs. This is shown in the bottom right portion of Figure 27-4. Notice that the AP is left with duties in Layers 1 and 2, where frames are moved into and out of the RF domain. The AP becomes totally dependent on the WLC for every other WLAN function, such as authenticating users, managing security policies, and even selecting RF channels and output power.

NOTE Remember that a lightweight AP cannot normally operate on its own; it is very dependent on a WLC somewhere in the network. The only exception is the FlexConnect architecture, which is discussed later in this chapter.

The lightweight AP-WLC division of labor is known as a *split-MAC architecture*, where the normal MAC operations are pulled apart into two distinct locations. This occurs for every AP in the network; each one must boot and bind itself to a WLC to support wireless clients. The WLC becomes the central hub that supports a number of APs scattered about in the network.

How does a lightweight AP bind with a WLC to form a complete working access point? The two devices must use a tunneling protocol between them, to carry 802.11-related messages and also client data. Remember that the AP and WLC can be located on the same VLAN or IP subnet, but they do not have to be. Instead, they can be located on two entirely different IP subnets in two entirely different locations.

The Control and Provisioning of Wireless Access Points (*CAPWAP*) tunneling protocol makes this all possible by encapsulating the data between the LAP and WLC within new IP packets. The tunneled data can then be switched or routed across the campus network. As Figure 27-5 shows, the CAPWAP relationship actually consists of two separate tunnels, as follows:

- **CAPWAP control messages:** Carries exchanges that are used to configure the AP and manage its operation. The control messages are authenticated and encrypted, so the AP is securely controlled by only the appropriate WLC, then transported over the control tunnel.
- **CAPWAP data:** Used for packets traveling to and from wireless clients that are associated with the AP. Data packets are transported over the data tunnel but are not encrypted by default. When data encryption is enabled for an AP, packets are protected with Datagram Transport Layer Security (DTLS).

NOTE CAPWAP is defined in RFCs 5415, 5416, 5417, and 5418. CAPWAP is based on the Lightweight Access Point Protocol (LWAPP), which was a legacy Cisco proprietary solution.

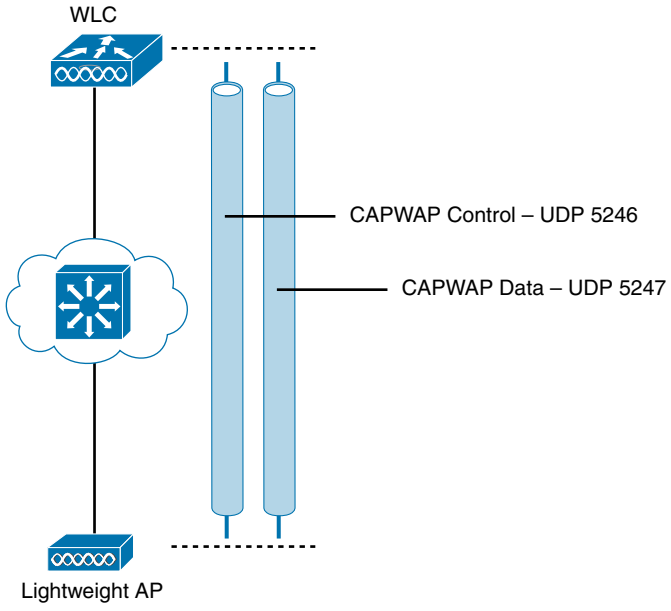


Figure 27-5 *Linking a Lightweight AP and WLC with CAPWAP*

Every AP and WLC must also authenticate each other with digital certificates. An X.509 certificate is preinstalled in each device when it is purchased. By using certificates behind the scenes, every device is properly authenticated before becoming part of the wireless network. This process helps assure that no one can add an unauthorized AP to your network.

The CAPWAP tunneling allows the AP and WLC to be separated geographically and logically. It also breaks the dependence on Layer 2 connectivity between them. For example, Figure 27-6 uses shaded areas to show the extent of VLAN 100. Notice how VLAN 100 exists at the WLC and in the air as SSID 100, near the wireless clients—but not in between the AP and the WLC. Instead, traffic to and from clients associated with SSID 100 is transported across the network infrastructure encapsulated inside the CAPWAP data tunnel. The tunnel exists between the IP address of the WLC and the IP address of the AP, which allows all of the tunneled packets to be routed at Layer 3.

Also, notice how the AP is known by only a single IP address: 10.10.10.10. Because the AP sits on the access layer where its CAPWAP tunnels terminate, it can use one IP address for both management and tunneling. No trunk link is needed because all of the VLANs it supports are encapsulated and tunneled as Layer 3 IP packets, rather than individual Layer 2 VLANs.

As the wireless network grows, the WLC simply builds more CAPWAP tunnels to reach more APs. Figure 27-7 depicts a network with four APs. Each AP has a control and a data tunnel back to the centralized WLC. SSID 100 can exist on every AP, and VLAN 100 can reach every AP through the network of tunnels.

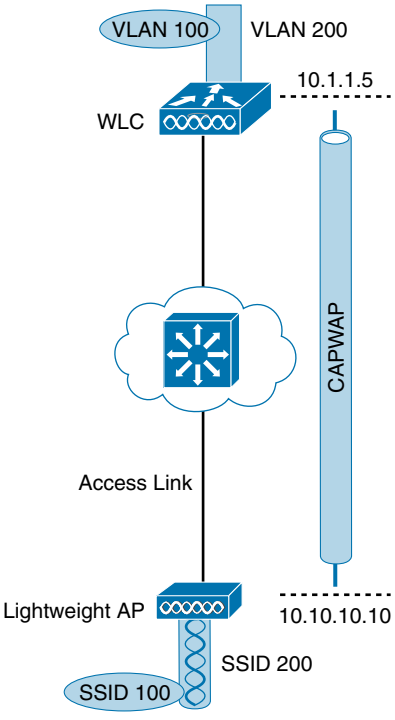


Figure 27-6 *Extent of VLAN 100 in a Cisco Wireless Network*

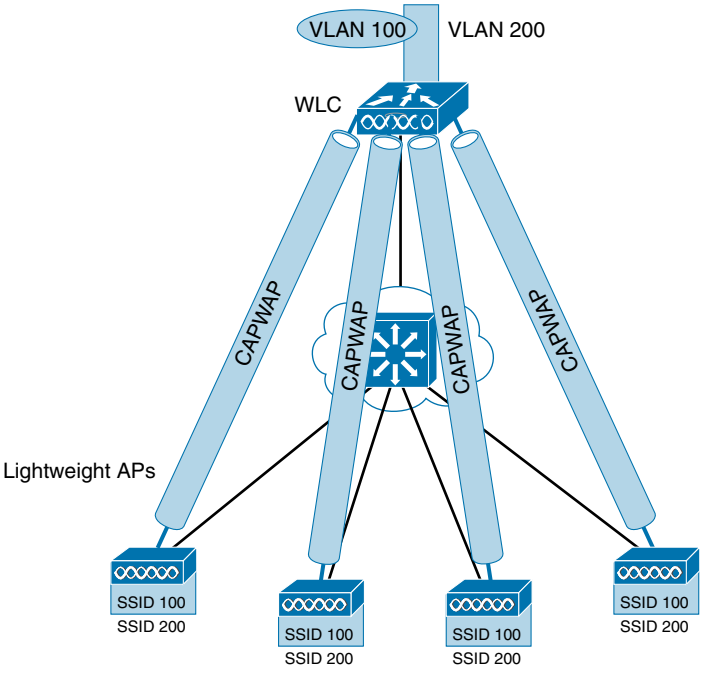


Figure 27-7 *Using CAPWAP Tunnels to Connect APs to One Central WLC*

Once CAPWAP tunnels are built from a WLC to one or more lightweight APs, the WLC can begin offering a variety of additional functions. Think of all the puzzles and shortcomings that were discussed for the traditional autonomous WLAN architecture as you read over the following list of WLC activities:

- **Dynamic channel assignment:** The WLC can automatically choose and configure the RF channel used by each AP, based on other active access points in the area.
- **Transmit power optimization:** The WLC can automatically set the transmit power of each AP based on the coverage area needed.
- **Self-healing wireless coverage:** If an AP radio dies, the coverage hole can be "healed" by turning up the transmit power of surrounding APs automatically.
- **Flexible client roaming:** Clients can roam between APs with very fast roaming times.
- **Dynamic client load balancing:** If two or more APs are positioned to cover the same geographic area, the WLC can associate clients with the least used AP. This distributes the client load across the APs.
- **RF monitoring:** The WLC manages each AP so that it scans channels to monitor the RF usage. By listening to a channel, the WLC can remotely gather information about RF interference, noise, signals from neighboring APs, and signals from rogue APs or ad hoc clients.
- **Security management:** The WLC can authenticate clients from a central service and can require wireless clients to obtain an IP address from a trusted DHCP server before allowing them to associate and access the WLAN.
- **Wireless intrusion protection system:** Leveraging its central location, the WLC can monitor client data to detect and prevent malicious activity.

Comparing Wireless LAN Controller Deployments

Suppose you want to deploy a WLC to support multiple lightweight APs in your network. Where should you put the WLC? The split-MAC concept can be applied to several different network architectures. Each architecture places the WLC in a different location within the network—a choice that also affects how many WLCs might be needed to support the number of APs required.

One approach is to locate the WLC in a central location so that you can maximize the number of APs joined to it. This is usually called a *unified* or *centralized WLC deployment*, which tends to follow the concept that most of the resources users need to reach are located in a central location such as a data center or the Internet. Traffic to and from wireless users would travel over CAPWAP tunnels that reach into the center of the network, near the core, as shown in Figure 27-8. A centralized WLC also provides a convenient place to enforce security policies that affect all wireless users.

Key Topic

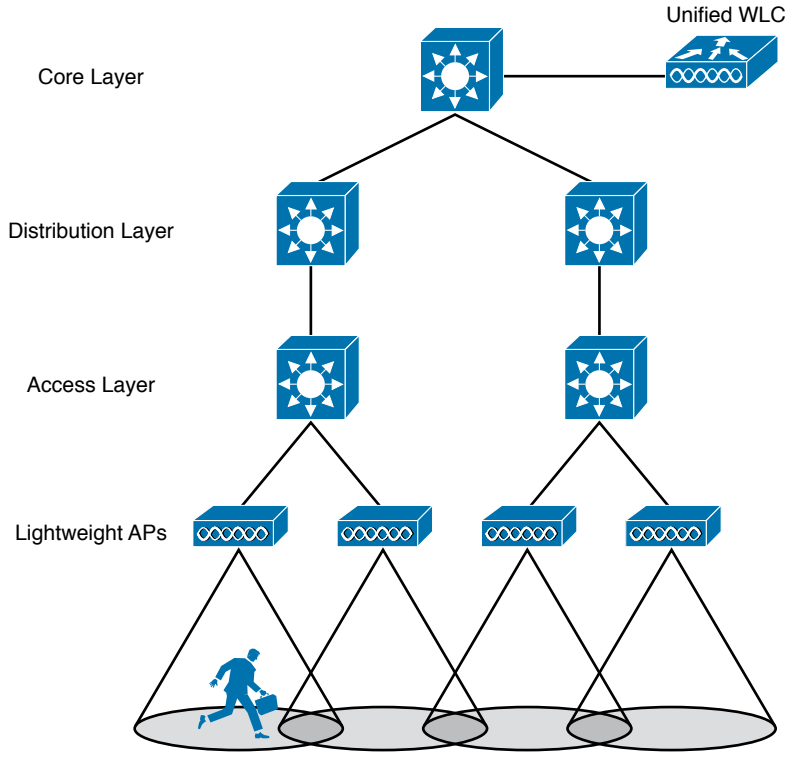


Figure 27-8 *WLC Location in a Unified Deployment*

Figure 27-8 shows four APs joined to a single WLC. Your network might have more APs—many, many more. A large enterprise network might have thousands of APs connected to its access layer. Scalability then becomes an important factor in the centralized design. Typical unified WLCs can support a maximum of 6000 APs. If you have more APs than the maximum, you will need to add more WLCs to the design, each located centrally.

A WLC can also be located in a central position in the network, inside a data center in a private cloud, as shown in Figure 27-9. This is known as a *cloud-based WLC deployment*, where the WLC exists as a virtual machine rather than a physical device. If the cloud computing platform already exists, then deploying a cloud-based WLC becomes straightforward. Such a controller can typically support up to 3000 APs. If your wireless network scales beyond that, then additional WLCs can be added as more virtual machines.

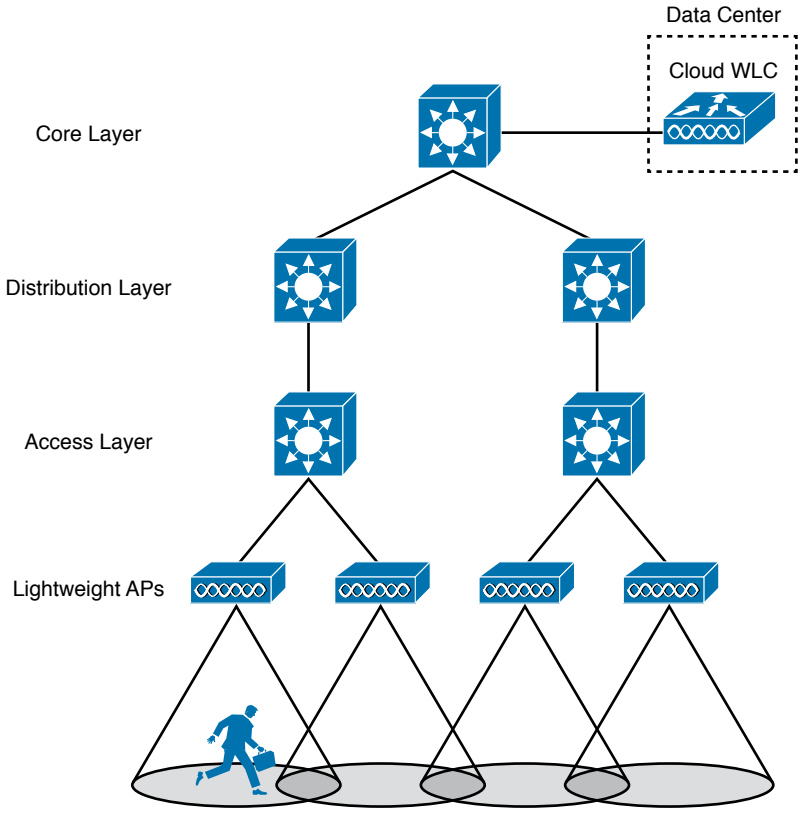


Figure 27-9 *WLC Location in a Cloud-based Deployment*

For small campuses or distributed branch locations, where the number of APs is relatively small in each, the WLC can be co-located with a stack of switches, as shown in Figure 27-10. This is known as an *embedded WLC deployment* because the controller is embedded within the switching hardware. Typical Cisco embedded WLCs can support up to 200 APs. The APs do not necessarily have to be connected to the switches that host the WLC; APs connected to other switches in other locations can join the embedded WLC too. As the number of APs grows, additional WLCs can be added by embedding them in other switch stacks at the site.

Key Topic

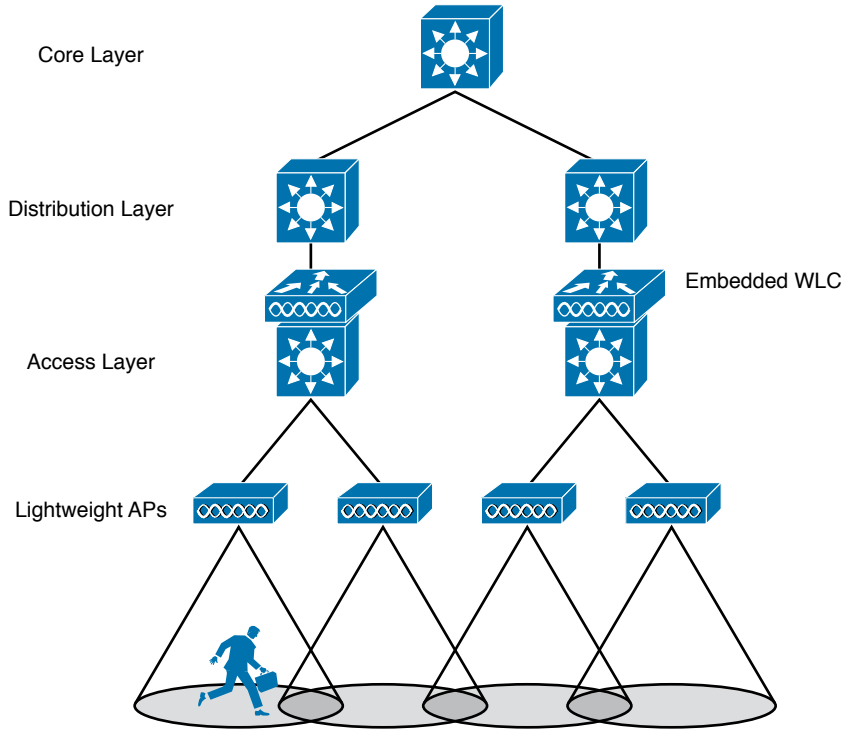


Figure 27-10 *WLC Location in an Embedded Deployment*

Finally, in small-scale environments, such as small, midsize, or multisite branch locations, you might not want to invest in dedicated WLCs at all. In this case, the WLC function can be co-located with an AP that is installed at the branch site. This is known as a Cisco *Mobility Express WLC deployment*, as shown in Figure 27-11. The AP that hosts the WLC forms a CAPWAP tunnel with the WLC, along with any other APs at the same location. A Mobility Express WLC can support up to 100 APs.

Key Topic

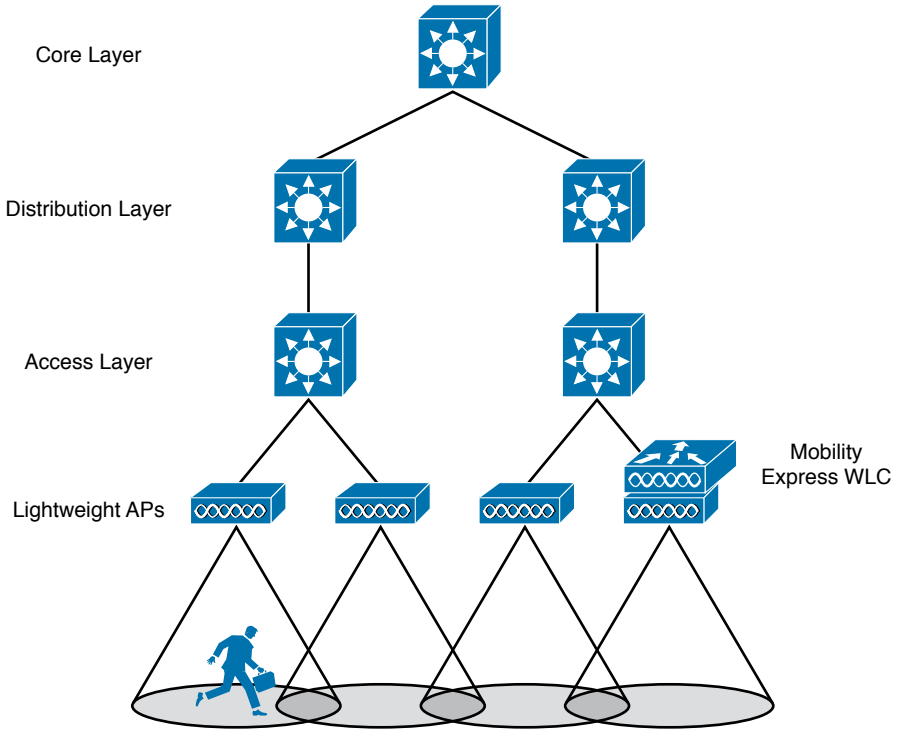


Figure 27-11 *WLC Location in a Mobility Express Deployment*

See Table 27-2 for a summary of WLC deployment models, WLC locations, and a typical maximum number of APs and clients that each one supports.

Table 27-2 Summary of WLC Deployment Models

Deployment Model	WLC Location (DC, Access, Central, AP)	APs Supported	Clients Supported	Typical Use
Unified	Central	6000	64,000	Large enterprise
Cloud	DC	3000	32,000	Private cloud
Embedded	Access	200	4000	Small campus
Mobility Express	Other	100	2000	Branch location
Autonomous	N/A	N/A	N/A	N/A

Cisco AP Modes

Many Cisco APs can operate in either autonomous or lightweight mode, depending on which code image is loaded and run. From the WLC, you can also configure a lightweight AP to operate in one of the following special-purpose modes:

Key Topic

- **Local:** The default lightweight mode that offers one or more functioning BSSs on a specific channel. During times that it is not transmitting, the AP will scan the other channels to measure the level of noise, measure interference, discover rogue devices, and match against intrusion detection system (IDS) events.
- **Monitor:** The AP does not transmit at all, but its receiver is enabled to act as a dedicated sensor. The AP checks for IDS events, detects rogue access points, and determines the position of stations through location-based services.
- **FlexConnect:** An AP at a remote site can locally switch traffic between an SSID and a VLAN if its CAPWAP tunnel to the WLC is down and if it is configured to do so.
- **Sniffer:** An AP dedicates its radios to receiving 802.11 traffic from other sources, much like a sniffer or packet capture device. The captured traffic is then forwarded to a PC running network analyzer software such as Wildpackets OmniPeek or WireShark, where it can be analyzed further.
- **Rogue detector:** An AP dedicates itself to detecting rogue devices by correlating MAC addresses heard on the wired network with those heard over the air. Rogue devices are those that appear on both networks.
- **Bridge:** An AP becomes a dedicated bridge (point-to-point or point-to-multipoint) between two networks. Two APs in bridge mode can be used to link two locations separated by a distance. Multiple APs in bridge mode can form an indoor or outdoor mesh network.
- **Flex+Bridge:** FlexConnect operation is enabled on a mesh AP.
- **SE-Connect:** The AP dedicates its radios to spectrum analysis on all wireless channels. You can remotely connect a PC running software such as MetaGeek Chanalyzer or Cisco Spectrum Expert to the AP to collect and analyze the spectrum analysis data to discover sources of interference.

NOTE Remember that a lightweight AP is normally in local mode when it is providing BSSs and allowing client devices to associate to wireless LANs. When an AP is configured to operate in one of the other modes, local mode (and the BSSs) is disabled.

Chapter Review

Review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 27-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 27-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Review the most important topics in this chapter, noted with the Key Topic icon in the outer margin of the page. Table 27-4 lists a reference of these key topics and the page numbers on which each is found.

Key Topic

Table 27-4 Key Topics for Chapter 28

Key Topic Element	Description	Page Number
Figure 27-1	Autonomous AP architecture	634
Figure 27-3	Cloud-based AP architecture	637
Figure 27-4	Split-MAC architecture	638
Figure 27-5	CAPWAP tunnels	640
Figure 27-8	Unified WLC deployment	643
Figure 27-9	Cloud-based WLC deployment	644
Figure 27-10	Embedded WLC deployment	645
Figure 27-11	Mobility Express WLC deployment	646
List	Cisco lightweight AP modes	647

Key Terms You Should Know

autonomous AP, CAPWAP, centralized WLC deployment, cloud-based AP, cloud-based WLC deployment, embedded WLC deployment, lightweight AP, local mode, Media Access Control (MAC) layer, Mobility Express WLC deployment, split-MAC architecture, unified WLC deployment, wireless LAN controller (WLC)

This page intentionally left blank

CHAPTER 28

Securing Wireless Networks

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.11 Describe wireless principles

1.11.d Encryption

5.0 Security Fundamentals

5.9 Describe wireless security protocols (WPA, WPA2, and WPA3)

As you know by now, wireless networks are complex. Many technologies and protocols work behind the scenes to give end users a stable, yet mobile, connection to a wired network infrastructure. From the user's perspective, a wireless connection should seem no different than a wired connection. A wired connection can give users a sense of security; data traveling over a wire is probably not going to be overheard by others. A wireless connection is inherently different; data traveling over the air can be overheard by anyone within range.

Therefore, securing a wireless network becomes just as important as any other aspect. A comprehensive approach to wireless security focuses on the following areas:

- Identifying the endpoints of a wireless connection
- Identifying the end user
- Protecting the wireless data from eavesdroppers
- Protecting the wireless data from tampering

The identification process is performed through various authentication schemes. Protecting wireless data involves security functions like encryption and frame authentication.

This chapter covers many of the methods you can use to secure a wireless network. Be warned: wireless security can be a confusing topic because it is filled with many acronyms. Some of the acronyms rhyme like words from a children's book. In fact, this chapter is a story about WEP, PSK, TKIP, MIC, AES, EAP, EAP-FAST, EAP-TLS, LEAP, PEAP, WPA, WPA2, WPA3, CCMP, GCMP, and on and on it goes. When you finish with this chapter, though, you will come away with a clearer view of what these terms mean and how they all fit together. You might even be ready to configure a wireless LAN with effective security.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 28-1 "Do I Know This Already?" Section-to-Question Mapping

Foundation Topics Section	Questions
Anatomy of a Secure Connection	1–2
Wireless Client Authentication Methods	3–4
Wireless Privacy and Integrity Methods	5–6
WPA, WPA2, and WPA3	7–8

1. Which of the following are necessary components of a secure wireless connection? (Choose all that apply.)
   - **a.** Encryption
   - **b.** MIC
   - **c.** Authentication
   - **d.** All of these answers are correct.
2. Which one of the following is used to protect the integrity of data in a wireless frame?
   - **a.** WIPS
   - **b.** WEP
   - **c.** MIC
   - **d.** EAP
3. Which one of the following is a wireless encryption method that has been found to be vulnerable and is not recommended for use?
   - **a.** AES
   - **b.** WPA
   - **c.** EAP
   - **d.** WEP
4. Which one of the following is used as the authentication framework when 802.1x is used on a WLAN?
   - **a.** Open authentication
   - **b.** WEP
   - **c.** EAP
   - **d.** WPA

5. Suppose you would like to select a method to protect the privacy and integrity of wireless data. Which one of the following methods should you avoid because it has been deprecated ?
   a. TKIP
   b. CCMP
   c. GCMP
   d. EAP
6. Which one of the following is the data encryption and integrity method used by WPA2?
   a. WEP
   b. TKIP
   c. CCMP
   d. WPA
7. The Wi-Fi Alliance offers which of the following certifications for wireless devices that correctly implement security standards? (Choose all that apply.)
   a. WEP
   b. WPA2
   c. 802.11
   d. AES
8. A pre-shared key is used in which of the following wireless security configurations? (Choose all that apply.)
   a. WPA2 personal mode
   b. WPA2 enterprise mode
   c. WPA3 personal mode
   d. WPA3 enterprise mode

Foundation Topics

Anatomy of a Secure Connection

In the previous chapters of this book, you learned about wireless clients forming associations with wireless access points (APs) and passing data back and forth across the air.

As long as all clients and APs conform to the 802.11 standard, they can all coexist—even on the same channel. Not every 802.11 device is friendly and trustworthy, however. Sometimes it is easy to forget that transmitted frames do not just go directly from the sender to the receiver, as in a wired or switched connection. Instead, they travel according to the transmitter's antenna pattern, potentially reaching any receiver that is within range.

Consider the scenario in Figure 28-1. The wireless client opens a session with some remote entity and shares a confidential password. Because two untrusted users are also located within range of the client's signal, they may also learn the password by capturing frames that have been sent on the channel. The convenience of wireless communication also makes it easy for transmissions to be overheard and exploited by malicious users.

If data is sent through open space, how can it be secured so that it stays private and intact? The 802.11 standard offers a framework of wireless security mechanisms that can be used to add trust, privacy, and integrity to a wireless network. The following sections give an overview of the wireless security framework.

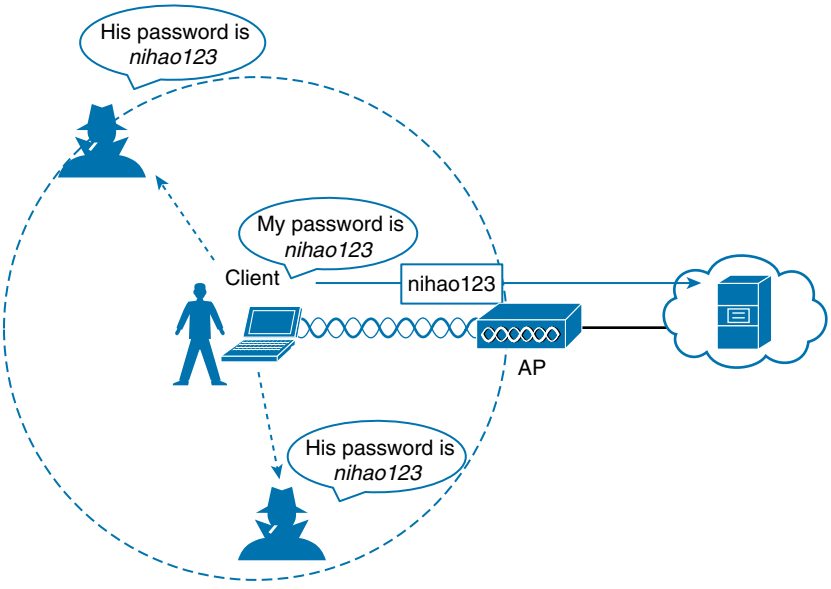


Figure 28-1 *Wireless Transmissions Reaching Unintended Recipients*

Authentication

To use a wireless network, clients must first discover a basic service set (BSS) and then request permission to associate with it. Clients should be authenticated by some means before they can become functioning members of the wireless LAN. Why?

Suppose that your wireless network connects to corporate resources where confidential information can be accessed. In that case, only devices known to be trusted and expected should be given access. Guest users, if they are permitted at all, should be allowed to join a different guest WLAN where they can access nonconfidential or public resources. Rogue clients, which are not expected or welcomed, should not be permitted to associate at all. After all, they are not affiliated with the corporate network and are likely to be unknown devices that happen to be within range of your network.

To control access, wireless networks can authenticate the client devices before they are allowed to associate. Potential clients must identify themselves by presenting some form of credentials to the APs. Figure 28-2 shows the basic client authentication process.

Wireless authentication can take many forms. Some methods require only a static text string that is common across all trusted clients and APs. The text string is stored on the client device and presented directly to the AP when needed. What might happen if the device was stolen or lost? Most likely, any user who possessed the device could still authenticate to the network. Other more stringent authentication methods require interaction with a corporate user database. In those cases, the end user must enter a valid username and password—something that would not be known to a thief or an imposter.

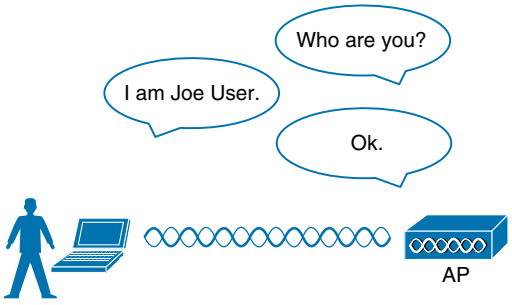


Figure 28-2 *Authenticating a Wireless Client*

If you have ever joined a wireless network, you might have focused on authenticating your device or yourself, while implicitly trusting the nearest AP. For example, if you turn on your wireless device and find a wireless network that is available at your workplace, you probably join it without hesitating. The same is true for wireless networks in an airport, a hotel, a hot spot, or in your home—you expect the AP that is advertising the SSID to be owned and operated by the entity where you are located. But how can you be sure?

Normally, the only piece of information you have is the SSID being broadcast or advertised by an AP. If the SSID looks familiar, you will likely choose to join it. Perhaps your computer is configured to automatically connect to a known SSID so that it associates without your intervention. Either way, you might unwittingly join the same SSID even if it was being advertised by an imposter.

Some common attacks focus on a malicious user pretending to be an AP. The fake AP can send beacons, answer probes, and associate clients just like the real AP it is impersonating. Once a client associates with the fake AP, the attacker can easily intercept all communication to and from the client from its central position. A fake AP could also send spoofed management frames to disassociate or deauthenticate legitimate and active clients, just to disrupt normal network operation.

To prevent this type of man-in-the-middle attack, the client should authenticate the AP before the client itself is authenticated. Figure 28-3 shows a simple scenario. Even further, any management frames received by a client should be authenticated too, as proof that they were sent by a legitimate and expected AP.

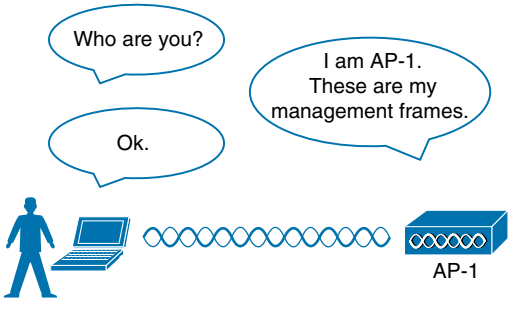


Figure 28-3 *Authenticating a Wireless AP*

Answers to the "Do I Know This Already?" quiz:

1 D **2** C **3** D **4** C **5** A **6** C **7** B **8** A, C

Message Privacy

Suppose that the client in Figure 28-3 must authenticate before joining the wireless network. It might also authenticate the AP and its management frames after it associates but before it is itself authenticated. The client's relationship with the AP might become much more trusted, but data passing to and from the client is still available to eavesdroppers on the same channel.

To protect data privacy on a wireless network, the data should be encrypted for its journey through free space. This is accomplished by encrypting the data payload in each wireless frame just prior to being transmitted, then decrypting it as it is received. The idea is to use an encryption method that the transmitter and receiver share, so the data can be encrypted and decrypted successfully.

In wireless networks, each WLAN may support only one authentication and encryption scheme, so all clients must use the same encryption method when they associate. You might think that having one encryption method in common would allow every client to eavesdrop on every other client. That is not necessarily the case because the AP should securely negotiate a unique encryption key to use for each associated client.

Ideally, the AP and a client are the only two devices that have the encryption keys in common so that they can understand each other's data. No other device should know about or be able to use the same keys to eavesdrop and decrypt the data. In Figure 28-4, the client's confidential password information has been encrypted before being transmitted. The AP can decrypt it successfully before forwarding it onto the wired network, but other wireless devices cannot.

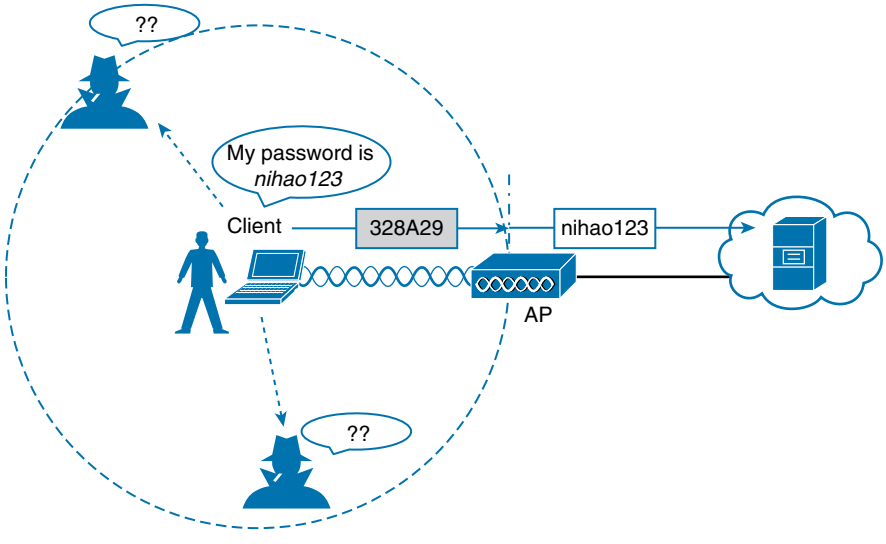


Figure 28-4 *Encrypting Wireless Data to Protect Data Privacy*

The AP also maintains a "group key" that it uses when it needs to send encrypted data to all clients in its cell at one time. Each of the associated clients uses the same group key to decrypt the data.

Message Integrity

Encrypting data obscures it from view while it is traveling over a public or untrusted network. The intended recipient should be able to decrypt the message and recover the original contents, but what if someone managed to alter the contents along the way? The recipient would have a very difficult time discovering that the original data had been modified.

A message integrity check (MIC) is a security tool that can protect against data tampering. You can think of a MIC as a way for the sender to add a secret stamp inside the encrypted data frame. The stamp is based on the contents of the data bits to be transmitted. Once the recipient decrypts the frame, it can compare the secret stamp to its own idea of what the stamp should be, based on the data bits that were received. If the two stamps are identical, the recipient can safely assume that the data has not been tampered with. Figure 28-5 shows the MIC process.

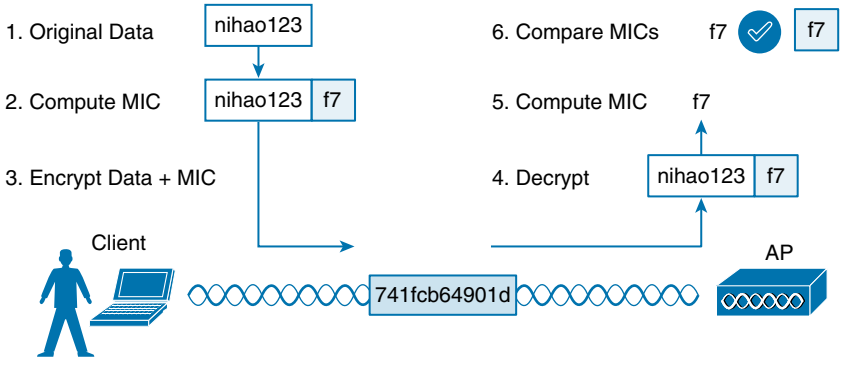


Figure 28-5 *Checking Message Integrity over a Wireless Network*

Wireless Client Authentication Methods

You can use many different methods to authenticate wireless clients as they try to associate with the network. The methods have been introduced over time and have evolved as security weaknesses have been exposed and wireless hardware has advanced. This section covers the most common authentication methods you might encounter.

Open Authentication

The original 802.11 standard offered only two choices to authenticate a client: open authentication and WEP.

Open authentication is true to its name; it offers open access to a WLAN. The only requirement is that a client must use an 802.11 authentication request before it attempts to associate with an AP. No other credentials are needed.

When would you want to use open authentication? After all, it does not sound very secure because it is not. With no challenge, any 802.11 client may authenticate to access the network. That is, in fact, the whole purpose of open authentication—to validate that a client is a valid 802.11 device by authenticating the wireless hardware and the protocol. Authenticating the user's identity is handled as a true security process through other means.

You have probably seen a WLAN with open authentication when you have visited a public location. If any client screening is used at all, it comes in the form of web authentication. A client can associate right away but must open a web browser to see and accept the terms for use and enter basic credentials. From that point, network access is opened up for the client. Most client operating systems flag such networks to warn you that your wireless data will not be secured in any way if you join.

WEP

As you might expect, open authentication offers nothing that can obscure or encrypt the data being sent between a client and an AP. As an alternative, the 802.11 standard has traditionally defined Wired Equivalent Privacy (WEP) as a method to make a wireless link more like or equivalent to a wired connection.

WEP uses the RC4 cipher algorithm to make every wireless data frame private and hidden from eavesdroppers. The same algorithm encrypts data at the sender and decrypts it at the receiver. The algorithm uses a string of bits as a key, commonly called a WEP key, to derive other encryption keys—one per wireless frame. As long as the sender and receiver have an identical key, one can decrypt what the other encrypts.

WEP is known as a shared-key security method. The same key must be shared between the sender and receiver ahead of time, so that each can derive other mutually agreeable encryption keys. In fact, every potential client and AP must share the same key ahead of time so that any client can associate with the AP.

The WEP key can also be used as an optional authentication method as well as an encryption tool. Unless a client can use the correct WEP key, it cannot associate with an AP. The AP tests the client's knowledge of the WEP key by sending it a random challenge phrase. The client encrypts the challenge phrase with WEP and returns the result to the AP. The AP can compare the client's encryption with its own to see whether the two WEP keys yield identical results.

WEP keys can be either 40 or 104 bits long, represented by a string of 10 or 26 hex digits. As a rule of thumb, longer keys offer more unique bits for the algorithm, resulting in more robust encryption. Except in WEP's case, that is. Because WEP was defined in the original 802.11 standard in 1999, every wireless adapter was built with encryption hardware specific to WEP. In 2001, a number of weaknesses were discovered and revealed, so work began to find better wireless security methods. By 2004, the 802.11i amendment was ratified and WEP was officially deprecated. Both WEP encryption and WEP shared-key authentication are widely considered to be weak methods to secure a wireless LAN.

802.1x/EAP

With only open authentication and WEP available in the original 802.11 standard, a more secure authentication method was needed. Client authentication generally involves some sort of challenge, a response, and then a decision to grant access. Behind the scenes, it can also involve an exchange of session or encryption keys, in addition to other parameters needed for client access. Each authentication method might have unique requirements as a unique way to pass information between the client and the AP.

Rather than build additional authentication methods into the 802.11 standard, a more flexible and scalable authentication framework, the Extensible Authentication Protocol (EAP), was chosen. As its name implies, EAP is extensible and does not consist of any one authentication method. Instead, EAP defines a set of common functions that actual authentication methods can use to authenticate users. As you read through this section, notice how many authentication methods have *EAP* in their names. Each method is unique and different, but each one follows the EAP framework.

EAP has another interesting quality: it can integrate with the IEEE 802.1x port-based access control standard. When 802.1x is enabled, it limits access to a network media until a client authenticates. This means that a wireless client might be able to associate with an AP but will not be able to pass data to any other part of the network until it successfully authenticates.

With open and WEP authentication, wireless clients are authenticated locally at the AP without further intervention. The scenario changes with 802.1x; the client uses open authentication to associate with the AP, and then the actual client authentication process occurs at a dedicated authentication server. Figure 28-6 shows the three-party 802.1x arrangement that consists of the following entities:

Key Topic

- **Supplicant:** The client device that is requesting access
- **Authenticator:** The network device that provides access to the network (usually a wireless LAN controller [WLC])
- **Authentication server (AS):** The device that takes user or client credentials and permits or denies network access based on a user database and policies (usually a RADIUS server)

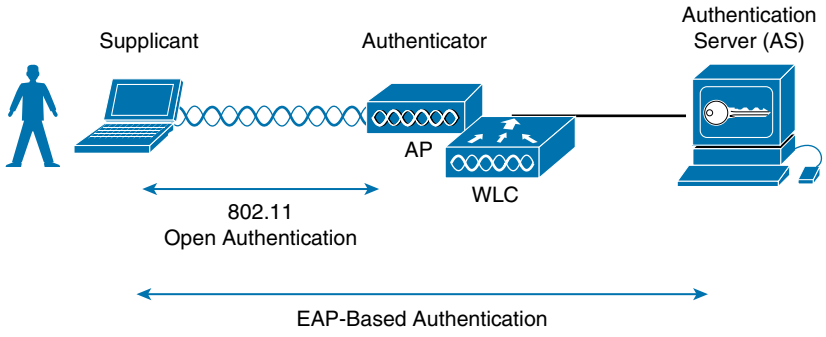


Figure 28-6 *802.1x Client Authentication Roles*

The wireless LAN controller becomes a middleman in the client authentication process, controlling user access with 802.1x and communicating with the authentication server using the EAP framework.

The following sections provide an overview of several common EAP-based authentication methods. The goal here is to become aware of the many methods without trying to memorize them all. In fact, even when you configure user authentication on a wireless LAN, you will not have to select a specific method. Instead, you select 802.1x on the WLC so that it is ready to handle a variety of EAP methods. It is then up to the client and the authentication server to use a compatible method. You will learn more about configuring security on a wireless LAN in Chapter 29, "Building a Wireless LAN."

LEAP

As an early attempt to address the weaknesses in WEP, Cisco developed a proprietary wireless authentication method called Lightweight EAP (LEAP). To authenticate, the client must supply username and password credentials. Both the authentication server and the client exchange challenge messages that are then encrypted and returned. This provides mutual authentication; as long as the messages can be decrypted successfully, the client and the AS have essentially authenticated each other.

At the time, WEP-based hardware was still widely used. Therefore, LEAP attempted to overcome WEP weaknesses by using dynamic WEP keys that changed frequently. Nevertheless, the method used to encrypt the challenge messages was found to be vulnerable, so LEAP has since been deprecated. Even though wireless clients and controllers still offer LEAP, you should not use it.

EAP-FAST

Cisco developed a more secure method called EAP Flexible Authentication by Secure Tunneling (EAP-FAST). Authentication credentials are protected by passing a protected access credential (PAC) between the AS and the supplicant. The PAC is a form of shared secret that is generated by the AS and used for mutual authentication. EAP-FAST is a sequence of three phases:

- **Phase 0:** The PAC is generated or provisioned and installed on the client.
- **Phase 1:** After the supplicant and AS have authenticated each other, they negotiate a Transport Layer Security (TLS) tunnel.
- **Phase 2:** The end user can then be authenticated through the TLS tunnel for additional security.

Notice that two separate authentication processes occur in EAP-FAST—one between the AS and the supplicant and another with the end user. These occur in a nested fashion, as an outer authentication (outside the TLS tunnel) and an inner authentication (inside the TLS tunnel).

Like other EAP-based methods, a RADIUS server is required. However, the RADIUS server must also operate as an EAP-FAST server to be able to generate PACs, one per user.

PEAP

Like EAP-FAST, the Protected EAP (PEAP) method uses an inner and outer authentication; however, the AS presents a digital certificate to authenticate itself with the supplicant in the outer authentication. If the supplicant is satisfied with the identity of the AS, the two will build a TLS tunnel to be used for the inner client authentication and encryption key exchange.

The digital certificate of the AS consists of data in a standard format that identifies the owner and is "signed" or validated by a third party. The third party is known as a certificate authority (CA) and is known and trusted by both the AS and the supplicants. The supplicant must also possess the CA certificate just so that it can validate the one it receives from the AS. The certificate is also used to pass a public key, in plain view, which can be used to help decrypt messages from the AS.

Notice that only the AS has a certificate for PEAP. That means the supplicant can readily authenticate the AS. The client does not have or use a certificate of its own, so it must be authenticated within the TLS tunnel using one of the following two methods:

- **MSCHAPv2:** Microsoft Challenge Authentication Protocol version 2
- **GTC:** Generic Token Card; a hardware device that generates one-time passwords for the user or a manually generated password

EAP-TLS

PEAP leverages a digital certificate on the AS as a robust method to authenticate the RADIUS server. It is easy to obtain and install a certificate on a single server, but the clients are left to identify themselves through other means. EAP Transport Layer Security (EAP-TLS) goes one step further by requiring certificates on the AS and on every client device.

With EAP-TLS, the AS and the supplicant exchange certificates and can authenticate each other. A TLS tunnel is built afterward so that encryption key material can be securely exchanged.

EAP-TLS is considered to be the most secure wireless authentication method available; however, implementing it can sometimes be complex. Along with the AS, each wireless client must obtain and install a certificate. Manually installing certificates on hundreds or thousands of clients can be impractical. Instead, you would need to implement a Public Key Infrastructure (PKI) that could supply certificates securely and efficiently and revoke them when a client or user should no longer have access to the network. This usually involves setting up your own CA or building a trust relationship with a third-party CA that can supply certificates to your clients.

NOTE EAP-TLS is practical only if the wireless clients can accept and use digital certificates. Many wireless devices, such as communicators, medical devices, and RFID tags, have an underlying operating system that cannot interface with a CA or use certificates.

Wireless Privacy and Integrity Methods

The original 802.11 standard supported only one method to secure wireless data from eavesdroppers: WEP. As you have learned in this chapter, WEP has been compromised, deprecated, and can no longer be recommended. What other options are available to encrypt data and protect its integrity as it travels through free space?

TKIP

During the time when WEP was embedded in wireless client and AP hardware, yet was known to be vulnerable, the Temporal Key Integrity Protocol (TKIP) was developed.

TKIP adds the following security features using legacy hardware and the underlying WEP encryption:

- **MIC:** This efficient algorithm adds a hash value to each frame as a message integrity check to prevent tampering; commonly called “Michael” as an informal reference to MIC.

- **Time stamp:** A time stamp is added into the MIC to prevent replay attacks that attempt to reuse or replay frames that have already been sent.
- **Sender's MAC address:** The MIC also includes the sender's MAC address as evidence of the frame source.
- **TKIP sequence counter:** This feature provides a record of frames sent by a unique MAC address, to prevent frames from being replayed as an attack.
- **Key mixing algorithm:** This algorithm computes a unique 128-bit WEP key for each frame.
- **Longer initialization vector (IV):** The IV size is doubled from 24 to 48 bits, making it virtually impossible to exhaust all WEP keys by brute-force calculation.

TKIP became a reasonably secure stopgap security method, buying time until the 802.11i standard could be ratified. Some attacks have been created against TKIP, so it, too, should be avoided if a better method is available. In fact, TKIP was deprecated in the 802.11-2012 standard.

CCMP

The Counter/CBC-MAC Protocol (CCMP) is considered to be more secure than TKIP. CCMP consists of two algorithms:

- AES counter mode encryption
- Cipher Block Chaining Message Authentication Code (CBC-MAC) used as a message integrity check (MIC)

The Advanced Encryption Standard (AES) is the current encryption algorithm adopted by U.S. National Institute of Standards and Technology (NIST) and the U.S. government, and widely used around the world. In other words, AES is open, publicly accessible, and represents the most secure encryption method available today.

Before CCMP can be used to secure a wireless network, the client devices and APs must support the AES counter mode and CBC-MAC in hardware. CCMP cannot be used on legacy devices that support only WEP or TKIP. How can you know if a device supports CCMP? Look for the WPA2 designation, which is described in the following section.

GCMP

The Galois/Counter Mode Protocol (GCMP) is a robust authenticated encryption suite that is more secure and more efficient than CCMP. GCMP consists of two algorithms:

- AES counter mode encryption
- Galois Message Authentication Code (GMAC) used as a message integrity check (MIC)

GCMP is used in WPA3, which is described in the following section.

WPA, WPA2, and WPA3

This chapter covers a variety of authentication methods and encryption and message integrity algorithms. When it comes time to configure a WLAN with wireless security,

should you try to select some combination of schemes based on which one is best or which one is not deprecated? Which authentication methods are compatible with which encryption algorithms?

The Wi-Fi Alliance (http://wi-fi.org), a nonprofit wireless industry association, has worked out straightforward ways to do that through its Wi-Fi Protected Access (WPA) industry certifications. To date, there are three different versions: WPA, WPA2, and WPA3. Wireless products are tested in authorized testing labs against stringent criteria that represent correct implementation of a standard. As long as the Wi-Fi Alliance has certified a wireless client device and an AP and its associated WLC for the same WPA version, they should be compatible and offer the same security components.

The Wi-Fi Alliance introduced its first generation WPA certification (known simply as WPA and not WPA1) while the IEEE 802.11i amendment for best practice security methods was still being developed. WPA was based on parts of 802.11i and included 802.1x authentication, TKIP, and a method for dynamic encryption key management.

Once 802.11i was ratified and published, the Wi-Fi Alliance included it in full in its WPA Version 2 (WPA2) certification. WPA2 is based around the superior AES CCMP algorithms, rather than the deprecated TKIP from WPA. It should be obvious that WPA2 was meant as a replacement for WPA.

In 2018, the Wi-Fi Alliance introduced WPA Version 3 (WPA3) as a future replacement for WPA2, adding several important and superior security mechanisms. WPA3 leverages stronger encryption by AES with the Galois/Counter Mode Protocol (GCMP). It also uses Protected Management Frames (PMF) to secure important 802.11 management frames between APs and clients, to prevent malicious activity that might spoof or tamper with a BSS's operation.

Table 28-2 summarizes the basic differences between WPA, WPA2, and WPA3. Each successive version is meant to replace prior versions by offering better security features. You should avoid using WPA and use WPA2 instead—at least until WPA3 becomes widely available on wireless client devices, APs, and WLCs.

Key Topic

Table 28-2 Comparing WPA, WPA2, and WPA3

Authentication and Encryption Feature Support	WPA	WPA2	WPA3*
Authentication with **Pre-Shared Keys?**	Yes	Yes	Yes
Authentication with **802.1x?**	Yes	Yes	Yes
Encryption and MIC with **TKIP?**	Yes	No	No
Encryption and MIC with **AES and CCMP?**	Yes	Yes	No
Encryption and MIC with **AES and GCMP?**	No	No	Yes

* WPA3 includes other features beyond WPA and WPA2, such as Simultaneous Authentication of Equals (SAE), Forward secrecy, and Protected management frames (PMF).

Notice that all three WPA versions support two client authentication modes: a pre-shared key (PSK) or 802.1x, based on the scale of the deployment. These are also known as

personal mode and *enterprise mode*, respectively. With personal mode, a key string must be shared or configured on every client and AP before the clients can connect to the wireless network. The pre-shared key is normally kept confidential so that unauthorized users have no knowledge of it. The key string is never sent over the air. Instead, clients and APs work through a four-way handshake procedure that uses the pre-shared key string to construct and exchange encryption key material that can be openly exchanged. Once that process is successful, the AP can authenticate the client and the two can secure data frames that are sent over the air.

With WPA-Personal and WPA2-Personal modes, a malicious user can eavesdrop and capture the four-way handshake between a client and an AP. That user can then use a dictionary attack to automate guessing the pre-shared key. If he is successful, he can then decrypt the wireless data or even join the network posing as a legitimate user.

WPA3-Personal avoids such an attack by strengthening the key exchange between clients and APs through a method known as Simultaneous Authentication of Equals (SAE). Rather than a client authenticating against a server or AP, the client and AP can initiate the authentication process equally and even simultaneously.

Even if a password or key is compromised, WPA3-Personal offers forward secrecy, which prevents attackers from being able to use a key to unencrypt data that has already been transmitted over the air.

NOTE The Personal mode of any WPA version is usually easy to deploy in a small environment or with clients that are embedded in certain devices because a simple text key string is all that is needed to authenticate the clients. Be aware that every device using the WLAN must be configured with an identical pre-shared key. If you ever need to update or change the key, you must touch every device to do so. As well, the pre-shared key should remain a well kept secret; you should never divulge the pre-shared key to any unauthorized person.

Notice from Table 28-2 that WPA, WPA2, and WPA3 also support 802.1x or enterprise authentication. This implies EAP-based authentication, but the WPA versions do not require any specific EAP method. Instead, the Wi-Fi Alliance certifies interoperability with well-known EAP methods like EAP-TLS, PEAP, EAP-TTLS, and EAP-SIM. Enterprise authentication is more complex to deploy than personal mode because authentication servers must be set up and configured as a critical enterprise resource.

NOTE The Wi-Fi Alliance has made wireless security configuration straightforward and consistent through its WPA, WPA2, and WPA3 certifications. Each version is meant to replace its predecessors because of improved security mechanisms. You should always select the highest WPA version that the clients and wireless infrastructure in your environment will support.

Chapter Review

At this point in the chapter, you might still be a little overwhelmed with the number of acronyms and security terms to learn and keep straight in your mind. Spend some time reviewing Table 28-3, which lists all of the topics described in this chapter. The table is organized in a way that should help you remember how the acronyms and functions are grouped together. Remember that an effective wireless security strategy includes a method to authenticate clients and a method to provide data privacy and integrity. These two types of methods are listed in the leftmost column. Work your way to the right to remember what types of authentication and privacy/integrity are available. The table also expands the name of each acronym as a memory tool.

Also remember that WPA, WPA2, and WPA3 simplify wireless network configuration and compatibility because they limit which authentication and privacy/integrity methods can be used.

Key Topic

Table 28-3 Review of Wireless Security Mechanisms and Options

Security Mechanism	Type		Type Expansion	Credentials Used
Authentication Methods	Open		Open Authentication	None, other than 802.11 protocol
	WEP		Wired Equivalent Privacy	Static WEP keys
	802.1x/EAP (Extensible Authentication Protocol)	LEAP	Lightweight EAP	Deprecated; uses dynamic WEP keys
		EAP-FAST	EAP Flexible Authentication by Secure Tunneling	Uses protected access credential (PAC)
		PEAP	Protected EAP	AS authenticated by digital certificate
		EAP-TLS	EAP Transport Layer Security	Client and AS authenticated by digital certificate
Privacy & Integrity Methods	TKIP		Temporal Key Integrity Protocol	N/A
	CCMP		Counter/CBC-MAC Protocol	N/A
	GCMP		Galois/Counter Mode Protocol	N/A

You should also review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 28-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 28-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Review the most important topics in this chapter, noted with the Key Topic icon in the outer margin of the page. Table 28-5 lists a reference of these key topics and the page numbers on which each is found.

Key Topic

Table 28-5 Key Topics for Chapter 28

Key Topic Element	Description	Page Number
List	802.1x entities	658
Table 28-2	WPA, WPA2, and WPA3 comparison	662
Table 28-3	Wireless security mechanism review	664

Key Terms You Should Know

802.1x, authentication server (AS), authenticator, certificate authority (CA), Counter/CBC-MAC Protocol (CCMP), EAP Flexible Authentication by Secure Tunneling (EAP-FAST), EAP Transport Layer Security (EAP-TLS), enterprise mode, Extensible Authentication Protocol (EAP), forward secrecy, Galois/Counter Mode Protocol (GCMP), Lightweight EAP (LEAP), message integrity check (MIC), open authentication, personal mode, protected access credential (PAC), Protected EAP (PEAP), Protected Management Frame (PMF), Public Key Infrastructure (PKI), RADIUS server, Simultaneous Authentication of Equals (SAE), supplicant, Temporal Key Integrity Protocol (TKIP), Wired Equivalent Privacy (WEP), Wi-Fi Protected Access (WPA), WPA Version 2 (WPA2), WPA Version 3 (WPA3)

CHAPTER 29

Building a Wireless LAN

This chapter covers the following exam topics:

2.0 Network Access

2.7 Describe physical infrastructure connections of WLAN components (AP, WLC, access/trunk ports, and LAG)

2.8 Describe AP and WLC management access connections (Telnet, SSH, HTTP, HTTPS, console, and TACACS+/RADIUS)

2.9 Configure the components of a wireless LAN access for client connectivity using GUI only, such as WLAN creation, security settings, QoS profiles, and advanced WLAN settings

5.0 Security Fundamentals

5.10 Configure WLAN using WPA2 PSK using the GUI

In Chapters 26 through 28, you learned about the fundamentals of wireless networks. As a CCNA, you will also need to know how to apply that knowledge toward building a functioning network with APs and a WLC.

In addition, based on the concepts you learned in Chapter 28, "Securing Wireless Networks," you will be able to configure the WLAN to use WPA2-Personal.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 29-1 "Do I Know This Already?" Section-to-Question Mapping

Foundation Topics Section	Questions
Connecting a Cisco AP	1–2
Accessing a Cisco WLC	3
Connecting a Cisco WLC	4–5
Configuring a WLAN	6–8

1. Suppose you need to connect a lightweight AP to a network. Which one of the following link types would be necessary?
   a. Access mode link
   b. Trunk mode link
   c. LAG mode link
   d. EtherChannel link
2. An autonomous AP will be configured to support three WLANs that correspond to three VLANs. The AP will connect to the network over which one of the following?
   a. Access mode link
   b. Trunk mode link
   c. LAG mode link
   d. EtherChannel link
3. Suppose you would like to connect to a WLC to configure a new WLAN on it. Which one of the following is a valid method to use?
   a. SSH
   b. HTTPS
   c. HTTP
   d. All of these answers are correct.
4. Which one of the following correctly describes the single logical link formed by bundling all of a controller's distribution system ports together?
   a. PHY
   b. DSP
   c. LAG
   d. GEC
5. Which one of the following controller interfaces maps a WLAN to a VLAN?
   a. Bridge interface
   b. Virtual interface
   c. WLAN interface
   d. Dynamic interface
6. Which two of the following things are bound together when a new WLAN is created?
   a. VLAN
   b. AP
   c. Controller interface
   d. SSID

7. What is the maximum number of WLANs you can configure on a Cisco wireless controller?
   a. 8
   b. 16
   c. 512
   d. 1024
8. Which of the following parameters are necessary when creating a new WLAN with the controller GUI? (Choose all that apply.)
   a. SSID
   b. VLAN number
   c. Interface
   d. BSSID
   e. IP subnet

Foundation Topics

Connecting a Cisco AP

A Cisco wireless network can consist of autonomous APs or lightweight APs that are coupled with one or more wireless LAN controllers. Both types of APs are covered in Chapter 27, "Analyzing Cisco Wireless Architectures," from a functional perspective. You should also understand how to connect the wired side of each type of AP so that it can pass traffic between the appropriate VLANs and WLANs.

Recall that an autonomous AP is a standalone device; nothing else is needed to forward Ethernet frames from a wired VLAN to a wireless LAN, and vice versa. In effect, the AP maps each VLAN to a WLAN and BSS. The autonomous AP has a single wired Ethernet interface, as shown in the left portion of Figure 29-1, which means that multiple VLANs must be brought to it over a trunk link.

NOTE A switch port providing a wired connection to an AP must be configured to support either access or trunk mode. In trunk mode, 802.1Q encapsulation tags each frame according to the VLAN number it came from. The wireless side of an AP inherently trunks 802.11 frames by marking them with the BSSID of the WLAN where they belong.

A lightweight AP also has a single wired Ethernet interface; however, it must be paired with a WLC to be fully functional. Wired VLANs that terminate at the WLC can be mapped to WLANs that emerge at the AP. Even though multiple VLANs are being extended from the WLC to the AP, they are all carried over the CAPWAP tunnel between the two. That means the AP needs only an access link to connect to the network infrastructure and terminate its end of the tunnel, as shown in the right portion of Figure 29-1.

Key Topic

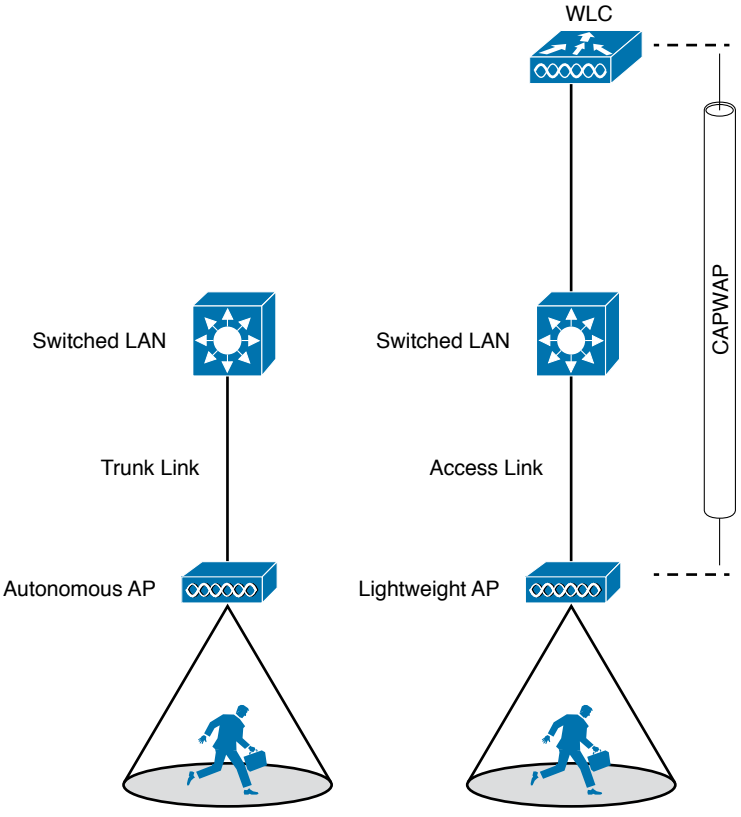


Figure 29-1 *Comparing Connections to Autonomous and Lightweight APs*

To configure and manage Cisco APs, you can connect a serial console cable from your PC to the console port on the AP. Once the AP is operational and has an IP address, you can also use Telnet or SSH to connect to its CLI over the wired network. Autonomous APs support browser-based management sessions via HTTP and HTTPS. You can manage lightweight APs from a browser session to the WLC.

Accessing a Cisco WLC

To connect and configure a WLC, you will need to open a web browser to the WLC's management address with either HTTP or HTTPS. This can be done only after the WLC has an initial configuration and a management IP address assigned to its management interface. The web-based GUI provides an effective way to monitor, configure, and troubleshoot a wireless network. You can also connect to a WLC with an SSH session, where you can use its CLI to monitor, configure, and debug activity.

Both the web-based GUI and the CLI require management users to log in. Users can be authenticated against an internal list of local usernames or against an authentication, authorization, and accounting (AAA) server, such as TACACS+ or RADIUS.

When you first open a web browser to the management address, you will see the initial login screen. Click on the **Login** button, as shown in Figure 29-2; then enter your user credentials as you are prompted for them.

Figure 29-2 *Accessing a WLC with a Web Browser*

NOTE The CCNA exam objectives focus on using the WLC GUI to configure a WLAN and a security suite. Therefore, the examples in this section assume that someone has already entered an initial configuration to give the WLC a working IP address for management.

When you are successfully logged in, the WLC will display a monitoring dashboard similar to the one shown in Figure 29-3. You will not be able to make any configuration changes there, so you must click on the **Advanced** link in the upper-right corner. This will bring up the full WLC GUI, as shown in Figure 29-4.

Notice the tabs across the top of the screen in Figure 29-4. You can select categories of functions from among Monitor, WLANs, Controller, Wireless, Security, and so on. As you select one of these categories, the vertical list of functions at the left side of the screen will change accordingly. You can expand the list entries if needed and select one to work on. The main screen area will display all of the relevant fields and options you can edit as you make configuration changes. You will get a feel for which tabs and list items you should use as you work through the remainder of the chapter.

Answers to the "Do I Know This Already?" quiz:

1 A **2** B **3** D **4** C **5** D **6** C, D **7** C **8** A, C

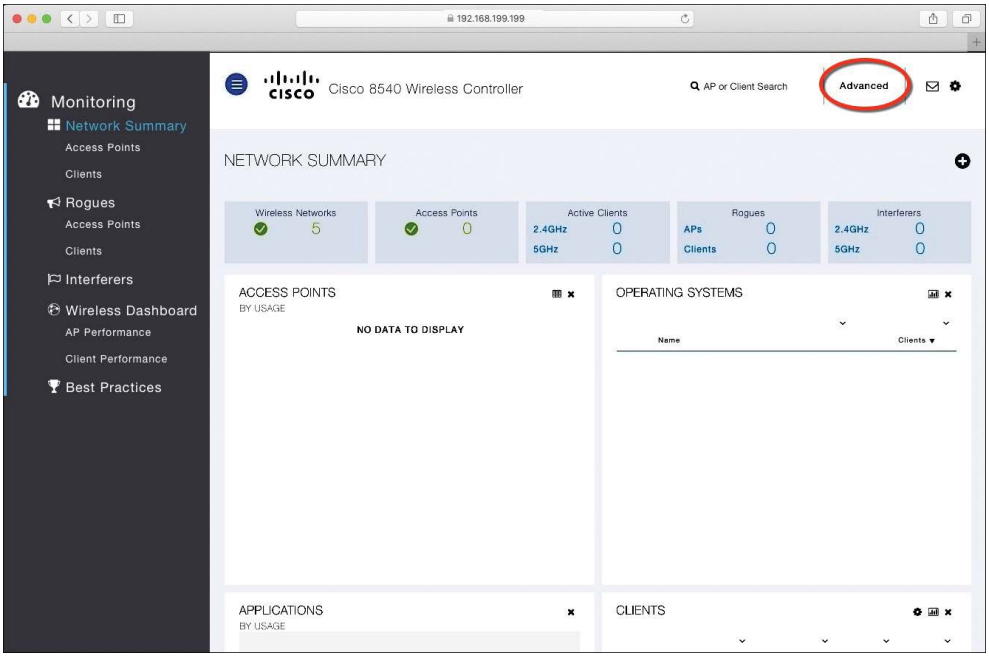


Figure 29-3 *Accessing the Advanced Configuration Interface*

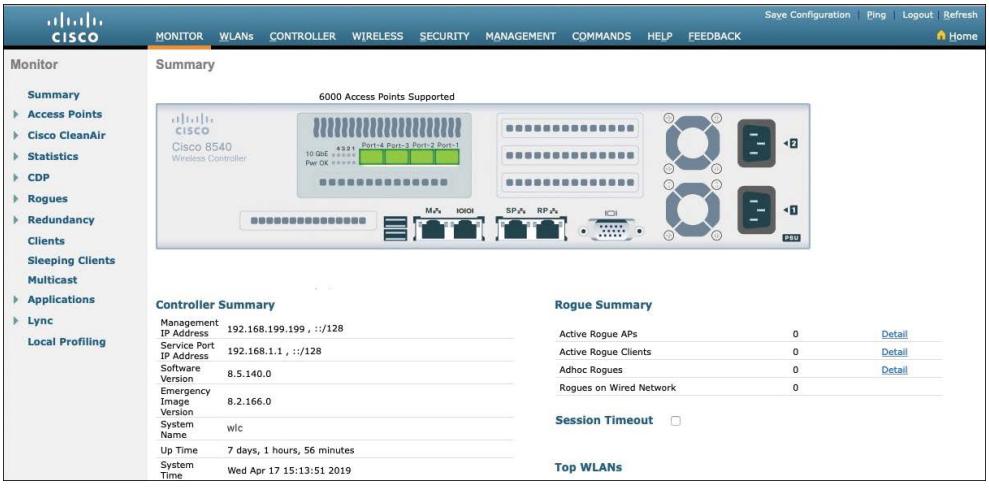


Figure 29-4 *The Advanced WLC Configuration GUI*

Connecting a Cisco WLC

Connecting a Cisco wireless LAN controller to the network is not quite as straightforward because it has several different types of connections. From your work with Cisco routers and switches, you probably know that the terms *interface* and *port* are usually interchangeable. For example, switches can come in 48-port models, and you apply configuration changes to the corresponding interfaces. Cisco wireless controllers differ a bit; ports and interfaces refer to different concepts.

Controller ports are physical connections made to an external wired or switched network, whereas interfaces are logical connections made internally within the controller. The following sections explain each connection type in more detail. You will learn more about configuring ports and interfaces in the "Configuring a WLAN" section later in the chapter.

Using WLC Ports

You can connect several different types of controller ports to your network, as shown in Figure 29-5 and discussed in the following list:

- **Service port:** Used for out-of-band management, system recovery, and initial boot functions; always connects to a switch port in access mode
- **Distribution system port:** Used for all normal AP and management traffic; usually connects to a switch port in 802.1Q trunk mode
- **Console port:** Used for out-of-band management, system recovery, and initial boot functions; asynchronous connection to a terminal emulator (9600 baud, 8 data bits, 1 stop bit, by default)
- **Redundancy port:** Used to connect to a peer controller for high availability (HA) operation

Key Topic

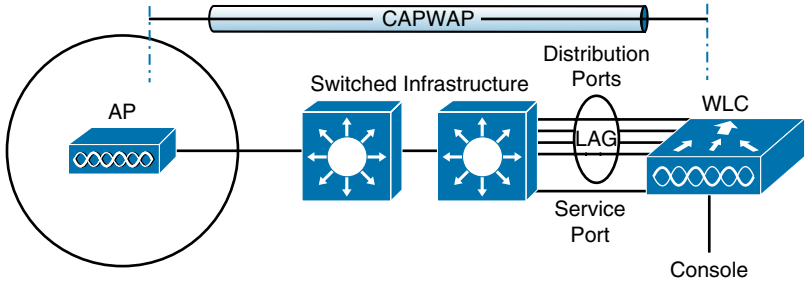


Figure 29-5 *Cisco Wireless LAN Controller Ports*

Controllers can have a single service port that must be connected to a switched network. Usually, the service port is assigned to a management VLAN so that you can access the controller with SSH or a web browser to perform initial configuration or for maintenance. Notice that the service port supports only a single VLAN, so the corresponding switch port must be configured for access mode only.

Controllers also have multiple distribution system ports that you must connect to the network. These ports carry most of the data coming to and going from the controller. For example, the CAPWAP tunnels (control and data) that extend to each of a controller's APs pass across the distribution system ports. Client data also passes from wireless LANs to wired VLANs over the ports. In addition, any management traffic using a web browser, SSH, Simple Network Management Protocol (SNMP), Trivial File Transfer Protocol (TFTP), and so on, normally reaches the controller in-band through the ports.

NOTE You might be thinking that *distribution system ports* is an odd name for what appear to be regular data ports. Recall from the section titled "Wireless LAN Topologies" in Chapter 26, "Fundamentals of Wireless Networks," that the wired network that connects APs together is called the distribution system (DS). With the split MAC architecture, the point where APs touch the DS is moved upstream to the WLC instead.

Because the distribution system ports must carry data that is associated with many different VLANs, VLAN tags and numbers become very important. For that reason, the distribution system ports always operate in 802.1Q trunking mode. When you connect the ports to a switch, you should also configure the switch ports for unconditional 802.1Q trunk mode.

The distribution system ports can operate independently, each one transporting multiple VLANs to a unique group of internal controller interfaces. For resiliency, you can configure distribution system ports in redundant pairs. One port is primarily used; if it fails, a backup port is used instead.

To get the most use out of each distribution system port, you can configure all of them to operate as a single logical group, much like an EtherChannel or port-channel on a switch. Controller distribution system ports can be configured as a link aggregation group (LAG) such that they are bundled together to act as one larger link. In Figure 29-5, the four distribution system ports are configured as a LAG. With a LAG configuration, traffic can be load-balanced across the individual ports that make up the LAG. In addition, LAG offers resiliency; if one individual port fails, traffic will be redirected to the remaining working ports instead.

NOTE Be aware that even though the LAG acts as a traditional EtherChannel, Cisco WLCs do not support any link aggregation negotiation protocol, like LACP or PAgP, at all. Therefore, you must configure the switch ports as an unconditional or always-on EtherChannel.

Using WLC Interfaces

Through its distribution system ports, a controller can connect to multiple VLANs on the switched network. Internally, the controller must somehow map those wired VLANs to equivalent logical wireless networks. For example, suppose that VLAN 10 is set aside for wireless users in the Engineering division of a company. That VLAN must be connected to a unique wireless LAN that exists on a controller and its associated APs. The wireless LAN must then be extended to every client that associates with the Service Set Identifier (SSID) "Engineering."

Cisco wireless controllers provide the necessary connectivity through internal logical interfaces, which must be configured with an IP address, subnet mask, default gateway, and a Dynamic Host Configuration Protocol (DHCP) server. Each interface is then assigned to a physical port and a VLAN ID. You can think of an interface as a Layer 3 termination on a VLAN.

Cisco controllers support the following interface types, also shown in Figure 29-6.

- **Management interface:** Used for normal management traffic, such as RADIUS user authentication, WLC-to-WLC communication, web-based and SSH sessions, SNMP, Network Time Protocol (NTP), syslog, and so on. The management interface is also used to terminate CAPWAP tunnels between the controller and its APs.
- **Redundancy management:** The management IP address of a redundant WLC that is part of a high availability pair of controllers. The active WLC uses the management interface address, while the standby WLC uses the redundancy management address.
- **Virtual interface:** IP address facing wireless clients when the controller is relaying client DHCP requests, performing client web authentication, and supporting client mobility.
- **Service port interface:** Bound to the service port and used for out-of-band management.
- **Dynamic interface:** Used to connect a VLAN to a WLAN.

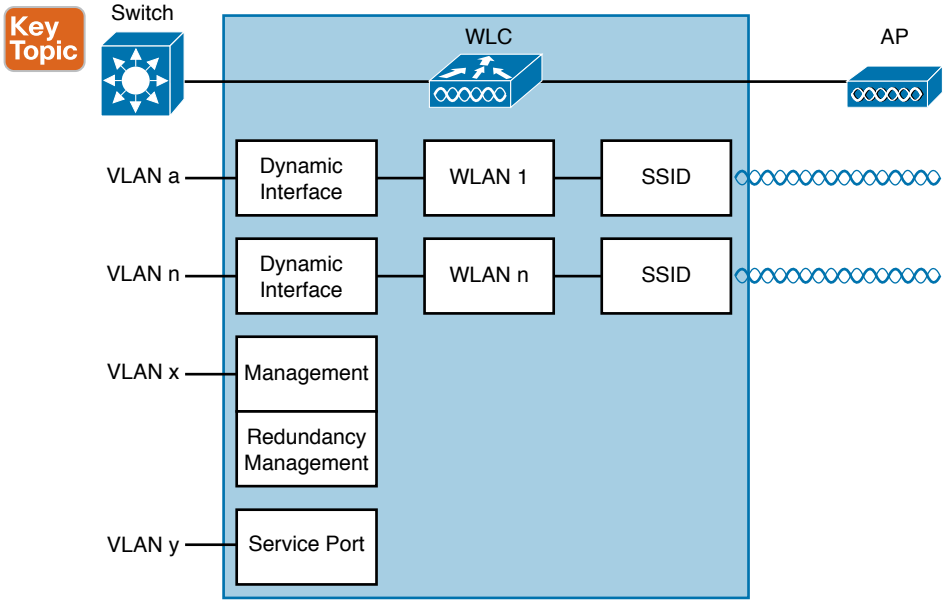


Figure 29-6 *Cisco Wireless LAN Controller Interfaces*

The management interface faces the switched network, where management users and APs are located. Management traffic will usually consist of protocols like HTTPS, SSH, SNMP, NTP, TFTP, and so on. In addition, management interface traffic consists of CAPWAP packets that carry control and data tunnels to and from the APs.

The virtual interface is used only for certain client-facing operations. For example, when a wireless client issues a request to obtain an IP address, the controller can relay the request on to an actual DHCP server that can provide the appropriate IP address. From the client's perspective, the DHCP server appears to be the controller's virtual interface address. Clients may see the virtual interface's address, but that address is never used when the controller communicates with other devices on the switched network.

Because the virtual interface is used only for some client management functions, you should configure it with a unique, nonroutable address. For example, you might use 10.1.1.1 because it is within a private address space defined in RFC 1918.

NOTE Traditionally, many people have assigned IP address 1.1.1.1 to the virtual interface. Although it is a unique address, it is routable and already in use elsewhere on the Internet. A better practice is to use an IP address from the RFC 1918 private address space that is unused or reserved, such as 192.168.1.1. You could also use a reserved address from RFC 5737 (192.0.2.0/24) that is set aside for documentation purposes and is never used.

The virtual interface address is also used to support client mobility. For that reason, every controller that exists in the same mobility group should be configured with a virtual address that is identical to the others. By using one common virtual address, all the controllers will appear to operate as a cluster as clients roam from controller to controller.

Dynamic interfaces map WLANs to VLANs, making the logical connections between wireless and wired networks. You will configure one dynamic interface for each wireless LAN that is offered by the controller's APs and then map the interface to the WLAN. Each dynamic interface must also be configured with its own IP address and can act as a DHCP relay for wireless clients. To filter traffic passing through a dynamic interface, you can configure an optional access list.

Configuring a WLAN

A wireless LAN controller and an access point work in concert to provide network connectivity to wireless clients. From a wireless perspective, the AP advertises a Service Set Identifier (SSID) for the client to join. From a wired perspective, the controller connects to a virtual LAN (VLAN) through one of its dynamic interfaces. To complete the path between the SSID and the VLAN, as illustrated in Figure 29-7, you must first define a WLAN on the controller.

NOTE Two of the CCNA exam objectives involve configuring a WLAN for client connectivity with WPA2 and a PSK using only the controller GUI. As you work through this section, you will find that it presents a complete WLAN example that is based on the topology shown in Figure 29-7 using the WPA2-Personal (PSK) security model.

Key Topic

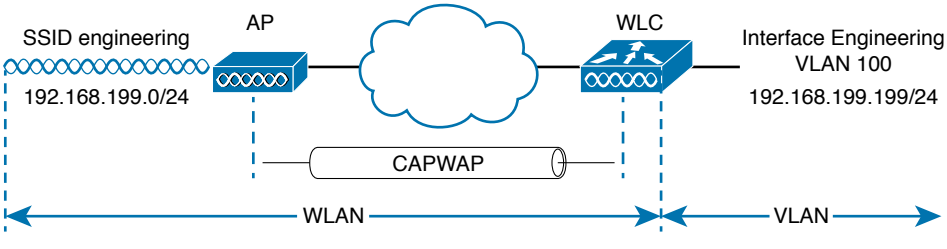


Figure 29-7 *Connecting Wired and Wireless Networks with a WLAN*

The controller will bind the WLAN to one of its interfaces and then push the WLAN configuration out to all of its APs by default. From that point on, wireless clients will be able to learn about the new WLAN by receiving its beacons and will be able to probe and join the new BSS.

Like VLANs, you can use WLANs to segregate wireless users and their traffic into logical networks. Users associated with one WLAN cannot cross over into another one unless their traffic is bridged or routed from one VLAN to another through the wired network infrastructure.

Before you begin to create new WLANs, it is usually wise to plan your wireless network first. In a large enterprise, you might have to support a wide variety of wireless devices, user communities, security policies, and so on. You might be tempted to create a new WLAN for every occasion, just to keep groups of users isolated from each other or to support different types of devices. Although that is an appealing strategy, you should be aware of two limitations:

- Cisco controllers support a maximum of 512 WLANs, but only 16 of them can be actively configured on an AP.
- Advertising each WLAN to potential wireless clients uses up valuable airtime.

Every AP must broadcast beacon management frames at regular intervals to advertise the existence of a BSS. Because each WLAN is bound to a BSS, each WLAN must be advertised with its own beacons. Beacons are normally sent 10 times per second, or once every 100 ms, at the lowest mandatory data rate. The more WLANs you have created, the more beacons you will need to announce them.

Even further, the lower the mandatory data rate, the more time each beacon will take to be transmitted. The end result is this: if you create too many WLANs, a channel can be starved of any usable airtime. Clients will have a hard time transmitting their own data because the channel is overly busy with beacon transmissions coming from the AP. As a rule of thumb, always limit the number of WLANs to five or fewer; a maximum of three WLANs is best.

By default, a controller has a limited initial configuration, so no WLANs are defined. Before you create a new WLAN, think about the following parameters it will need to have:

- SSID string
- Controller interface and VLAN number
- Type of wireless security needed

As you work through this section, you will create the appropriate dynamic controller interface to support the new WLAN; then you will enter the necessary WLAN parameters. Each configuration step is performed using a web browser session that is connected to the WLC's management IP address.

Step 1. Configure a RADIUS Server

If your new WLAN will use a security scheme that requires a RADIUS server, such as WPA2-Enterprise or WPA3-Enterprise, you will need to define the server first. Select **Security > AAA > RADIUS > Authentication** to see a list of servers that have already been configured, as shown in Figure 29-8. If multiple servers are defined, the controller will try them in sequential order. Click **New** to create a new server.

Next, enter the server's IP address, shared secret key, and port number, as shown in Figure 29-9. Because the controller already had two other RADIUS servers configured, the server at 192.168.200.30 will be index number 3. Be sure to set the server status to **Enabled** so that the controller can begin using it. At the bottom of the page, you can select the type of user that will be authenticated with the server. Check **Network User** to authenticate wireless clients or **Management** to authenticate wireless administrators that will access the controller's management functions. Click **Apply** to complete the server configuration.

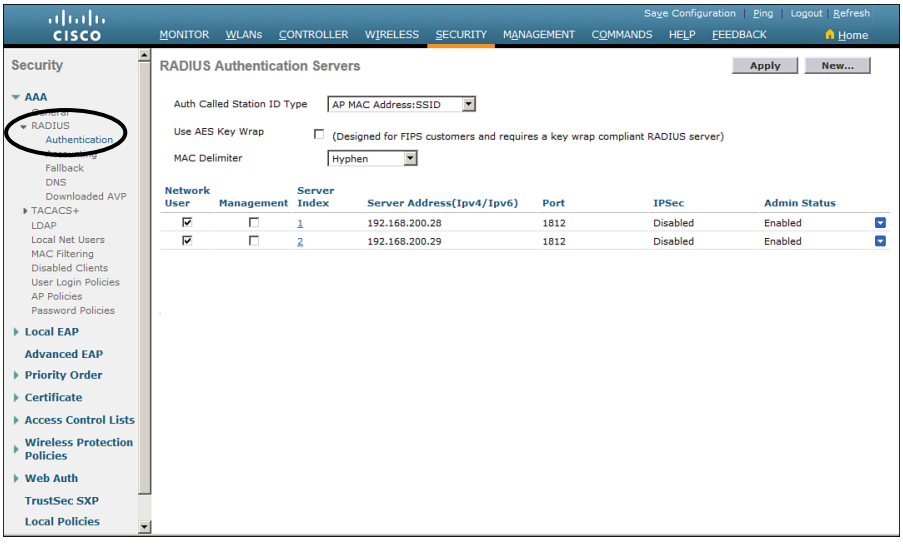


Figure 29-8 *Displaying the List of RADIUS Authentication Servers*

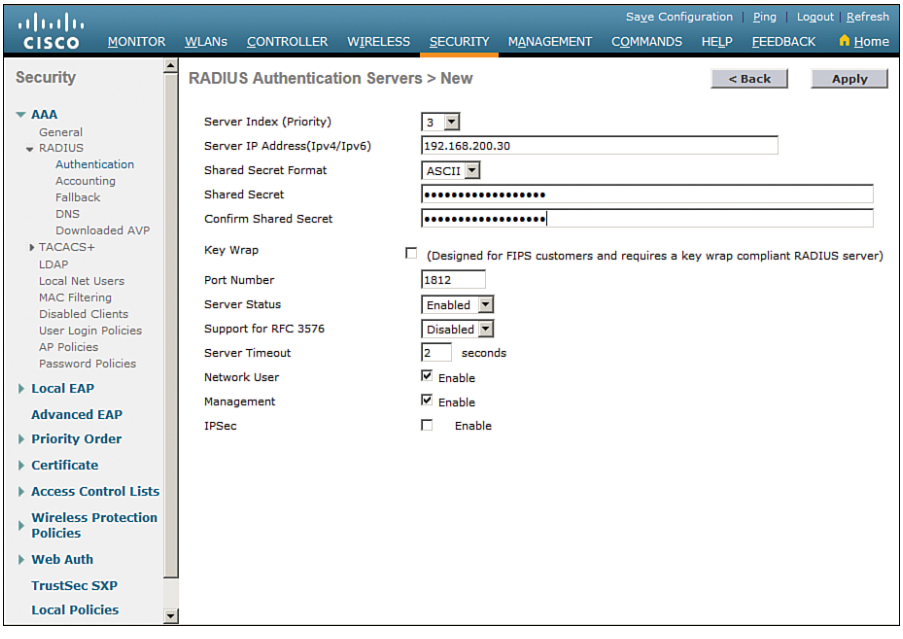


Figure 29-9 *Configuring a New RADIUS Server*

Step 2. Create a Dynamic Interface

In the "Using WLC Interfaces" section of this chapter, you learned about the different types of controller interfaces. A dynamic interface is used to connect the controller to a VLAN on the wired network. When you create a WLAN, you will bind the dynamic interface (and VLAN) to a wireless network.

To create a new dynamic interface, navigate to **Controller > Interfaces**. You should see a list of all the controller interfaces that are currently configured. In Figure 29-10, two interfaces named "management" and "virtual" already exist. Click the **New** button to define a new interface. Enter a name for the interface and the VLAN number it will be bound to. In Figure 29-11, the interface named Engineering is mapped to wired VLAN 100. Click the **Apply** button.

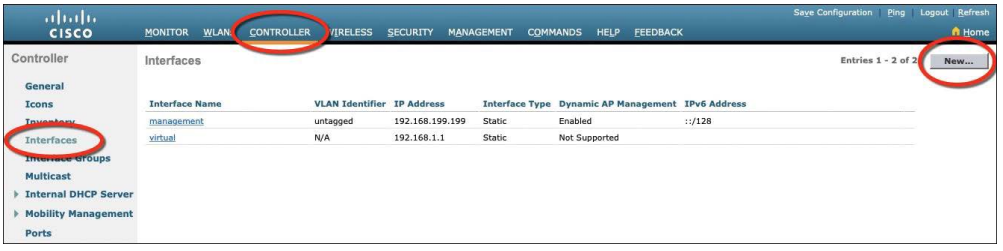


Figure 29-10 *Displaying a List of Dynamic Interfaces*

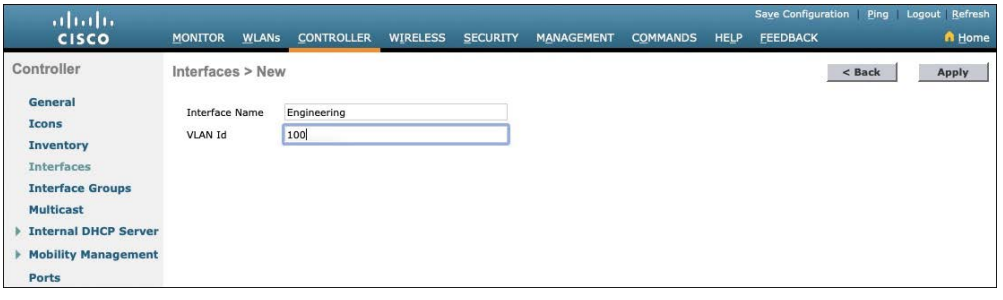


Figure 29-11 *Defining a Dynamic Interface Name and VLAN ID*

Next, enter the IP address, subnet mask, and gateway address for the interface. You should also define primary and secondary DHCP server addresses that the controller will use when it relays DHCP requests from clients that are bound to the interface. Figure 29-12 shows how the interface named Engineering has been configured with IP address 192.168.100.10, subnet mask 255.255.255.0, gateway 192.168.100.1, and DHCP servers 192.168.1.17 and 192.168.1.18. Click the **Apply** button to complete the interface configuration and return to the list of interfaces.

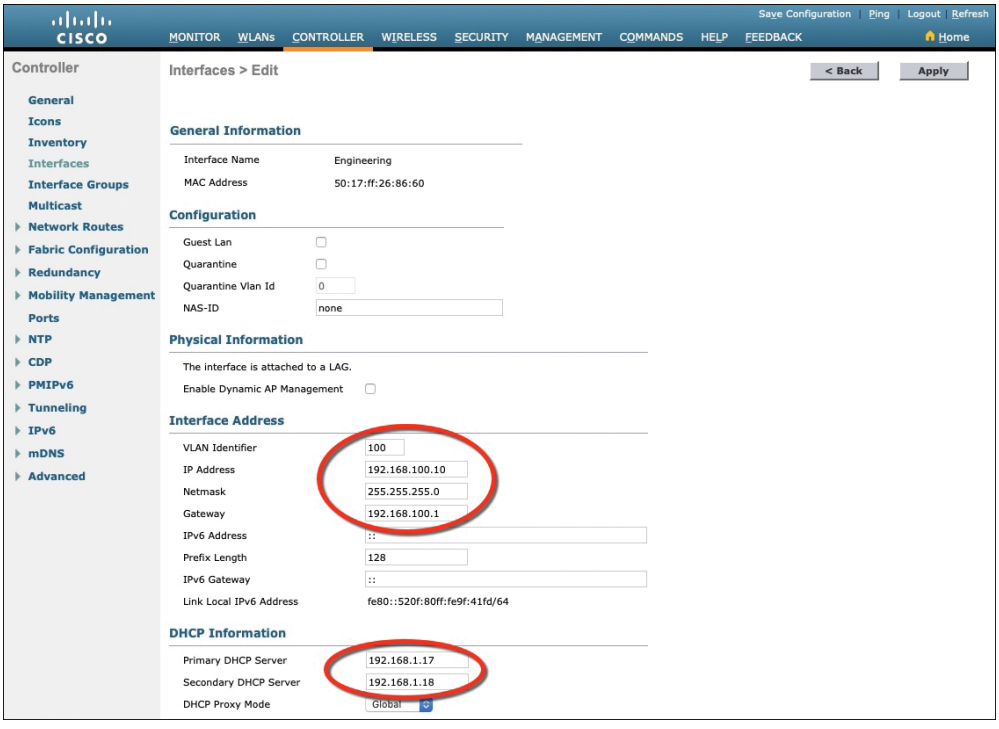


Figure 29-12 *Editing the Dynamic Interface Parameters*

Step 3. Create a New WLAN

You can display a list of the currently defined WLANs by selecting **WLANs** from the top menu bar. In Figure 29-13, the controller does not have any WLANs already defined. You can create a new WLAN by selecting **Create New** from the drop-down menu and then clicking the **Go** button.

Figure 29-13 *Displaying a List of WLANs*

Next, enter a descriptive name as the profile name and the SSID text string. In Figure 29-14, the profile name and SSID are identical, just to keep things straightforward. The ID number is used as an index into the list of WLANs that are defined on the controller. The ID number becomes useful when you use templates in Prime Infrastructure (PI) to configure WLANs on multiple controllers at the same time.

NOTE WLAN templates are applied to specific WLAN ID numbers on controllers. The WLAN ID is only locally significant and is not passed between controllers. As a rule, you should keep the sequence of WLAN names and IDs consistent across multiple controllers so that any configuration templates you use in the future will be applied to the same WLANs on each controller.

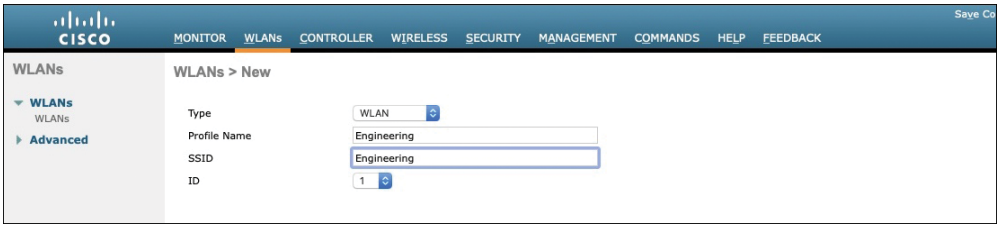


Figure 29-14 *Creating a New WLAN*

Click the **Apply** button to create the new WLAN. The next page will allow you to edit four categories of parameters, corresponding to the tabs across the top as shown in Figure 29-15. By default, the General tab is selected.

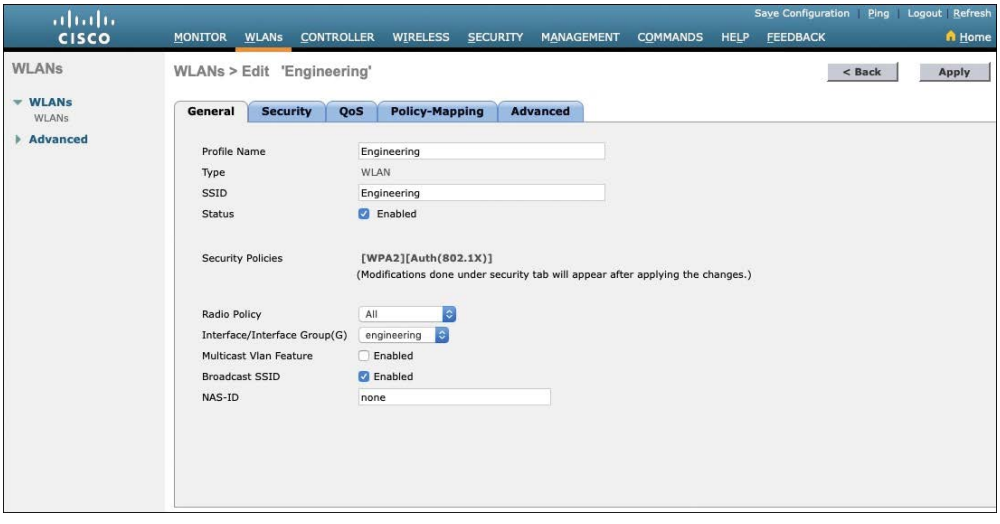


Figure 29-15 *Configuring the General WLAN Parameters*

You can control whether the WLAN is enabled or disabled with the Status check box. Even though the General page shows a specific security policy for the WLAN (the default WPA2 with 802.1x), you can make changes in a later step through the Security tab.

Under Radio Policy, select the type of radio that will offer the WLAN. By default, the WLAN will be offered on all radios that are joined with the controller. You can select a more specific policy with 802.11a only, 802.11a/g only, 802.11g only, or 802.11b/g only. For example, if you are creating a new WLAN for devices that have only a 2.4-GHz radio, it probably does not make sense to advertise the WLAN on both 2.4- and 5-GHz AP radios.

Next, select which of the controller's dynamic interfaces will be bound to the WLAN. By default, the management interface is selected. The drop-down list contains all the interface names that are available. In Figure 29-15, the new engineering WLAN will be bound to the Engineering interface.

Finally, use the Broadcast SSID check box to select whether the APs should broadcast the SSID name in the beacons they transmit. Broadcasting SSIDs is usually more convenient for users because their devices can learn and display the SSID names automatically. In fact, most devices actually need the SSID in the beacons to understand that the AP is still available for that SSID. Hiding the SSID name, by not broadcasting it, does not really provide any worthwhile security. Instead, it just prevents user devices from discovering an SSID and trying to use it as a default network.

Configuring WLAN Security

Select the Security tab to configure the security settings. By default, the Layer 2 Security tab is selected. From the Layer 2 Security drop-down menu, select the appropriate security scheme to use. Table 29-2 lists the types that are available.

Key Topic

Table 29-2 Layer 2 WLAN Security Type

Option	Description
None	Open authentication
WPA+WPA2	Wi-Fi protected access WPA or WPA2
802.1x	EAP authentication with dynamic WEP
Static WEP	WEP key security
Static WEP + 802.1x	EAP authentication or static WEP
CKIP	Cisco Key Integrity Protocol
None + EAP Passthrough	Open authentication with remote EAP authentication

As you select a security type, be sure to remember which choices are types that have been deprecated or proven to be weak, and avoid them if possible. Further down the screen, you can select which specific WPA, WPA2, and WPA3 methods to support on the WLAN. You can select more than one, if you need to support different types of wireless clients that require several security methods.

In Figure 29-16, WPA+WPA2 has been selected from the pull-down menu; then only WPA2 and AES encryption have been selected. WPA and TKIP have been avoided because they are legacy, deprecated methods. Under the Authentication Key Management section, you can select the authentication methods the WLAN will use. Only PSK has been selected in the figure, so the WLAN will allow only WPA2-Personal with pre-shared key authentication.

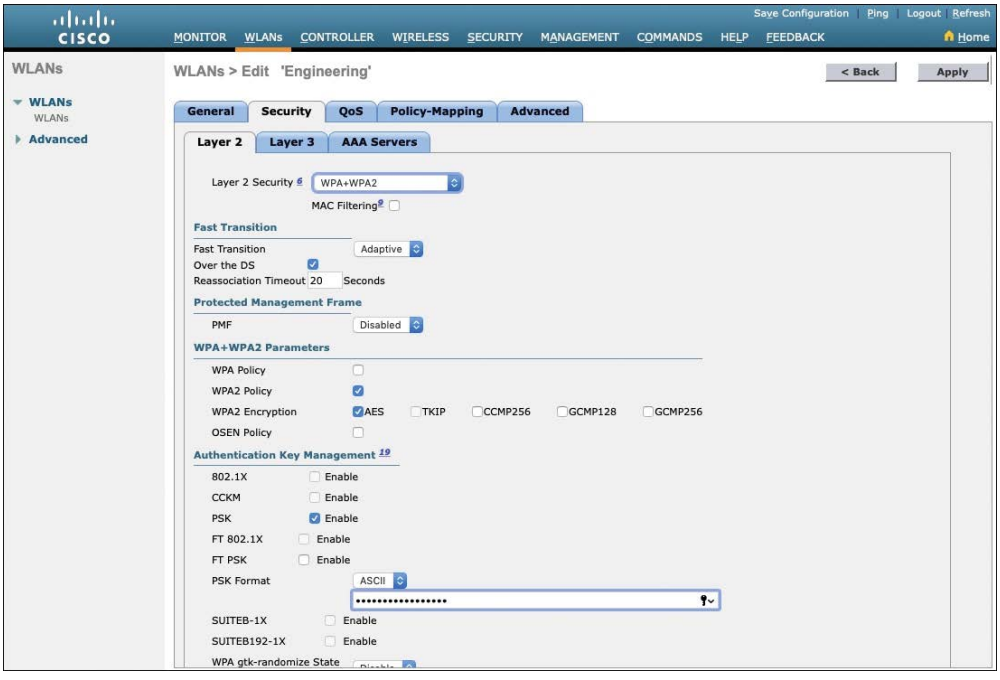


Figure 29-16 *Configuring Layer 2 WLAN Security*

To use WPA2-Enterprise, the 802.1X option would be selected. In that case, 802.1x and EAP would be used to authenticate wireless clients against one or more RADIUS servers. The controller would use servers from the global list you have defined under **Security > AAA > RADIUS > Authentication**, as described in the "Step 1. Configure a RADIUS Server" section in this chapter. To specify which servers the WLAN should use, you would select the Security tab and then the AAA Servers tab in the WLAN edit screen. You can identify up to six specific RADIUS servers in the WLAN configuration. Beside each server, select a specific server IP address from the drop-down menu of globally defined servers. The servers are tried in sequential order until one of them responds. Although the example in this chapter uses WPA2-Personal, Figure 29-17 shows what a WLAN configured for WPA2-Enterprise might look like, with servers 1 through 3 being set to 192.168.200.28, 192.168.200.29, and 192.168.200.30, respectively.

By default, a controller will contact a RADIUS server from its management interface. You can override this behavior by checking the box next to Radius Server Overwrite Interface so that the controller sources RADIUS requests from the dynamic interface that is associated with the WLAN.

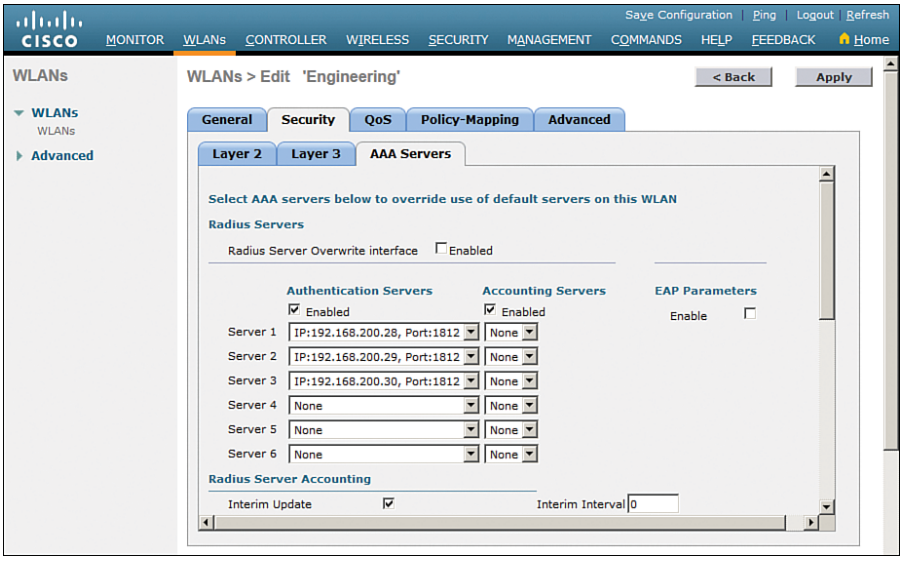


Figure 29-17 *Selecting RADIUS Servers for WLAN Authentication*

Configuring WLAN QoS

Select the **QoS** tab to configure quality of service settings for the WLAN, as shown in Figure 29-18. By default, the controller will consider all frames in the WLAN to be normal data, to be handled in a "best effort" manner. You can set the Quality of Service (QoS) drop-down menu to classify all frames in one of the following ways:

- Platinum (voice)
- Gold (video)
- Silver (best effort)
- Bronze (background)

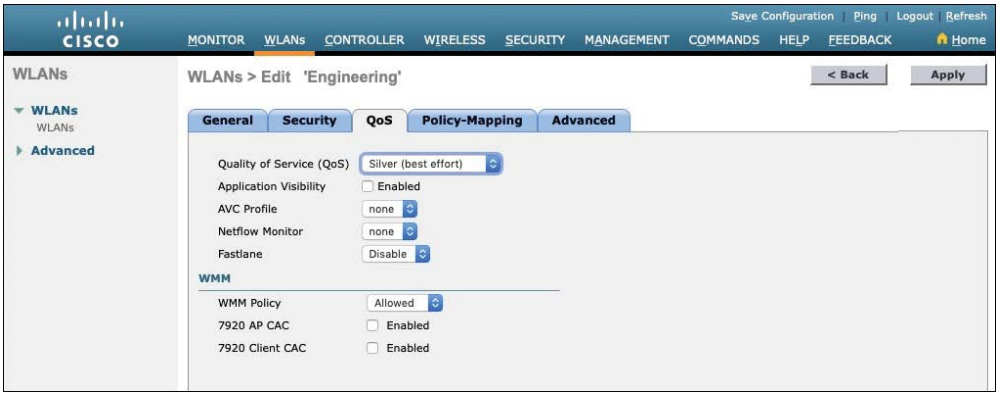


Figure 29-18 *Configuring QoS Settings*

You can also set the Wi-Fi Multimedia (WMM) policy, call admission control (CAC) policies, and bandwidth parameters on the QoS page. You can learn more about QoS in the *CCNA 200-301 Official Cert Guide, Volume 2*, in Chapter 11, "Quality of Service."

Configuring Advanced WLAN Settings

Finally, you can select the Advanced tab to configure a variety of advanced WLAN settings. From the page shown in Figure 29-19, you can enable functions such as coverage hole detection, peer-to-peer blocking, client exclusion, client load limits, and so on.

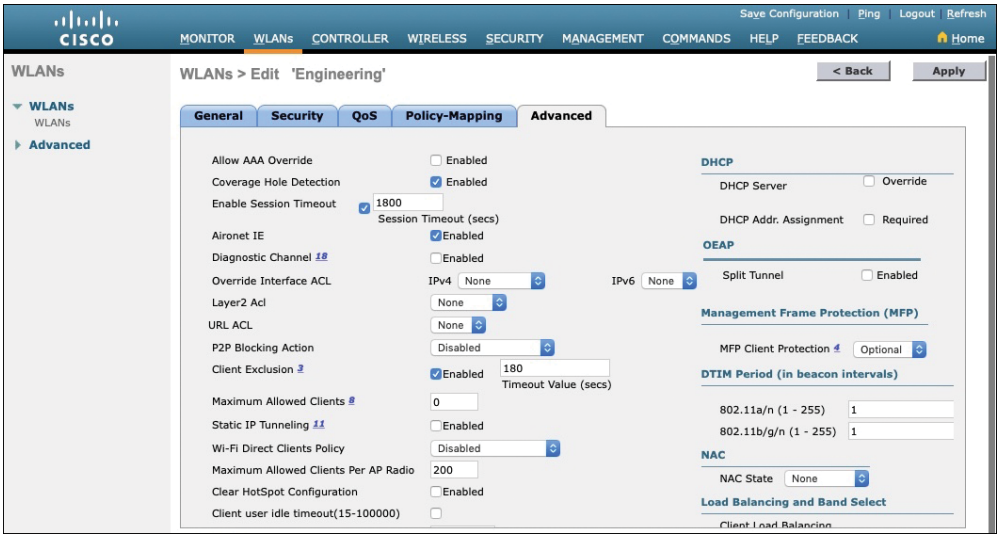


Figure 29-19 *Configuring Advanced WLAN Settings*

Although most of the advanced settings are beyond the scope of the CCNA objectives, you should be aware of a few defaults that might affect your wireless clients.

By default, client sessions with the WLAN are limited to 1800 seconds (30 minutes). Once that session time expires, a client will be required to reauthenticate. This setting is controlled by the Enable Session Timeout check box and the Timeout field.

A controller maintains a set of security policies that are used to detect potentially malicious wireless clients. If a client exhibits a certain behavior, the controller can exclude it from the WLAN for a period of time. By default, all clients are subject to the policies configured under **Security > Wireless Protection Policies > Client Exclusion Policies**. These policies include excessive 802.11 association failures, 802.11 authentication failures, 802.1x authentication failures, web authentication failures, and IP address theft or reuse. Offending clients will be automatically excluded or blocked for 60 seconds, as a deterrent to attacks on the wireless network.

NOTE Is 60 seconds really enough time to deter an attack coming from a wireless client? In the case of a brute-force attack, where passwords are guessed from a dictionary of possibilities, 60 seconds is enough to disrupt and delay an attacker's progress. What might have taken 2 minutes to find a matching password without an exclusion policy would take 15 years with one.

Finalizing WLAN Configuration

When you are satisfied with the settings in each of the WLAN configuration tabs, click the Apply button in the upper-right corner of the WLAN Edit screen. The WLAN will be created and added to the controller configuration. In Figure 29-20, the Engineering WLAN has been added as WLAN ID 1 and is enabled for use.

Figure 29-20 *Displaying WLANs Configured on a Controller*

Be aware that, by default, a controller will not allow management traffic that is initiated from a WLAN. That means you (or anybody else) cannot access the controller GUI or CLI from a wireless device that is associated to the WLAN. This is considered to be a good security practice because the controller is kept isolated from networks that might be easily accessible or where someone might eavesdrop on the management session traffic. Instead, you can access the controller through its wired interfaces.

You can change the default behavior on a global basis (all WLANs) by selecting the Management tab and then selecting Mgmt Via Wireless, as shown in Figure 29-21. Check the box to allow management sessions from any WLAN that is configured on the controller.

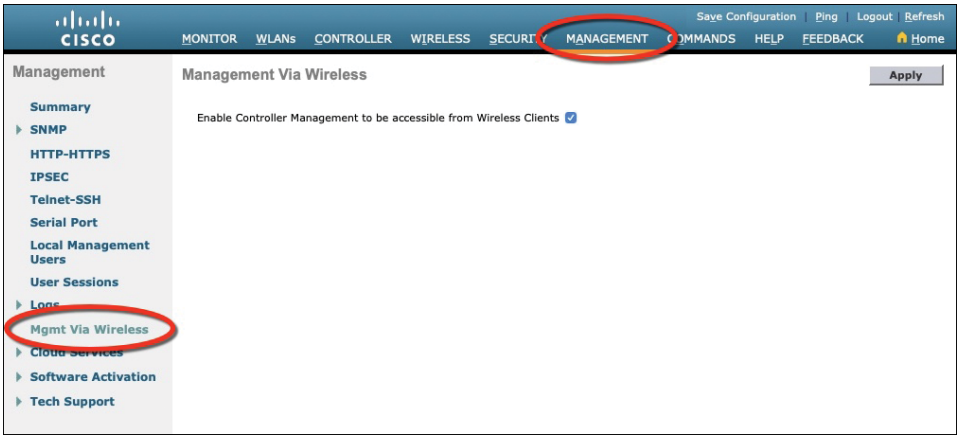


Figure 29-21 *Configuring Management Access from Wireless Networks*

Chapter Review

Review this chapter's material using either the tools in the book or the interactive tools for the same material found on the book's companion website. Table 29-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 29-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP

Review All the Key Topics

Review the most important topics in this chapter, noted with the Key Topic icon in the outer margin of the page. Table 29-4 lists a reference of these key topics and the page numbers on which each is found.

Key Topic

Table 29-4 Key Topics for Chapter 29

Key Topic Element	Description	Page Number
Figure 29-1	Physical connections to an AP	669
Figure 29-5	Wireless LAN controller ports	672
Figure 29-6	Wireless LAN controller interfaces	674
Figure 29-7	Creating a WLAN	675
Table 29-2	Configuring WLAN security	681

This page intentionally left blank

Part VIII Review

Keep track of your part review progress with the checklist in Table P8-1. Details on each task follow the table.

Table P8-1 Part VIII Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		

Repeat All DIKTA Questions

For this task, use the PCPT software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or using the Key Topics application on the companion website.

This page intentionally left blank

Part IX

Appendixes

Glossary

Appendix A Numeric Reference Tables

Appendix B Exam Updates

Appendix C Answers to the “Do I Know This Already?” Quizzes

This page intentionally left blank

APPENDIX A

Numeric Reference Tables

This appendix provides several useful reference tables that list numbers used throughout this book. Specifically:

Table A-1: A decimal-binary cross reference, useful when converting from decimal to binary and vice versa.

Table A-1 Decimal-Binary Cross Reference, Decimal Values 0–255

Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value
0	00000000	32	00100000	64	01000000	96	01100000
1	00000001	33	00100001	65	01000001	97	01100001
2	00000010	34	00100010	66	01000010	98	01100010
3	00000011	35	00100011	67	01000011	99	01100011
4	00000100	36	00100100	68	01000100	100	01100100
5	00000101	37	00100101	69	01000101	101	01100101
6	00000110	38	00100110	70	01000110	102	01100110
7	00000111	39	00100111	71	01000111	103	01100111
8	00001000	40	00101000	72	01001000	104	01101000
9	00001001	41	00101001	73	01001001	105	01101001
10	00001010	42	00101010	74	01001010	106	01101010
11	00001011	43	00101011	75	01001011	107	01101011
12	00001100	44	00101100	76	01001100	108	01101100
13	00001101	45	00101101	77	01001101	109	01101101
14	00001110	46	00101110	78	01001110	110	01101110
15	00001111	47	00101111	79	01001111	111	01101111
16	00010000	48	00110000	80	01010000	112	01110000
17	00010001	49	00110001	81	01010001	113	01110001
18	00010010	50	00110010	82	01010010	114	01110010
19	00010011	51	00110011	83	01010011	115	01110011
20	00010100	52	00110100	84	01010100	116	01110100
21	00010101	53	00110101	85	01010101	117	01110101
22	00010110	54	00110110	86	01010110	118	01110110
23	00010111	55	00110111	87	01010111	119	01110111
24	00011000	56	00111000	88	01011000	120	01111000
25	00011001	57	00111001	89	01011001	121	01111001
26	00011010	58	00111010	90	01011010	122	01111010
27	00011011	59	00111011	91	01011011	123	01111011
28	00011100	60	00111100	92	01011100	124	01111100
29	00011101	61	00111101	93	01011101	125	01111101
30	00011110	62	00111110	94	01011110	126	01111110
31	00011111	63	00111111	95	01011111	127	01111111

Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value
128	10000000	160	10100000	192	11000000	224	11100000
129	10000001	161	10100001	193	11000001	225	11100001
130	10000010	162	10100010	194	11000010	226	11100010
131	10000011	163	10100011	195	11000011	227	11100011
132	10000100	164	10100100	196	11000100	228	11100100
133	10000101	165	10100101	197	11000101	229	11100101
134	10000110	166	10100110	198	11000110	230	11100110
135	10000111	167	10100111	199	11000111	231	11100111
136	10001000	168	10101000	200	11001000	232	11101000
137	10001001	169	10101001	201	11001001	233	11101001
138	10001010	170	10101010	202	11001010	234	11101010
139	10001011	171	10101011	203	11001011	235	11101011
140	10001100	172	10101100	204	11001100	236	11101100
141	10001101	173	10101101	205	11001101	237	11101101
142	10001110	174	10101110	206	11001110	238	11101110
143	10001111	175	10101111	207	11001111	239	11101111
144	10010000	176	10110000	208	11010000	240	11110000
145	10010001	177	10110001	209	11010001	241	11110001
146	10010010	178	10110010	210	11010010	242	11110010
147	10010011	179	10110011	211	11010011	243	11110011
148	10010100	180	10110100	212	11010100	244	11110100
149	10010101	181	10110101	213	11010101	245	11110101
150	10010110	182	10110110	214	11010110	246	11110110
151	10010111	183	10110111	215	11010111	247	11110111
152	10011000	184	10111000	216	11011000	248	11111000
153	10011001	185	10111001	217	11011001	249	11111001
154	10011010	186	10111010	218	11011010	250	11111010
155	10011011	187	10111011	219	11011011	251	11111011
156	10011100	188	10111100	220	11011100	252	11111100
157	10011101	189	10111101	221	11011101	253	11111101
158	10011110	190	10111110	222	11011110	254	11111110
159	10011111	191	10111111	223	11011111	255	11111111

Table A-2: A hexadecimal-binary cross reference, useful when converting from hex to binary and vice versa.

Table A-2 Hex-Binary Cross Reference

Hex	4-Bit Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Table A-3: Powers of 2, from 2^1 through 2^{32}.

Table A-3 Powers of 2

X	2^X	X	2^X
1	2	17	131,072
2	4	18	262,144
3	8	19	524,288
4	16	20	1,048,576
5	32	21	2,097,152
6	64	22	4,194,304
7	128	23	8,388,608
8	256	24	16,777,216
9	512	25	33,554,432
10	1024	26	67,108,864
11	2048	27	134,217,728
12	4096	28	268,435,456
13	8192	29	536,870,912
14	16,384	30	1,073,741,824
15	32,768	31	2,147,483,648
16	65,536	32	4,294,967,296

Table A-4: Table of all 33 possible subnet masks, in all three formats.

Table A-4 All Subnet Masks

Decimal	Prefix	Binary
0.0.0.0	/0	00000000 00000000 00000000 00000000
128.0.0.0	/1	10000000 00000000 00000000 00000000
192.0.0.0	/2	11000000 00000000 00000000 00000000
224.0.0.0	/3	11100000 00000000 00000000 00000000
240.0.0.0	/4	11110000 00000000 00000000 00000000
248.0.0.0	/5	11111000 00000000 00000000 00000000
252.0.0.0	/6	11111100 00000000 00000000 00000000
254.0.0.0	/7	11111110 00000000 00000000 00000000
255.0.0.0	/8	11111111 00000000 00000000 00000000
255.128.0.0	/9	11111111 10000000 00000000 00000000
255.192.0.0	/10	11111111 11000000 00000000 00000000
255.224.0.0	/11	11111111 11100000 00000000 00000000
255.240.0.0	/12	11111111 11110000 00000000 00000000
255.248.0.0	/13	11111111 11111000 00000000 00000000
255.252.0.0	/14	11111111 11111100 00000000 00000000
255.254.0.0	/15	11111111 11111110 00000000 00000000
255.255.0.0	/16	11111111 11111111 00000000 00000000
255.255.128.0	/17	11111111 11111111 10000000 00000000
255.255.192.0	/18	11111111 11111111 11000000 00000000
255.255.224.0	/19	11111111 11111111 11100000 00000000
255.255.240.0	/20	11111111 11111111 11110000 00000000
255.255.248.0	/21	11111111 11111111 11111000 00000000
255.255.252.0	/22	11111111 11111111 11111100 00000000
255.255.254.0	/23	11111111 11111111 11111110 00000000
255.255.255.0	/24	11111111 11111111 11111111 00000000
255.255.255.128	/25	11111111 11111111 11111111 10000000
255.255.255.192	/26	11111111 11111111 11111111 11000000
255.255.255.224	/27	11111111 11111111 11111111 11100000
255.255.255.240	/28	11111111 11111111 11111111 11110000
255.255.255.248	/29	11111111 11111111 11111111 11111000
255.255.255.252	/30	11111111 11111111 11111111 11111100
255.255.255.254	/31	11111111 11111111 11111111 11111110
255.255.255.255	/32	11111111 11111111 11111111 11111111

APPENDIX B

CCNA 200-301, Volume 1 Exam Updates

Over time, reader feedback allows Pearson to gauge which topics give our readers the most problems when taking the exams. To assist readers with those topics, the authors create new materials clarifying and expanding on those troublesome exam topics. As mentioned in the Introduction, the additional content about the exam is contained in a PDF on this book's companion website, at http://www.ciscopress.com/title/9780135792735.

This appendix provides you with updated information if Cisco makes minor modifications to the exam topics during the life of the 200-301 exam. In particular, this appendix does the following:

- Mentions technical items that might not have been mentioned elsewhere in the book
- Covers new topics if Cisco adds new content to the exam over time
- Provides a way to get up-to-the-minute current information about content for the exam

Note that this appendix shows updated information related to the subset of CCNA 200-301 exam topics covered in this book. Refer also to the *CCNA 200-301 Official Cert Guide, Volume 2*, for more details about the rest of the exam topics and for an Appendix B similar to that of this book.

Always Get the Latest at the Book's Product Page

Many of you are reading the version of this appendix that was available when your book was printed or when you downloaded the e-book. However, given that the main purpose of this appendix is to be a living, changing document, it is important that you look for the latest version online at the book's companion website. To do so, follow these steps:

Step 1. Browse to www.ciscopress.com/title/9780135792735.

Step 2. Click the **Updates** tab.

Step 3. If there is a new Appendix B document on the page, download the latest Appendix B document.

NOTE The downloaded document has a version number. Comparing the version of the print Appendix B (**Version 1.0**) with the latest downloadable version of this appendix, you should do the following:

- **Same version:** Ignore the PDF that you downloaded from the companion website.
- **Website has a later version:** Ignore this Appendix B in your book and read only the latest version that you downloaded from the companion website.

Technical Content

The current **Version 1.0** of this appendix does not contain additional technical coverage.

APPENDIX C

Answers to the "Do I Know This Already?" Quizzes

Chapter 1

1. D and F. Of the remaining answers, Ethernet defines both physical and data-link protocols, PPP is a data-link protocol, IP is a network layer protocol, and SMTP and HTTP are application layer protocols.
2. A and G. Of the remaining answers, IP is a network layer protocol, TCP and UDP are transport layer protocols, and SMTP and HTTP are application layer protocols.
3. B. Adjacent-layer interaction occurs on one computer, with two adjacent layers in the model. The higher layer requests services from the next lower layer, and the lower layer provides the services to the next higher layer.
4. B. Same-layer interaction occurs on multiple computers. The functions defined by that layer typically need to be accomplished by multiple computers—for example, the sender setting a sequence number for a segment and the receiver acknowledging receipt of that segment. A single layer defines that process, but the implementation of that layer on multiple devices is required to accomplish the function.
5. A. Encapsulation is defined as the process of adding a header in front of data supplied by a higher layer (and possibly adding a trailer as well).
6. D. By convention, the term *frame* refers to the part of a network message that includes the data-link header and trailer, with encapsulated data. The term *packet* omits the data-link header and trailer, leaving the network layer header with its encapsulated data. The term *segment* omits the network layer header, leaving the transport layer header and its encapsulated data.
7. B. The term frame refers to the data-link (that is, Layer 2) data structure created by a Layer 2 protocol. As a result, the matching OSI term for protocol data units (PDUs) mentions that same layer, that is, Layer 2 PDU, or L2PDU.

Chapter 2

1. A. The IEEE defines Ethernet LAN standards, with standard names that begin with 802.3, all of which happen to use cabling. The IEEE also defines wireless LAN standards, with standard names that begin with 802.11, which are separate standards from Ethernet.
2. C. The number before the word *BASE* defines the speed, in megabits per second (Mbps). 1000 Mbps equals 1 gigabit per second (1 Gbps). The *T* in the suffix implies twisted-pair or UTP cabling, so 1000BASE-T is the UTP-based Gigabit Ethernet standard name.

3. B. Crossover cables cross the wire at one node's transmit pin pair to the different pins used as the receive pins on the other device. For 10- and 100-Mbps Ethernet, the specific crossover cable wiring connects the pair at pins 1 and 2 on each end of the cable to pins 3 and 6 on the other end of the cable, respectively.
4. B, D, and E. Routers, wireless access point Ethernet ports, and PC NICs all send using pins 1 and 2, whereas hubs and LAN switches transmit on pins 3 and 6. Straight-through cables connect devices that use opposite pin pairs for sending, because the cable does not need to cross the pairs.
5. B. Multimode fiber works with LED-based transmitters rather than laser-based transmitters. Two answers mention the type of transmitters, making one of those answers correct and one incorrect.

   Two answers mention distance. The answer that mentions the longest distance possible is incorrect because single-mode cables, not multimode cables, provide the longest distances. The other (correct) answer mentions the tradeoff of multimode being used for distances just longer than UTP's 100 meter limit, while happening to use less expensive hardware than single mode.
6. B. NICs (and switch ports) use the carrier sense multiple access with collision detection (CSMA/CD) algorithm to implement half-duplex logic. CSMA/CD attempts to avoid collisions, but it also notices when collisions do occur, with rules about how the Ethernet nodes should stop sending, wait, and try again later.
7. C. The 4-byte Ethernet FCS field, found in the Ethernet trailer, allows the receiving node to see what the sending node computed with a math formula that is a key part of the error-detection process. Note that Ethernet defines the process of detecting errors (error detection), but not error recovery.
8. B, C, and E. The pre-assigned universal MAC address, given to each Ethernet port when manufactured, breaks the address into two 3-byte halves. The first half is called the organizationally unique identifier (OUI), which the IEEE assigns to the company that builds the product as a unique hex number to be used only by that company.
9. C and D. Ethernet supports unicast addresses, which identify a single Ethernet node, and group addresses, which can be used to send one frame to multiple Ethernet nodes. The two types of group addresses are the *broadcast address* and *multicast address*.

Chapter 3

1. B. The standard HDLC header does not include a Type field, which identifies the type of packet encapsulated inside the HDLC frame.
2. B and D. The physical installation uses a model in which each router uses a physical Ethernet link to connect to some SP device in an SP facility called a point of presence (PoP). The Ethernet link does not span from each customer device to the other. From a data-link perspective, both routers use the same Ethernet standard header and trailer used on LANs; HDLC does not matter on these Ethernet WAN links.
3. A. PC1 will send an Ethernet frame to Router 1, with PC1's MAC address as the source address and Router 1's MAC address as the destination address. Router 1 will remove the encapsulated IP packet from that Ethernet frame, discarding the frame header and

trailer. Router 1 will forward the IP packet by first encapsulating it inside an HDLC frame, but Router 1 will not encapsulate the Ethernet frame in the HDLC frame but rather the IP packet. Router 2 will de-encapsulate the IP packet from the HDLC frame and forward it onto the Ethernet LAN, adding a new Ethernet header and trailer, but this header will differ. It will list Router 2's MAC address as the source address and PC2's MAC address as the destination address.

4. C. Routers compare the packet's destination IP address to the router's IP routing table, making a match and using the forwarding instructions in the matched route to forward the IP packet.
5. C. IPv4 hosts generally use basic two-branch logic. To send an IP packet to another host on the same IP network or subnet that is on the same LAN, the sender sends the IP packet directly to that host. Otherwise, the sender sends the packet to its default router (also called the default gateway).
6. A and C. Routers do all the actions listed in all four answers; however, the routing protocol does the functions in the two listed answers. Independent of the routing protocol, a router learns routes for IP subnets and IP networks directly connected to its interfaces. Routers also forward (route) IP packets, but that process is called IP routing, or IP forwarding, and is an independent process compared to the work of a routing protocol.
7. C. Address Resolution Protocol (ARP) does allow PC1 to learn information, but the information is not stored on a server. The **ping** command does let the user at PC1 learn whether packets can flow in the network, but it again does not use a server. With the Domain Name System (DNS), PC1 acts as a DNS client, relying on a DNS server to respond with information about the IP addresses that match a given hostname.

Chapter 4

1. A and B. The command in the question is an EXEC command that happens to require only user mode access. As such, you can use this command in both user mode and enable mode. Because it is an EXEC command, you cannot use the command (as shown in the question) in configuration mode. Note that you can put the word **do** in front of the EXEC command while in configuration mode (for example, **do show mac address-table**) to issue the command from inside any configuration mode.
2. B. The command referenced in the question, the **reload** command, is an EXEC command that happens to require privileged mode, also known as enable mode. This command is not available in user mode. Note that you can put the word **do** in front of the EXEC command while in configuration mode (for example, **do reload**) to issue the command from inside any configuration mode.
3. B. SSH provides a secure remote login option, encrypting all data flows, including password exchanges. Telnet sends all data (including passwords) as clear text.
4. A. Switches (and routers) keep the currently used configuration in RAM, using NVRAM to store the configuration file that is loaded when the switch (or router) next loads the IOS.
5. F. The startup-config file is in NVRAM, and the running-config file is in RAM.

6. B and C. The **exit** command moves the user one config mode backward, toward global configuration mode, or if already in global configuration mode, it moves the user back to enable mode. From console mode, it moves the user back to global configuration mode. The **end** command and the Ctrl+Z key sequence both move the user back to enable mode regardless of the current configuration submode.

Chapter 5

1. A. A switch compares the destination MAC address to the MAC address table. If a matching entry is found, the switch forwards the frame out the appropriate interface. If no matching entry is found, the switch floods the frame.
2. C. A switch floods broadcast frames, multicast frames (if no multicast optimizations are enabled), and unknown unicast destination frames (frames whose destination MAC address is not in the MAC address table).
3. A. A switch floods broadcast frames, multicast frames (if no multicast optimizations are enabled), and unknown unicast destination frames (frames whose destination MAC address is not in the MAC address table).
4. B. Switches need to learn the location of each MAC address used in the LAN relative to that local switch. When a switch receives a frame, the source MAC identifies the sender. The interface in which the frame arrives identifies the local switch interface closest to that node in the LAN topology.
5. C. The **show interfaces status** command lists one line of output per interface. Cisco Catalyst switches name the type of interface based on the fastest speed of the interface, so 10/100 interfaces would be Fast Ethernet. With a working connection, ports from FastEthernet 0/1 through 0/10 would be listed in a connected state, while the rest would be listed in a notconnected state.
6. D. For the correct answer, each entry lists the learned MAC address. By definition, dynamically learned MAC addresses are learned by looking at the source MAC address of received frames. (That fact rules out one of the incorrect answers as well.)

   The **show mac address-table dynamic** command lists the current list of MAC table entries, with three known entries at the point at which the command output was gathered. The counter in the last line of output lists the number of current entries, not the total number of learned MAC addresses since the last reboot. For instance, the switch could have learned other MAC addresses whose entries timed out from the MAC address table.

   Finally, the answer that claims that port Gi0/2 connects directly to a device with a particular MAC address may or may not be true. That port could connect to another switch, and another, and so on, with one of those switches connecting to the device that uses the listed MAC address.

Chapter 6

1. B. If both commands are configured, IOS accepts only the password as configured in the **enable secret** command.

2. A. To answer this question, it might be best to first think of the complete configuration and then find any answers that match the configuration. The commands, in vty line configuration mode, would be **password** *password* and **login.** Only one answer lists a vty subcommand that is one of these two commands.

   Of note in the incorrect answers:

   One answer mentions console subcommands. The console does not define what happens when remote users log in; those details sit in the vty line configuration.

   One answer mentions the **login local** command; this command means that the switch should use the local list of configured usernames/passwords. The question stated that the engineer wanted to use passwords only, with no usernames.

   One answer mentions the **transport input ssh** command, which, by omitting the **telnet** keyword, disables Telnet. While that command can be useful, SSH does not work when using passwords only; SSH requires both a username and a password. So, by disabling Telnet (and allowing SSH only), the configuration would allow no one to remotely log in to the switch.

3. B and C. SSH requires the use of usernames in addition to a password. Using the **username** global command would be one way to define usernames (and matching passwords) to support SSH. The vty lines would also need to be configured to require the use of usernames, with the **login local** vty subcommand being one such option.

   The **transport input ssh** command could be part of a meaningful configuration, but it is not a global configuration command (as claimed in one wrong answer). Likewise, one answer refers to the **username** command as a command in vty config mode, which is also the wrong mode.

4. A, D, and F. To allow access through Telnet, the switch must have password security enabled, at a minimum using the **password** vty line configuration subcommand. In addition, the switch needs an IP address (configured under one VLAN interface) and a default gateway when the switch needs to communicate with hosts in a different subnet.

5. B and C. To allow SSH or Telnet access, a switch must have a correct IP configuration. That includes the configuration of a correct IP address and mask on a VLAN interface. That VLAN interface then must have a path out of the switch via ports assigned to that VLAN. In this case, with all ports assigned to VLAN 2, the switch must use interface VLAN 2 (using the **interface vlan 2** configuration command).

   To meet the requirement to support login from hosts outside the local subnet, the switch must configure a correct default gateway setting with the **ip default-gateway 172.16.2.254** global command in this case.

6. A. The **logging synchronous** line subcommand synchronizes the log message display with other command output so the log message does not interrupt a **show** command's output. The **no ip domain-lookup** command is not a line subcommand. The other two incorrect answers are line subcommands but do not configure the function listed in the question.

Chapter 7

1. F. Cisco switches do not have a command to disable autonegotiation of speed and duplex. Instead, a switch port that has both **speed** and **duplex** configured disables autonegotiation.
2. E. Cisco switches can be configured for speed (with the **speed** command) and duplex (with the **duplex** command) in interface configuration mode.
3. A and D. The IEEE autonegotiation rules dictate that if a device attempts autonegotiation but the other side does not participate, use the slowest speed it supports. However, Cisco switches override that logic, instead sampling the electrical signal to detect the speed used by the connected device, so the switch will operate at 1000 Mbps. The switch uses the IEEE default setting for duplex based on the speed, and the IEEE default for duplex when using 1000 Mbps is to use full duplex. So in this case, the switch will match both the speed and the duplex setting made on the PC.
4. A, B, and D. The disabled state in the **show interfaces status** command is the same as an "administratively down and down" state shown in the **show interfaces** command. The interface must be in a connected state (per the **show interfaces status** command) before the switch can send frames out the interface.
5. A and D. SW2 has effectively disabled IEEE standard autonegotiation by configuring both speed and duplex. However, Cisco switches can detect the speed used by the other device, even with autonegotiation turned off. Also, at 1 Gbps, the IEEE autonegotiation standard says to use full duplex. If the duplex setting cannot be negotiated, both ends use 1 Gbps, full duplex.
6. D. For the two answers about a duplex mismatch, that condition does cause collisions, particularly late collisions, but only the side using CSMA/CD logic (the half-duplex side) has any concept of collisions. So, if switch SW1 was using half duplex, and switch SW2 using full duplex, SW1 would likely see late collisions and see that counter increment over time.

   If switch SW2 had shut down its interface, switch SW1's interface would be in a down/down state, and none of the counters would increment. Also, if both switch ports had been configured with different speeds, again the ports would be in a down/down state, and none of the interface counters would increment.

Chapter 8

1. B. A VLAN is a set of devices in the same Layer 2 broadcast domain. A subnet often includes the exact same set of devices, but it is a Layer 3 concept. A collision domain refers to a set of Ethernet devices, but with different rules than VLAN rules for determining which devices are in the same collision domain.
2. D. Although a subnet and a VLAN are not equivalent concepts, the devices in one VLAN are typically in the same IP subnet and vice versa.
3. B. 802.1Q defines a 4-byte header, inserted after the original frame's destination and source MAC address fields. The insertion of this header does not change the original frame's source or destination address. The header itself holds a 12-bit VLAN ID field, which identifies the VLAN associated with the frame.

4. A and C. The **dynamic auto** setting means that the switch can negotiate trunking, but it can only respond to negotiation messages, and it cannot initiate the negotiation process. So, the other switch must be configured to trunk or to initiate the negotiation process (based on being configured with the **dynamic desirable** option).
5. A and B. The configured VTP setting of VTP transparent mode means that the switch can configure VLANs, so the VLAN is configured. In addition, the VLAN configuration details, including the VLAN name, show up as part of the running-config file.
6. B and C. The **show interfaces switchport** command lists both the administrative and operational status of each port. When a switch considers a port to be trunking, this command lists an operational trunking state of "trunk." The **show interfaces trunk** command lists a set of interfaces—the interfaces that are currently operating as trunks. So, both of these commands identify interfaces that are operational trunks.
7. A and B. On switches that do not use VTP (by using VTP modes off or transparent), the switch lists all VLAN configuration in the configuration file (making one answer correct). Also, the **show vlan brief** command lists all defined VLANs, regardless of VTP mode and regardless of shutdown state. As a result, the two answers that mention commands are correct.

   The two incorrect answers are incorrect because VLAN 30 has been shut down, which means the switch will not forward frames in that VLAN, regardless of whether they arrive on access or trunk ports.
8. B. The first list of VLAN IDs includes all VLANs (1–4094) except those overtly removed per the details in any **switchport trunk allowed vlan** interface subcommands on the trunk interface. If no such commands are configured, the first list in the output will include 1–4094. The two incorrect answers that mention VLAN 30 both list conditions that change the second of two lists of VLANs in the command output, while STP's choice to block an interface would impact the third list.

Chapter 9

1. A and B. Listening and learning are transitory port states, used only when moving from the blocking to the forwarding state. Discarding is not an STP port state.
2. C. The smallest numeric bridge ID wins the election.
3. C and D. Listening and learning are transitory port states used only when moving from the blocking to the forwarding state. Discarding is not an STP port state. Forwarding and blocking are stable states.
4. B. Nonroot switches forward Hellos received from the root; the root sends these Hellos based on the root's configured Hello timer.
5. B and D. RSTP uses port state forwarding, learning, and discarding. Forwarding and learning perform the same functions as the port states used by traditional STP.
6. A and D. With RSTP, an alternate port is an alternate to the root port when a switch's root port fails. A backup port takes over for a designated port if the designated port fails.

7. D. The PortFast feature allows STP to move a port from blocking to forwarding without going through the interim listening and learning states. STP allows this exception when the link is known to have no switch on the other end of the link, removing the risk of a switching loop. BPDU Guard is a common feature to use at the same time as PortFast because it watches for incoming bridge protocol data units (BPDU), which should not happen on an access port, and prevents the loops from a rogue switch by disabling the port.

Chapter 10

1. A. Of the four answers, only **pvst** and **rapid-pvst** are valid options on the command. Of those, the **rapid-pvst** option enables Rapid Per VLAN Spanning Tree (RPVST+), which uses RSTP. The **pvst** option enables Per VLAN Spanning Tree (PVST) which uses STP, not RSTP. The other two options, if attempted, would cause the command to be rejected because the option does not exist.

2. A and C. The system ID extension (or extended system ID) part of a bridge ID contains 12 bits and sits after the 4-bit priority field and before the 48-bit system ID. Switches use this field to store the VLAN ID when using STP or RSTP to build spanning trees per VLAN. So of the two answers that mention the system ID extension, the one that lists the VLAN ID, in this case 5, is correct.

   The output also lists a priority of 32773. However, that output lists the decimal equivalent of the 16-bit priority value. In reality, this decimal value is the sum of the configured decimal priority plus the VLAN ID: 32768 + 5 = 32773. So in this case, the root's configured priority is 32,768.

3. A, B, and D. The Cisco Rapid Per VLAN Spanning Tree (RPVST+) creates one spanning tree instance per VLAN. To do so, it sends BPDUs per-VLAN. Each switch identifies itself with a unique Bridge ID (BID) per VLAN, made unique per-VLAN by adding the VLAN ID to the system ID extension 12-bit field of the BID. RVPST also adds a new Type-Length Value (TLV) to the BPDU itself, which includes a place to list the VLAN ID. Finally, when transmitting the BPDUs over VLAN trunks, the switch uses a trunking header that lists the VLAN ID (a practice sometimes called tunneling in 802.1Q.) The receiving switch can check all three locations that list the VLAN ID to ensure that they all agree about what VLAN the BPDU is describing. Of the four answers, the three correct answers describe the three actual locations in which RPVST+ lists the VLAN ID.

4. D. IOS uses the **channel-group** configuration command to create an EtherChannel. Then the term *etherchannel* is used in the **show etherchannel** command, which displays the status of the channel. The output of this **show** command then names the channel a *PortChannel*. The only answer that is not used somewhere in IOS to describe this multilink channel is *Ethernet-Channel*.

5. B and D. The channel-group command will direct the switch to use LACP to dynamically negotiate to add a link to an EtherChannel when the command uses the **active** and **passive** keywords, respectively. The **desirable** and **passive** keywords direct the switch to use PaGP instead of LACP. Of the four answers, the two correct answers use two LACP values, while the two incorrect answers use at least one value that would cause the switch to use PaGP, making the answer incorrect.

Of the two correct answers, both combinations result in the switches attempting to add the link to an EtherChannel using LACP as the negotiation protocol. If both switches used the **passive** keyword, they would both sit and wait for the other switch to begin sending LACP messages and therefore never attempt to add the link to the channel.

6. C. EtherChannel load distribution, or load balancing, on Cisco Catalyst switches uses an algorithm. The algorithm examines some fields in the various headers, so messages that have the same values in those fields always flow over the same link in a particular EtherChannel. Note that it does not break the frames into smaller fragments nor use a round-robin approach that ignores the header values, and it does not examine link utilization when making the choice.

Chapter 11

1. B and D. The general rule to determine whether two devices' interfaces should be in the same subnet is whether the two interfaces are separated from each other by a router. To provide a way for hosts in one VLAN to send data to hosts outside that VLAN, a local router must connect its LAN interface to the same VLAN as the hosts and have an address in the same subnet as the hosts. All the hosts in that same VLAN on the same switch would not be separated from each other by a router, so these hosts would also be in the same subnet. However, another PC, connected to the same switch but in a different VLAN, will require its packets to flow through a router to reach Host A, so Host A's IP address would need to be in a different subnet compared to this new host.

2. D. By definition, two address values in every IPv4 subnet cannot be used as host IPv4 addresses: the first (lowest) numeric value in the subnet for the subnet ID and the last (highest) numeric value in the subnet for the subnet broadcast address.

3. B and C. At least 7 subnet bits are needed because $2^6 = 64$, so 6 subnet bits could not number 100 different subnets. Seven subnet bits could because $2^7 = 128 >= 100$. Similarly, 6 host bits is not enough because $2^6 - 2 = 62$, but 7 host bits is enough because $2^7 - 2 = 126 >= 100$.

   The number of network, subnet, and host bits must total 32 bits, making one of the answers incorrect. The answer with 8 network bits cannot be correct because the question states that a Class B network is used, so the number of network bits must always be 16. The two correct answers have 16 network bits (required because the question states the use of a Class B network) and at least 7 subnet and host bits each.

4. A and C. The private IPv4 networks, defined by RFC 1918, are Class A network 10.0.0.0, the 16 Class B networks from 172.16.0.0 to 172.31.0.0, and the 256 Class C networks that begin with 192.168.

5. A, D, and E. The private IPv4 networks, defined by RFC 1918, are Class A network 10.0.0.0, the 16 Class B networks from 172.16.0.0 to 172.31.0.0, and the 256 Class C networks that begin with 192.168. The three correct answers are from the public IP network range, and none are reserved values.

6. A and C. An unsubnetted Class A, B, or C network has two parts: the network and host parts.

7. B. An unsubnetted Class A, B, or C network has two parts: the network and host parts. To perform subnetting, the engineer creates a new subnet part by borrowing host bits, shrinking the number of host bits. The subnet part of the address structure exists only after the engineer chooses a nondefault mask. The network part remains a constant size.

Chapter 12

1. B and C. Class A networks have a first octet in the range of 1–126, inclusive, and their network IDs have a 0 in the last three octets. 130.0.0.0 is actually a Class B network (first octet range 128–191, inclusive). All addresses that begin with 127 are reserved, so 127.0.0.0 is not a Class A network.
2. E. All Class B networks begin with values between 128 and 191, inclusive, in their first octets. The network ID has any value in the 128–191 range in the first octet, and any value from 0 to 255 inclusive in the second octet, with decimal 0s in the final two octets. Two of the answers show a 255 in the second octet, which is acceptable. Two of the answers show a 0 in the second octet, which is also acceptable.
3. B and D. The first octet (172) is in the range of values for Class B addresses (128–191). As a result, the network ID can be formed by copying the first two octets (172.16) and writing 0s for the last two octets (172.16.0.0). The default mask for all Class B networks is 255.255.0.0, and the number of host bits in all unsubnetted Class B networks is 16.
4. A and C. The first octet (192) is in the range of values for Class C addresses (192–223). As a result, the network ID can be formed by copying the first three octets (192.168.6) and writing 0 for the last octet (192.168.6.0). The default mask for all Class C networks is 255.255.255.0, and the number of host bits in all unsubnetted Class C networks is 8.
5. D. To find the network broadcast address, first determine the class, and then determine the number of host octets. At that point, convert the host octets to 255 to create the network broadcast address. In this case, 10.1.255.255 is in a Class A network, with the last three octets as host octets, for a network broadcast address of 10.255.255.255. For 192.168.255.1, it is a Class C address, with the last octet as the host part, for a network broadcast address of 192.168.255.255. Address 224.1.1.255 is a Class D address, so it is not in any unicast IP network and the question does not apply. For 172.30.255.255, it is a Class B address, with the last two octets as host octets, so the network broadcast address is 172.30.255.255.

Chapter 13

1. C. If you think about the conversion one octet at a time, the first two octets each convert to 8 binary 1s. 254 converts to 8-bit binary 11111110, and decimal 0 converts to 8-bit binary 00000000. So, the total number of binary 1s (which defines the prefix length) is 8 + 8 + 7 + 0 = /23.
2. B. If you think about the conversion one octet at a time, the first three octets each convert to 8 binary 1s. 240 converts to 8-bit binary 11110000, so the total number of binary 1s (which defines the prefix length) is 8 + 8 + 8 + 4 = /28.

3. B. /30 is the equivalent of the mask that in binary has 30 binary 1s. To convert that to DDN format, write down all the binary 1s (30 in this case), followed by binary 0s for the remainder of the 32-bit mask. Then take 8 bits at a time and convert from binary to decimal (or memorize the nine possible DDN mask octet values and their binary equivalents). Using the /30 mask in this question, the binary mask is 11111111 11111111 11111111 11111100. Each of the first three octets is all binary 1s, so each converts to 255. The last octet, 11111100, converts to 252, for a DDN mask of 255.255.255.252. See Appendix A, "Numeric Reference Tables," for a decimal/binary conversion table.
4. C. The size of the network part is always either 8, 16, or 24 bits, based on whether it is Class A, B, or C, respectively. As a Class A address, N=8. The mask 255.255.255.0, converted to prefix format, is /24. The number of subnet bits is the difference between the prefix length (24) and N, so S=16 in this case. The size of the host part is a number that, when added to the prefix length (24), gives you 32, so H=8 in this case.
5. A. The size of the network part is always either 8, 16, or 24 bits, based on whether it is Class A, B, or C, respectively. As a Class C address, N=24. The number of subnet bits is the difference between the prefix length (27) and N, so S=3 in this case. The size of the host part is a number that, when added to the prefix length (27), gives you 32, so H=5 in this case.
6. D. Classless addressing rules define a two-part IP address structure: the prefix and the host part. This logic ignores Class A, B, and C rules, and can be applied to the 32-bit IPv4 addresses from any address class. By ignoring Class A, B, and C rules, classless addressing ignores any distinction as to the network part of an IPv4 address.
7. A and B. The masks in binary define a number of binary 1s, and the number of binary 1s defines the length of the prefix (network + subnet) part. With a Class B network, the network part is 16 bits. To support 100 subnets, the subnet part must be at least 7 bits long. Six subnet bits would supply only $2^6 = 64$ subnets, while 7 subnet bits supply $2^7 = 128$ subnets. The /24 answer supplies 8 subnet bits, and the 255.255.255.252 answer supplies 14 subnet bits.

Chapter 14

1. D. When using classful IP addressing concepts as described in Chapter 13, "Analyzing Subnet Masks," addresses have three parts: network, subnet, and host. For addresses in a single classful network, the network parts must be identical for the numbers to be in the same network. For addresses in the same subnet, both the network and subnet parts must have identical values. The host part differs when comparing different addresses in the same subnet.
2. B and D. In any subnet, the subnet ID is the smallest number in the range, the subnet broadcast address is the largest number, and the usable IP addresses sit between them. All numbers in a subnet have identical binary values in the prefix part (classless view) and network + subnet part (classful view). To be the lowest number, the subnet ID must have the lowest possible binary value (all 0s) in the host part. To be the largest number, the broadcast address must have the highest possible binary value (all binary 1s) in the host part. The usable addresses do not include the subnet ID and subnet broadcast address, so the addresses in the range of usable IP addresses never have a value of all 0s or 1s in their host parts.

3. C. The mask converts to 255.255.255.0. To find the subnet ID, for each octet of the mask that is 255, you can copy the IP address's corresponding values. For mask octets of decimal 0, you can record a 0 in that octet of the subnet ID. As such, copy the 10.7.99 and write a 0 for the fourth octet, for a subnet ID of 10.7.99.0.
4. C. First, the resident subnet (the subnet ID of the subnet in which the address resides) must be numerically smaller than the IP address, which rules out one of the answers. The mask converts to 255.255.255.252. As such, you can copy the first three octets of the IP address because of their value of 255. For the fourth octet, the subnet ID value must be a multiple of 4, because 256 – 252 (mask) = 4. Those multiples include 96 and 100, and the right choice is the multiple closest to the IP address value in that octet (97) without going over. So, the correct subnet ID is 192.168.44.96.
5. C. The resident subnet ID in this case is 172.31.77.192. You can find the subnet broadcast address based on the subnet ID and mask using several methods. Following the decimal process in the book, the mask converts to 255.255.255.224, making the interesting octet be octet 4, with magic number 256 – 224 = 32. For the three octets where the mask = 255, copy the subnet ID (172.31.77). For the interesting octet, take the subnet ID value (192), add magic (32), and subtract 1, for 223. That makes the subnet broadcast address 172.31.77.223.
6. C. To answer this question, you need to find the range of addresses in the subnet, which typically then means you need to calculate the subnet ID and subnet broadcast address. With a subnet ID/mask of 10.1.4.0/23, the mask converts to 255.255.254.0. To find the subnet broadcast address, following the decimal process described in this chapter, you can copy the subnet ID's first two octets because the mask's value is 255 in each octet. You write a 255 in the fourth octet because the mask has a 0 on the fourth octet. In octet 3, the interesting octet, add the magic number (2) to the subnet ID's value (4), minus 1, for a value of 2 + 4 – 1 = 5. (The magic number in this case is calculated as 256 – 254 = 2.) That makes the broadcast address 10.1.5.255. The last usable address is 1 less: 10.1.5.254. The range that includes the last 100 addresses is 10.1.5.155 – 10.1.5.254.

Chapter 15

1. B and E. Cisco routers have an on/off switch, but Cisco switches generally do not.
2. B. Cisco routers that do not also have any Layer 2 switch features support commands needed for Layer 3 routing as well as commands in common between Layer 2 switching and Layer 3 routing devices. In this case, the **show interfaces status** and **show mac address-table** commands happen to be commands supported on Layer 2 switches but not on routers. Both types of devices use the **show running-config** command. Of the answers, only the **show ip interface brief** command is unique to routers.
3. A and C. To route packets on an interface, the router interface configuration must include an IP address and mask. One correct command shows the correct single command used to configure both values, while one incorrect command shows those settings as two separate commands. Also, to route packets, the interface must reach an "up/up" state; that is, the **show interfaces** and other commands list two status values, and both must be "up." The **no shutdown** command enables the interface.

4. B. If the first of the two status codes is "down," it typically means that a Layer 1 problem exists. In this case, the question states that the router connects to a switch with a UTP straight-through cable, which is the correct cable pinout. Of the two answers that mention the **shutdown** command, if the router interface were shut down, the first router status code would be "administratively down," so that answer is incorrect. However, if the neighboring device interface sits in a shutdown state, the router will sense no electrical signals over the cable, seeing that as a physical problem, and place the interface into a "down/down" state, making that answer correct.

   Second, the two answers that mention interface IP addresses have no impact on the status codes of the **show interfaces brief** command. Both answers imply that the interface does not have an IP address configured. However, both the first and second status codes are not related to whether IP addresses have been configured or not, making both answers incorrect.

5. C and E. The **show ip interface brief** command lists all the interface IPv4 addresses but none of the masks. The **show version** command lists none of the IP addresses and none of the masks. The other three commands list both the address and mask.

6. B. A router has one IPv4 address for each interface in use, whereas a LAN switch has a single IPv4 address that is just used for accessing the switch. The rest of the answers list configuration settings that use the same conventions on both routers and switches.

Chapter 16

1. A and C. The route defines the group of addresses represented by the route using the subnet ID and mask. The router can use those numbers to find the range of addresses that should be matched by this route. The other two answers list facts useful when forwarding packets that happen to match the route.

2. A and D. First, for the subnetting math, address 10.1.1.100, with mask /26, implies a subnet ID of 10.1.1.64. Also, mask /26 converts to a DDN mask of 255.255.255.192. For any working router interface, after adding the **ip address** command to configure an address and mask, the router adds a connected route for the subnet. In this case, that means the router adds a connected route for subnet 10.1.1.64 255.255.255.192. The router also adds a route called a local route, which is a route for the interface IP address with a 255.255.255.255 mask. In this case, that means the router adds a local route for address 10.1.1.100 with mask 255.255.255.255.

3. C. The **ip route** command can refer to the IP address of the next-hop router or to the local router's interface. It also refers to the subnet ID and matching subnet mask, defining the range of addresses matched by the route.

4. A. The correct syntax lists a subnet number, then a subnet mask in dotted-decimal form, and then either an outgoing interface or a next-hop IP address.

5. B. The **ip route** command can reference an outgoing interface or a next-hop IP address, and the command lists a next-hop IP address, which rules out one answer. The command does use the correct syntax, ruling out another answer. There is no requirement for a router to have any particular interface IP addresses in relation to the configuration of an **ip route** command, ruling out yet another answer.

The checks that IOS uses when looking at a new **ip route** command include whether the outgoing interface is up/up, whether the next-hop address is reachable, and, if there is a competing route from another source, whether the other route has a better administrative distance.

6. D. Destination address 10.1.15.122 matches all the routes listed except the host route to 10.1.15.100/32. In that case, the router will choose the matching route that has the longest prefix length, that is, the prefix-style mask with the highest number. In this case, that route lists subnet 10.1.15.96 and mask /27, which lists interface G0/3/0 as the outgoing interface.

Chapter 17

1. A and F. Of all the commands listed, only the two correct answers are syntactically correct router configuration commands. The command to enable 802.1Q trunking is **encapsulation dot1q** *vlan_id*.
2. B and C. Subinterface G0/1.1 must be in an administratively down state due to the **shutdown** command being issued on that subinterface. For subinterface G0/1.2, its status cannot be administratively down because of the **no shutdown** command. G0/1.2's state will then track to the state of the underlying physical interface. With a physical interface state of down/down, subinterface G0/1.2 will be in a down/down state in this case.
3. C. The configuration of the Layer 3 switch's routing feature uses VLAN interfaces. The VLAN interface numbers must match the associated VLAN ID, so with VLANs 1, 2, and 3 in use, the switch will configure **interface vlan 1**, **interface vlan 2** (which is the correct answer), and **interface vlan 3**. The matching connected routes, like all connected IP routes, will list the VLAN interfaces.

   As for the incorrect answers, a list of connected routes will not list any next-hop IP addresses. Each route will list an outgoing interface; the outgoing interface will not be a physical interface, but rather a VLAN interface, because the question states that the configuration uses SVIs. Finally, all the listed subnets have a /25 mask, which is 255.255.255.128, so none of the routes will list a 255.255.255.0 mask.
4. C and D. First, for the correct answers, a Layer 3 switch will not route packets on a VLAN interface unless it is in an up/up state. A VLAN interface will only be up/up if the matching VLAN (with the same VLAN number) exists on the switch. If VTP deletes the VLAN, then the VLAN interface moves to a down/down state, and routing in/out that interface stops. Also, disabling VLAN 2 with the **shutdown** command in VLAN configuration mode also causes the matching VLAN 2 interface to fail, which makes routing on interface VLAN 2 stop as well.

   As for the incorrect answers, a Layer 3 switch needs only one access port or trunk port forwarding for a VLAN to enable routing for that VLAN, so nine of the ten access ports in VLAN 2 could fail, leaving one working port, and the switch would keep routing for VLAN 2.

   A **shutdown** of VLAN 4 has no effect on routing for VLAN interfaces 2 and 3. Had that answer listed VLANs 2 or 3, it would definitely be a reason to make routing fail for that VLAN interface.

5. A and C. With a Layer 3 EtherChannel, the physical ports and the port-channel interface must disable the behavior of acting like a switch port, and therefore act like a routed port, through the configuration of the **no switchport** interface subcommand. (The **routedport** command is not an IOS command.) Once created, the physical interfaces should not have an IP address configured. The port-channel interface (the interface representing the EtherChannel) should be configured with the IP address.
6. B and C. With a Layer 3 EtherChannel, two configuration settings must be the same on all the physical ports, specifically the speed and duplex as set with the **speed** and **duplex** commands. Additionally, the physical ports and port-channel port must all have the **no switchport** command configured to make each act as a routed port. So, having a different speed setting, or being configured with **switchport** rather than **no switchport**, would prevent IOS from adding interface G0/2 to the Layer 3 EtherChannel.

   As for the wrong answers, both have to do with Layer 2 configuration settings. Once Layer 2 operations have been disabled because of the **no switchport** command, those settings related to Layer 2 that could cause problems on Layer 2 EtherChannels do not then cause problems for the Layer 3 EtherChannel. So, Layer 2 settings about access VLANs, trunking allowed lists, and STP settings, which must match before an interface can be added to a Layer 2 EtherChannel, do not matter for a Layer 3 EtherChannel.

Chapter 19

1. D. Both versions of RIP use distance vector logic, and EIGRP uses a different kind of logic, characterized either as advanced distance vector or a balanced hybrid.
2. C and D. Both versions of RIP use the same hop-count metric, neither of which is affected by link bandwidth. EIGRP's metric, by default, is calculated based on bandwidth and delay. OSPF's metric is a sum of outgoing interfaces costs, with those costs (by default) based on interface bandwidth.
3. B, C, and D. Of the listed routing protocols, only the old RIP Version 1 (RIP-1) protocol does not support variable-length subnet masks (VLSM).
4. C. LSAs contain topology information that is useful in calculating routes, but the LSAs do not directly list the route that a router should add to its routing table. In this case, R1 would run a calculation called the Shortest Path First (SPF) algorithm, against the LSAs, to determine what IP routes to add to the IP routing table.
5. B. Neighboring OSPF routers that complete the database exchange are considered fully adjacent and rest in a full neighbor state. The up/up and final states are not OSPF states at all. The 2-way state is either an interim state or a stable state between some routers on the same VLAN.
6. C. The correct answer is the one advantage of using a single-area design. The three wrong answers are advantages of using a multiarea design, with all reasons being much more important with a larger internetwork.

Chapter 20

1. B. The **network 10.0.0.0 0.255.255.255 area 0** command works because it matches all interfaces whose first octet is 10. The rest of the commands match as follows: all addresses that end with 0.0.0 (wildcard mask 255.0.0.0); all addresses that begin with 10.0.0 (wildcard mask 0.0.0.255); and all addresses that begin with 10.0 (wildcard mask 0.0.255.255).
2. A. The **network 10.1.0.0 0.0.255.255 area 0** command matches all IP addresses that begin with 10.1, enabling OSPF in area 0 on all interfaces. The answer with wildcard mask 0.255.255.0 is illegal because it represents more than one string of binary 0s separated by binary 1s. The answer with x's is syntactically incorrect. The answer with wildcard mask 255.0.0.0 means "Match all addresses whose last three octets are 0.0.0," so none of the three interfaces are matched.
3. A and E. Of the three wrong answers, two are real commands that simply do not list the OSPF neighbors. **show ip ospf interface brief** lists interfaces on which OSPF is enabled but does not list neighbors. **show ip interface** lists IPv4 details about interfaces, but none related to OSPF. One incorrect answer, **show ip neighbor**, is not a valid IOS command.
4. B. With OSPFv2 interface configuration mode, the configuration looks just like the traditional configuration, with a couple of exceptions. The **network** router subcommand is no longer required. Instead, each interface on which OSPF should be enabled is configured with an **ip ospf** *process-id* **area** *area-id* interface subcommand. This command refers to the OSPF routing process that should be enabled on the interface and specifies the OSPFv2 area.
5. B. SPF calculates the cost of a route as the sum of the OSPF interface costs for all outgoing interfaces in the route. The interface cost can be set directly (**ip ospf cost**), or IOS uses a default based on the reference bandwidth and the interface bandwidth. Of the listed answers, **delay** is the only setting that does not influence OSPFv2 metric calculations.
6. A and D. The configuration enables OSPF and identifies the area number to use with the interface using an interface subcommand in interface mode: the **ip ospf** *process-id* **area** *area-number* command. However, to explicitly configure the router ID, the configuration must use the **router-id** *router-id-value* command, which is a command issued in OSPF router mode.

Chapter 21

1. B and D. By default, IOS assigns Ethernet interfaces an OSPF network type of broadcast, with an OSPF interface priority of 1. As a result, both routers attempt to discover the other routers on the link (which identifies one correct answer).

   The broadcast network type means that the routers also attempt to elect a DR and BDR. With a tie-in priority, the routers choose the DR based on the highest router ID (RID) values, meaning that R2 will become the DR and R1 will become the BDR. These facts combine to show why the two incorrect answers are incorrect. The other correct answer is correct because the **show ip ospf neighbor** command lists the local router's neighbor relationship state (FULL) and the role filled by that neighbor (DR), which would be the output shown on R1 when R2 is acting as DR.
2. B and C. First, the OSPF point-to-point network type causes the two routers to dynamically discover neighbors, making one answer correct.

Next, IOS assigns a default OSPF interface priority of 1, so R1's configured priority of 11 would be better in a DR/BDR election. However, the point-to-point network type causes the router to not use a DR/BDR on the interface. As a result, the answer about R1 becoming the DR is incorrect (because no DR exists at all), and the answer listing a state of "FULL/DR" is incorrect for the same reason. However, the answer that claims that R2 will be neither DR nor BDR is true because no DR or BDR is elected.

3. D. The **show ip ospf interface brief** command lists a pair of counters under the heading "Nbrs F/C" on the far right of the output. The first of the two numbers represents the number of fully adjacent neighbors (2 in this case), and the second number represents the total number of neighbors.

4. A and D. As worded, the correct answers list a scenario that would prevent the neighbor relationship. One correct answer mentions the use of two different OSPF areas on the potential OSPF neighbors; to become neighbors, the two routers must use the same area number. The other correct answer mentions the use of two different Hello timers, a mismatch that causes two routers to reject each other and to not become neighbors.

   The two incorrect answers list scenarios that do not cause issues, making them incorrect answers. One mentions mismatched OSPF process IDs; OSPF process IDs do not need to match for two routers to become neighbors. The other incorrect answer (that is, a scenario that does not cause a problem) mentions the use of two different priority values. The priority values give OSPF a means to prefer one router over the other when electing a DR/BDR, so the setting is intended to be set to different values on different routers and does not cause a problem.

5. C. As worded, the correct answers should be a scenario that would prevent the neighbor relationship. The answers all list values that are identical or similar on the two routers. Of those, the use of an identical OSPF router ID (RID) on the two routers prevents them from becoming neighbors, making that one answer correct.

   Of the incorrect answers, both routers must have the same Dead interval, so both using a Dead interval of 40 causes no issues. The two routers can use any OSPF process ID (the same or different value, it does not matter), making that answer incorrect. Finally, the two routers' IP addresses must be in the same subnet, so again that scenario does not prevent R13 and R14 from becoming neighbors.

6. D. The OSPF **shutdown** command tells the OSPF process to stop operating. That process includes removing any OSPF-learned routes from the IP routing table, clearing the router's LSDB, and closing existing OSPF neighbor relationships. In effect, it causes OSPF to stop working on the router, but it does retain the configuration so that a **no shutdown** command will cause the router to start using OSPF again with no changes to the configuration.

Chapter 22

1. C. NAT, specifically the PAT feature that allows many hosts to use private IPv4 addresses while being supported by a single public IPv4 address, was one short-term solution to the IPv4 address exhaustion problem. IP version 5 existed briefly as an experimental protocol and had nothing to do with IPv4 address exhaustion. IPv6 directly addresses the IPv4 address exhaustion problem, but it is a long-term solution. ARP has no impact on the number of IPv4 addresses used.

2. A. Routers use the same process steps when routing IPv6 packets as they do when routing IPv4 packets. Routers route IPv6 packets based on the IPv6 addresses, listed inside the IPv6 header in the IPv6 packets, by comparing the destination IPv6 address to the router's IPv6 routing table. As a result, the router discards the incoming frame's data-link header and trailer, leaving an IPv6 packet. The router compares the destination (not source) IPv6 address in the header to the router's IPv6 (not IPv4) routing table and then forwards the packet based on the matched route.
3. D. If you are following the steps in the book, the first step removes up to three leading 0s in each quartet, leaving FE80:0:0:100:0:0:0:123. This leaves two strings of consecutive all-0 quartets; by changing the longest string of all 0s to ::, the address is FE80:0:0:100::123.
4. B. This question has many quartets that make it easy to make a common mistake: removing trailing 0s in a quartet of hex digits. To abbreviate IPv6 addresses, only leading 0s in a quartet should be removed. Many of the quartets have trailing 0s (0s on the right side of the quartet), so make sure to not remove those 0s.
5. A. The unabbreviated version of an IPv6 address must have 32 digits, and only one answer has 32 hex digits. In this case, the original number shows four quartets and a ::. So, the :: was replaced with four quartets of 0000, making the number have eight quartets. Then, for each quartet with fewer than four digits, leading 0s were added so that each quartet has four hex digits.
6. C. The /64 prefix length means that the last 64 bits, or last 16 digits, of the address should be changed to all 0s. That process leaves the unabbreviated prefix as 2000:0000:0000:0005:0000:0000:0000:0000. The last four quartets are all 0s, making that string of all 0s be the longest and best string of 0s to replace with ::. After removing the leading 0s in other quartets, the answer is 2000:0:0:5::/64.

Chapter 23

1. C. Unique local addresses begin with FD in the first two digits.
2. A. Global unicast addresses can begin with many different initial values, but most commonly begin with either a hex 2 or 3.
3. D. The global routing prefix is the address block, represented as a prefix value and prefix length, given to an organization by some numbering authority. All IPv6 addresses inside the company have the same value in these initial bits of their IPv6 addresses. Similarly, when a company uses a public IPv4 address block, all the addresses have the same value in the network part.
4. B. Subnetting a global unicast address block, using a single prefix length for all subnets, breaks the addresses into three parts. The parts are the global routing prefix, subnet, and interface ID.
5. D. Unique local addresses begin with a 2-hex-digit prefix of FD, followed by the 10-hex-digit global ID.

Chapter 24

1. A. The one correct answer lists the exact same IPv6 address listed in the question, with a /64 prefix length and no spaces in the syntax of the answer. Another (incorrect) answer is identical, except that it leaves a space between the address and prefix length, which is incorrect syntax. The two answers that list the **eui-64** parameter list an address and not a prefix; they should list a prefix to be correct, although neither would have resulted in the IPv6 address listed in the question.
2. B. With the **eui-64** parameter, the router will calculate the interface ID portion of the IPv6 address based on its MAC address. Beginning with 5055.4444.3333, the router injects FF FE in the middle (5055.44FF.FE44.3333). Then the router inverts the seventh bit in the first byte. Mentally, this converts hex 50 to binary 01010000, changing bit 7 so that the string is 0101 0010 and converting back to hex 52. The final interface ID value is 5255:44FF:FE44:3333. The wrong answers simply list a different value.
3. A and C. Of the four answers, the two correct answers show the minimal required configuration to support IPv6 on a Cisco router: enabling IPv6 routing (**ipv6 unicast-routing**) and enabling IPv6 on each interface, typically by adding a unicast address to each interface (**ipv6 address…**). The two incorrect answers list nonexistent commands.
4. A. With an **ipv6 address** command configured for a global unicast address, but without a link-local address configured with an **ipv6 address** command, the router calculates its link-local address on the interface based on its MAC address and EUI-64 rules. The first half of the link-local address begins FE80:0000:0000:0000. The router then calculates the second half of the link-local address value by taking the MAC address (0200.0001.000A), injecting FF FE in the middle (0200.00FF.FE01.000A), and flipping the seventh bit (0000.00FF.FE01.000A).
5. B. FF02::1 is used by all IPv6 hosts on the link, FF02::5 is used by all OSPFv3 routers, and FF02::A is used by all EIGRPv6 routers. FF02::2 is used to send packets to all IPv6 routers on a link.

Chapter 25

1. A and C. With an IPv6 address on a working interface, the router adds a connected route for the prefix (subnet) implied by the **ipv6 address** command. It also adds a local host route (with a /128 prefix length) based on the unicast address. The router does not add a route based on the link-local address.
2. A and C. The two correct answers show the correct subnet ID (prefix) and prefix length for the two connected subnets: 3111:1:1:1::/64 and 3222:2:2:2::/64. The answer with the /128 prefix length is shown in a local route, but those routes are not displayed by the **show ipv6 route connected** command. The other incorrect answer lists the entire IPv6 address with a /64 prefix length, and the entire address would not be displayed as a prefix when using a /64 prefix.

3. A. All four answers show examples of commands that use an outgoing interface. The two commands that begin with **ip route** define only IPv4 routes; the commands would be rejected because of the IPv6 prefixes listed in the commands. The two commands that begin with **ipv6 route** are syntactically correct, but the command should list the local router's interface (an interface on the router on which the command is being configured). R5 needs to use its local S0/1/1 interface as the outgoing interface.

4. B. All four answers show examples of commands that use a next-hop router IPv6 address. Two of the answers list R5's own IPv6 address (unicast or link-local), which is incorrect; the answer should be an address on the neighboring router, R6 in this case. For the two answers that list addresses on Router R6, the one that lists R6's global unicast address is correct. The one that lists R6's link-local address would also require R5's outgoing interface, so the answer that lists FE80::FF:FE00:6 would be rejected as well.

5. C. IOS will add a new static route to the IPv6 routing table if, when using a next-hop global unicast address, the router has a working route to reach that next-hop address and there is no better (lower administrative distance) route for the exact same subnet. So, the correct answer identifies one reason why the route would not appear. The answer that mentions a better route with administrative distance of 110 is a valid reason for the static route to not appear, but the question states that no route for the subnet appears in the routing table, so clearly that competing route does not exist.

   The other two answers are incorrect about the **ipv6 route** command. This command can use a link-local next-hop address but does not have to do so. Also, when using a global unicast address as next-hop, the command does not also require an outgoing interface parameter.

6. A and B. The output shows two static routes, as noted with the "S" code on the far left. Both were added to the IPv6 routing table because of **ipv6 route** commands. Both have an administrative distance of 1, which is listed as the first number in brackets.

   For the two incorrect answers, note that the **ipv6 address** interface subcommand does cause IOS to add connected IPv6 routes to the routing table, and the phrase "directly connected" with one route might make you think this is a connected route. However, the "S" in the far left identifies the source of the route. Likewise, the answer that mentions an IPv6 routing protocol is incorrect because both routes have a code of S, meaning static.

7. B. PC1 needs to discover PC2's MAC address. Unlike IPv4, IPv6 does not use ARP, instead using NDP. Specifically, PC1 uses the NDP Neighbor Solicitation (NS) message to request that PC2 send back an NDP Neighbor Advertisement (NA). SLAAC relates to address assignment, and not to discovering a neighbor's MAC address.

8. A and C. The NDP RA lists the router IPv6 address, the IPv6 prefixes known on the link, and the matching prefix lengths. When using DHCPv6, the host learns the IPv6 address of the DNS server through DHCPv6 messages. For MAC addresses of on-link neighbors, hosts use NDP NS and NA messages.

Chapter 26

1. C. The IEEE 802.3 standard defines Ethernet, while 802.11 defines Wi-Fi.
2. B. WLANs require half-duplex operation because all stations must contend for use of a channel to transmit frames.
3. C. An AP offers a basic service set (BSS). BSA is incorrect because it is a Basic Service Area, or the cell footprint of a BSS. BSD is incorrect because it does not pertain to wireless at all. IBSS is incorrect because it is an Independent BSS, or an ad hoc network, where an AP or BSS is not needed at all.
4. B. The AP at the heart of a BSS or cell identifies itself (and the BSS) with a Basic Service Set Identifier (BSSID). It also uses an SSID to identify the wireless network, but that is not unique to the AP or BSS. Finally, the radio MAC address is used as the basis for the BSSID value, but the value can be altered to form the BSSID for each SSID that the AP supports.
5. B. A workgroup bridge acts as a wireless client, but bridges traffic to and from a wired device connected to it.
6. B. In a mesh network, each mesh AP builds a standalone BSS. The APs relay client traffic to each other over wireless backhaul links, rather than wired Ethernet. Therefore, Ethernet cabling to each AP is not required.
7. D and E. Wi-Fi commonly uses the 2.4- and 5-GHz bands.
8. C and D. In the 2.4-GHz band, consecutively numbered channels are too wide to not overlap. Only channels 1, 6, and 11 are spaced far enough apart to avoid overlapping each other. In the 5-GHz band, all channels are considered to be nonoverlapping.

Chapter 27

1. A. An autonomous AP can operate independently without the need for a centralized wireless LAN controller.
2. B. The Cisco Meraki APs are autonomous APs that are managed through a centralized platform in the Meraki cloud.
3. C. On a lightweight AP, the MAC function is divided between the AP hardware and the WLC. Therefore, the architecture is known as split-MAC.
4. B. An LAP builds a CAPWAP tunnel with a WLC.
5. A. A trunk link carrying three VLANs is not needed at all. A lightweight AP in local mode needs only an access link with a single VLAN; everything else is carried over the CAPWAP tunnel to a WLC. The WLC will need to be connected to three VLANs so that it can work with the LAP to bind them to the three SSIDs.
6. C. A unified WLC deployment model is based around locating the WLC in a central location, to support a very large number of APs.
7. A. The local mode is the default mode, where the AP provides at least one functional BSS that wireless clients can join to connect to the network. Normal and client modes are not valid modes. Monitor mode is used to turn the AP into a dedicated wireless sensor.
8. D. The SE-Connect mode is used for spectrum analysis. "SE" denotes the Cisco Spectrum Expert software. Otherwise, an AP can operate in only one mode at a time. The local mode is the default mode.

Chapter 28

1. D. For effective security, you should leverage authentication, MIC, and encryption.
2. C. A message integrity check (MIC) is an effective way to protect against data tampering. WIPS is not correct because it provides intrusion protection functions. WEP is not correct because it does not provide data integrity along with its weak encryption. EAP is not correct because it defines the framework for authentication.
3. D. WEP is known to have a number of weaknesses and has been compromised. Therefore, it has been officially deprecated and should not be used in a wireless network. AES is not a correct answer because it is the current recommended encryption method. WPA is not correct because it defines a suite of security methods. EAP is not correct because it defines a framework for authentication.
4. C. EAP works with 802.1x to authenticate a client and enable access for it. Open authentication and WEP cannot be correct because both define a specific authentication method. WPA is not correct because it defines a suite of security methods in addition to authentication.
5. A. The TKIP method was deprecated when the 802.11 standard was updated in 2012. CCMP and GCMP are still valid methods. EAP is an authentication framework and is not related to data encryption and integrity.
6. C. WPA2 uses CCMP only. WEP has been deprecated and is not used in any of the WPA versions. TKIP has been deprecated but can be used in WPA only. WPA is not a correct answer because it is an earlier version of WPA2.
7. B. The Wi-Fi Alliance offers the WPA, WPA2, and WPA3 certifications for wireless security. WEP, AES, and 802.11 are not certifications designed and awarded by the Wi-Fi Alliance.
8. A and C. The personal mode for WPA, WPA2, and WPA3 is used to require a preshared key authentication. Enterprise mode uses 802.1x instead.

Chapter 29

1. A. A lightweight AP requires connectivity to only a single VLAN, so access mode is used.
2. B. An autonomous AP must connect to each of the VLANs it will extend to wireless LANs. Therefore, its link should be configured as a trunk.
3. D. You can use HTTP and HTTPS to access the GUI of a wireless LAN controller, as well as SSH to access its CLI. While HTTP is a valid management protocol on a WLC, it is usually disabled to make the WLC more secure.
4. C. Controllers use a link aggregation group (LAG) to bundle multiple ports together.
5. D. A dynamic interface makes a logical connection between a WLAN and a VLAN, all internal to the controller.
6. C and D. A WLAN binds an SSID to a controller interface so that the controller can link the wired and wireless networks. Although the WLAN ultimately reaches a wired VLAN, it does so only through a controller interface. It is the interface that is configured with a VLAN number.

7. C. You can configure a maximum of 512 WLANs on a controller. However, a maximum of only 16 of them can be configured on an AP.
8. A and C. The SSID and controller interface are the only parameters from the list that are necessary. The VLAN number is not because it is supplied when a controller interface is configured.

GLOSSARY

NUMERIC

10/100 A short reference to an Ethernet NIC or switch port that supports speed of 10 Mbps and 100 Mbps.

10/100/1000 A short reference to an Ethernet NIC or switch port that supports speeds of 10 Mbps, 100 Mbps, and 1000 Mbps (that is, 1 Gbps).

10BASE-T The 10-Mbps baseband Ethernet specification using two pairs of twisted-pair cabling (Categories 3, 4, or 5): one pair transmits data and the other receives data. 10BASE-T, which is part of the IEEE 802.3 specification, has a distance limit of approximately 100 m (328 feet) per segment.

100BASE-T A name for the IEEE Fast Ethernet standard that uses two-pair copper cabling, a speed of 100 Mbps, and a maximum cable length of 100 meters.

1000BASE-T A name for the IEEE Gigabit Ethernet standard that uses four-pair copper cabling, a speed of 1000 Mbps (1 Gbps), and a maximum cable length of 100 meters.

2-way state In OSPF, a neighbor state that implies that the router has exchanged Hellos with the neighbor and that all required parameters match.

802.11a The IEEE standard for wireless LANs using the U-NII spectrum, OFDM encoding, and speeds of up to 54 Mbps.

802.11b The IEEE standard for wireless LANs using the ISM spectrum, DSSS encoding, and speeds of up to 11 Mbps.

802.11g The IEEE standard for wireless LANs using the ISM spectrum, OFDM or DSSS encoding, and speeds of up to 54 Mbps.

802.11n The IEEE standard for wireless LANs using the ISM spectrum, OFDM encoding, and multiple antennas for single-stream speeds up to 150 Mbps.

802.1Q The IEEE standardized protocol for VLAN trunking, which also includes RSTP details.

802.1x An IEEE standard that defines port-based access control for wired and wireless networks.

A

AAA Authentication, authorization, and accounting. Authentication confirms the identity of the user or device. Authorization determines what the user or device is allowed to do. Accounting records information about access attempts, including inappropriate requests.

AAA server A server that holds security information and provides services related to user login, particularly authentication (is the user who he says he is?), authorization (once authenticated, what do we allow the user to do?), and accounting (tracking the user).

ABR *See* Area Border Router.

access interface A LAN network design term that refers to a switch interface connected to end-user devices, configured so that it does not use VLAN trunking.

access layer In a campus LAN design, the switches that connect directly to endpoint devices (servers, user devices), and also connect into the distribution layer switches.

access link In Frame Relay, the physical serial link that connects a Frame Relay DTE device, usually a router, to a Frame Relay switch. The access link uses the same physical layer standards as do point-to-point leased lines.

access point (AP) A device that provides wireless service for clients within its coverage area or cell, with the AP connecting to both the wireless LAN and the wired Ethernet LAN.

accounting In security, the recording of access attempts. *See also* AAA.

ad hoc network *See* independent basic service set (IBSS).

address block A set of consecutive IPv4 addresses. The term is most often used for a classless prefix as defined by CIDR but can also refer to any subnet or IPv4 network.

adjacent-layer interaction The general topic of how, on one computer, two adjacent layers in a networking architectural model work together, with the lower layer providing services to the higher layer.

administrative distance In Cisco routers, a means for one router to choose between multiple routes to reach the same subnet when those routes were learned by different routing protocols. The lower the administrative distance, the better the source of the routing information.

ADSL Asymmetric digital subscriber line. One of many DSL technologies, ADSL is designed to deliver more bandwidth downstream (from the central office to the customer site) than upstream.

all-nodes multicast address A specific IPv6 multicast address, FF02::1, with link-local scope, used to send packets to all devices on the link that support IPv6.

all-routers multicast address A specific IPv6 multicast address, FF02::2, with link-local scope, used to send packets to all devices that act as IPv6 routers on the local link.

alternate port With RSTP, a port role in which the port acts as an alternative to a switch's root port, so that when the switch's root port fails, the alternate port can immediately take over as the root port.

anycast address An address shared by two or more hosts that exist in different parts of the network, so that by design, the routers will forward packets to the nearest of the two servers, allowing clients to communicate with the nearest such server, not caring which particular server with which the client communicates.

Area Border Router (ABR) A router using OSPF in which the router has interfaces in multiple OSPF areas.

ARP Address Resolution Protocol. An Internet protocol used to map an IP address to a MAC address. Defined in RFC 826.

ARP table A list of IP addresses of neighbors on the same VLAN, along with their MAC addresses, as kept in memory by hosts and routers.

ARPANET The first packet-switched network, first created around 1970, which served as the predecessor to the Internet.

ASBR Autonomous System Border Router. A router using OSPF in which the router learns routes via another source, usually another routing protocol, exchanging routes that are external to OSPF with the OSPF domain.

asymmetric A feature of many Internet access technologies, including DSL, cable, and modems, in which the downstream transmission rate is higher than the upstream transmission rate.

asynchronous The lack of an imposed time ordering on a bit stream. Practically, both sides agree to the same speed, but there is no check or adjustment of the rates if they are slightly different. However, because only 1 byte per transfer is sent, slight differences in clock speed are not an issue.

authentication In security, the verification of the identity of a person or a process. *See also* AAA.

authentication server (AS) An 802.1x entity that authenticates users or clients based on their credentials, as matched against a user database. In a wireless network, a RADIUS server is an AS.

authenticator An 802.1x entity that exists as a network device that provides access to the network. In a wireless network, a WLC acts as an authenticator.

authorization In security, the determination of the rights allowed for a particular user or device. *See also* AAA.

autonegotiation An IEEE standard mechanism (802.3u) with which two nodes can exchange messages for the purpose of choosing to use the same Ethernet standards on both ends of the link, ensuring that the link functions and functions well.

autonomous AP A wireless AP operating in a standalone mode, such that it can provide a fully functional BSS and connect to the DS.

autonomous system An internetwork in the administrative control of one organization, company, or governmental agency, inside which that organization typically runs an interior gateway protocol (IGP).

auxiliary port A physical connector on a router that is designed to be used to allow a remote terminal, or PC with a terminal emulator, to access a router using an analog modem.

B

backbone area In OSPFv2 and OSPFv3, the special area in a multiarea design, with all nonbackbone areas needing to connect to the backbone area, area 0.

back-to-back link A serial link between two routers, created without CSU/DSUs, by connecting a DTE cable to one router and a DCE cable to the other. Typically used in labs to build serial links without the expense of an actual leased line from the telco.

backup designated router An OSPF router connected to a multiaccess network that monitors the work of the designated router (DR) and takes over the work of the DR if the DR fails.

backup port With RSTP, a port role in which the port acts as a backup to one of the switch's ports acting as a designated port. If the switch's designated port fails, the switch will use the backup port to immediately take over as the designated port.

band A contiguous range of frequencies.

bandwidth A reference to the speed of a networking link. Its origins come from earlier communications technology in which the range, or width, of the frequency band dictated how fast communications could occur.

basic service set (BSS) Wireless service provided by one AP to one or more associated clients.

basic service set identifier (BSSID) A unique MAC address that is used to identify the AP that is providing a BSS.

binary mask An IPv4 subnet mask written as a 32-bit binary number.

bitwise Boolean AND A Boolean AND between two numbers of the same length in which the first bit in each number is ANDed, and then the second bit in each number, and then the third, and so on.

blocking state In STP, a port state in which no received frames are processed and the switch forwards no frames out the interface, with the exception of STP messages.

Boolean AND A math operation performed on a pair of one-digit binary numbers. The result is another one-digit binary number. 1 AND 1 yields 1; all other combinations yield a 0.

BPDU Bridge protocol data unit. The generic name for Spanning Tree Protocol messages.

BPDU Guard A Cisco switch feature that listens for incoming STP BPDU messages, disabling the interface if any are received. The goal is to prevent loops when a switch connects to a port expected to only have a host connected to it.

bridge ID (BID) An 8-byte identifier for bridges and switches used by STP and RSTP. It is composed of a 2-byte priority field followed by a 6-byte System ID field that is usually filled with a MAC address.

bridge protocol data unit *See* BPDU.

broadcast address Generally, any address that represents all devices, and can be used to send one message to all devices. In Ethernet, the MAC address of all binary 1s, or FFFF.FFFF.FFFF in hex. For IPv4, *see* subnet broadcast address.

broadcast domain A set of all devices that receive broadcast frames originating from any device within the set. Devices in the same VLAN are in the same broadcast domain.

broadcast frame An Ethernet frame sent to destination address FFFF.FFFF.FFFF, meaning that the frame should be delivered to all hosts on that LAN.

broadcast subnet When subnetting a Class A, B, or C network, the one subnet in each classful network for which all subnet bits have a value of binary 1. The subnet broadcast address in this subnet has the same numeric value as the classful network's networkwide broadcast address.

C

cable Internet An Internet access technology that uses a cable TV (CATV) cable, normally used for video, to send and receive data.

CAPWAP A standards-based tunneling protocol that defines communication between a lightweight AP and a wireless LAN controller.

cell The area of wireless coverage provided by an AP; also known as the basic service area.

centralized WLC deployment *See* unified WLC deployment.

certificate authority (CA) A trusted entity that generates and signs digital certificates.

channel An arbitrary index that points to a specific frequency within a band.

Channel-group One term Cisco switches use to reference a bundle of links that are, in some respects, treated like a single link. Other similar terms include *EtherChannel* and *PortChannel*.

CIDR Classless interdomain routing. An RFC-standard tool for global IP address range assignment. CIDR reduces the size of Internet routers' IP routing tables, helping deal with the rapid growth of the Internet. The term *classless* refers to the fact that the summarized groups of networks represent a group of addresses that do not conform to IPv4 classful (Class A, B, and C) grouping rules.

CIDR mask Another term for a prefix mask, one that uses prefix or CIDR notation, in which the mask is represented by a slash (/) followed by a decimal number.

CIDR notation *See* prefix notation.

cladding In fiber-optic cabling, the second layer of the cable, surrounding the core of the cable, with the property of reflecting light back into the core.

classful addressing A concept in IPv4 addressing that defines a subnetted IP address as having three parts: network, subnet, and host.

classful IP network An IPv4 Class A, B, or C network; called a classful network because these networks are defined by the class rules for IPv4 addressing.

classful routing protocol Does not transmit the mask information along with the subnet number and therefore must consider Class A, B, and C network boundaries and perform autosummarization at those boundaries. Does not support VLSM.

classless addressing A concept in IPv4 addressing that defines a subnetted IP address as having two parts: a prefix (or subnet) and a host.

classless interdomain routing The name of an RFC that defines several important features related to public IPv4 addressing: a global address assignment strategy to keep the size of IPv4 routing tables smaller, and the ability to assign public IPv4 addresses in sizes based on any prefix length.

classless prefix A range of public IPv4 addresses as defined by CIDR.

classless prefix length The mask (prefix length) used when defining a classless prefix.

classless routing protocol An inherent characteristic of a routing protocol, specifically that the routing protocol does send subnet masks in its routing updates, thereby removing any need to make assumptions about the addresses in a particular subnet or network, making it able to support VLSM and manual route summarization.

CLI Command-line interface. An interface that enables the user to interact with the operating system by entering commands and optional arguments.

clock rate The speed at which a serial link encodes bits on the transmission medium.

clock source The device to which the other devices on the link adjust their speed when using synchronous links.

clocking The process of supplying a signal over a cable, either on a separate pin on a serial cable or as part of the signal transitions in the transmitted signal so that the receiving device can keep synchronization with the sending device.

cloud-based AP A wireless AP operating much like an autonomous AP, but having management and control functions present in the Internet cloud.

cloud-based WLC deployment A wireless network design that places a WLC centrally within a network topology, as a virtual machine in the private cloud portion of a data center.

collapsed core design A campus LAN design in which the design does not use a separate set of core switches in addition to the distribution switches—in effect collapsing the core into the distribution switches.

collision domain A set of network interface cards (NIC) for which a frame sent by one NIC could result in a collision with a frame sent by any other NIC in the same collision domain.

command-line interface *See* CLI.

configuration mode A part of the Cisco IOS Software CLI in which the user can type configuration commands that are then added to the device's currently used configuration file (running-config).

connected The single-item status code listed by a **switch show interfaces status** command, with this status referring to a working interface.

connected route On a router, an IP route added to the routing table when the router interface is both up and has an IP address configured. The route is for the subnet that can be calculated based on the configured IP address and mask.

console port A physical socket on a router or switch to which a cable can be connected between a computer and the router/switch, for the purpose of allowing the computer to use a terminal emulator and use the CLI to configure, verify, and troubleshoot the router/switch.

contiguous network A network topology in which subnets of network X are not separated by subnets of any other classful network.

convergence The time required for routing protocols to react to changes in the network, removing bad routes and adding new, better routes so that the current best routes are in all the routers' routing tables.

core In fiber-optic cabling, the center cylinder of the cable, made of fiberglass, through which light passes.

core design A campus LAN design that connects each access switch to distribution switches, and distribution switches into core switches, to provide a path between all LAN devices.

Counter/CBC-MAC Protocol (CCMP) A wireless security scheme based on 802.11i that uses AES counter mode for encryption and CBC-MAC for data integrity

crossover cable An Ethernet cable that swaps the pair used for transmission on one device to a pair used for receiving on the device on the opposite end of the cable. In 10BASE-T and 100BASE-TX networks, this cable swaps the pair at pins 1,2 to pins 3,6 on the other end of the cable, and the pair at pins 3,6 to pins 1,2 as well.

CSMA/CD Carrier sense multiple access with collision detection. A media-access mechanism in which devices ready to transmit data first check the channel for a carrier. If no carrier is sensed for a specific period of time, a device can transmit. If two devices transmit at once, a collision occurs and is detected by all colliding devices. This collision subsequently delays retransmissions from those devices for some random length of time.

CSU/DSU Channel service unit/data service unit. A device that understands the Layer 1 details of serial links installed by a telco and how to use a serial cable to communicate with networking equipment such as routers.

D

data VLAN A VLAN used by typical data devices connected to an Ethernet, like PCs and servers. Used in comparison to a voice VLAN.

Database Description An OSPF packet type that lists brief descriptions of the LSAs in the OSPF LSDB.

DCE Data communications equipment. From a physical layer perspective, the device providing the clocking on a WAN link, typically a CSU/DSU, is the DCE. From a packet-switching perspective, the service provider's switch, to which a router might connect, is considered the DCE.

DDN *See* dotted-decimal notation.

Dead Interval In OSPF, a timer used for each neighbor. A router considers the neighbor to have failed if no Hellos are received from that neighbor in the time defined by the timer.

decimal mask An IPv4 subnet mask written in dotted-decimal notation; for example, 255.255.255.0.

de-encapsulation On a computer that receives data over a network, the process in which the device interprets the lower-layer headers and, when finished with each header, removes the header, revealing the next-higher-layer PDU.

default gateway/default router On an IP host, the IP address of some router to which the host sends packets when the packet's destination address is on a subnet other than the local subnet.

default mask The mask used in a Class A, B, or C network that does not create any subnets; specifically, mask 255.0.0.0 for Class A networks, 255.255.0.0 for Class B networks, and 255.255.255.0 for Class C networks.

default route On a router, the route that is considered to match all packets that are not otherwise matched by some more specific route.

default VLAN A reference to the default setting of 1 (meaning VLAN ID 1) on the **switchport access vlan** *vlan-id* **interface** subcommand on Cisco switches, meaning that by default, a port will be assigned to VLAN 1 if acting as an access port.

designated port In both STP and RSTP, a port role used to determine which of multiple interfaces on multiple switches, each connected to the same segment or collision domain, should forward frames to the segment. The switch advertising the lowest-cost Hello BPDU onto the segment becomes the DP.

designated router In OSPF, on a multiaccess network, the router that wins an election and is therefore responsible for managing a streamlined process for exchanging OSPF topology information between all routers attached to that network.

DHCP Dynamic Host Configuration Protocol. A protocol used by hosts to dynamically discover and lease an IP address, and learn the correct subnet mask, default gateway, and DNS server IP addresses.

DHCP client Any device that uses DHCP protocols to ask to lease an IP address from a DHCP server, or to learn any IP settings from that server.

Dijkstra Shortest Path First (SPF) algorithm The name of the algorithm used by link-state routing protocols to analyze the LSDB and find the least-cost routes from that router to each subnet.

directed broadcast address *See* subnet broadcast address.

disabled port In STP, a port role for nonworking interfaces—in other words, interfaces that are not in a connect or up/up interface state.

discarding state An RSTP interface state in which no received frames are processed and the switch forwards no frames out the interface, with the exception of RSTP messages.

discontiguous network A network topology in which subnets of network X are separated by subnets of some other classful network.

distance vector The logic behind the behavior of some interior routing protocols, such as RIP. Distance vector routing algorithms call for each router to send its entire routing table in each update, but only to its neighbors. Distance vector routing algorithms can be prone to routing loops but are computationally simpler than link-state routing algorithms.

distribution layer In a campus LAN design, the switches that connect to access layer switches as the most efficient means to provide connectivity from the access layer into the other parts of the LAN.

distribution system (DS) The wired Ethernet that connects to an AP and transports traffic between a wired and wireless network.

DNS Domain Name System. An application layer protocol used throughout the Internet for translating hostnames into their associated IP addresses.

DNS Reply In the Domain Name System (DNS), a message sent by a DNS server to a DNS client in response to a DNS Request, identifying the IP address assigned to a particular hostname or fully qualified domain name (FQDN).

DNS Request In the Domain Name System (DNS), a message sent by a DNS client to a DNS server, listing a hostname or fully qualified domain name (FQDN), asking the server to discover and reply with the IP address associated with that hostname or FQDN.

dotted-decimal notation (DDN) The format used for IP version 4 addresses, in which four decimal values are used, separated by periods (dots).

DSL Digital subscriber line. Public network technology that delivers high bandwidth over conventional telco local-loop copper wiring at limited distances. Typically used as an Internet access technology, connecting a user to an ISP.

DSL modem A device that connects to a telephone line, using DSL standards, to transmit and receive data to/from a telco using DSL.

DTE Data terminal equipment. From a Layer 1 perspective, the DTE synchronizes its clock based on the clock sent by the DCE. From a packet-switching perspective, the DTE is the device outside the service provider's network, typically a router.

dual stack A mode of operation in which a host or router runs both IPv4 and IPv6.

duplex mismatch On opposite ends of any Ethernet link, the condition in which one of the two devices uses full-duplex logic and the other uses half-duplex logic, resulting in unnecessary frame discards and retransmissions on the link.

duplicate address detection (DAD) A term used in IPv6 to refer to how hosts first check whether another host is using a unicast address before the first host uses that address.

E

EAP Flexible Authentication by Secure Tunneling (EAP-FAST) A Cisco authentication method that is based on EAP and uses a PAC as a credential for outer authentication and a TLS tunnel for inner authentication

EAP Transport Layer Security (EAP-TLS) An authentication method that uses digital certificates on both the server and the supplicant for mutual authentication. A TLS tunnel is used during client authentication and key exchanges.

EIGRP Enhanced Interior Gateway Routing Protocol. An advanced version of IGRP developed by Cisco. Provides superior convergence properties and operating efficiency and combines the advantages of link-state protocols with those of distance vector protocols.

EIGRP version 6 The version of the EIGRP routing protocol that supports IPv6, and not IPv4.

electromagnetic interference (EMI) The name of the effect in which electricity passes through one cable as normal, inducing a magnetic field outside the conductor. That magnetic field, if it passes through another conductor, like a nearby cable, induces new electrical current in the second cable, interfering with the use of electricity to transmit data on the second cable.

embedded WLC deployment A wireless network design that places a WLC in the access layer, co-located with a LAN switch stack, near the APs it controls.

enable mode A part of the Cisco IOS CLI in which the user can use the most powerful and potentially disruptive commands on a router or switch, including the ability to then reach configuration mode and reconfigure the router.

encapsulation The placement of data from a higher-layer protocol behind the header (and in some cases, between a header and trailer) of the next-lower-layer protocol. For example, an IP packet could be encapsulated in an Ethernet header and trailer before being sent over an Ethernet.

encryption Applying a specific algorithm to data to alter the appearance of the data, making it incomprehensible to those who are not authorized to see the information.

enterprise mode 802.1x EAP-based authentication requirement for WPA, WPA2, and WPA3.

enterprise router A term to describe the general role of a router as a router at a permanent site owned or leased by the enterprise, like an office building, manufacturing facility, branch office, or retail location. These sites typically have enough users to justify separate routers, switches, and wireless access points, and are more likely to justify private WAN services, in comparison to SOHO routers.

error detection The process of discovering whether a data-link level frame was changed during transmission. This process typically uses a Frame Check Sequence (FCS) field in the data-link trailer.

error disabled An interface state on LAN switches that can be the result of one of many security violations.

error recovery The process of noticing when some transmitted data was not successfully received and resending the data until it is successfully received.

EtherChannel A feature in which up to eight parallel Ethernet segments exist between the same two devices, each using the same speed. May be a Layer 2 EtherChannel, which acts like a single link for forwarding and Spanning Tree Protocol logic, or a Layer 3 EtherChannel, which acts like a single link for the switch's Layer 3 routing logic.

EtherChannel Load Distribution The logic used by switches when forwarding messages over EtherChannels by which the switch chooses the specific physical link out which the switch will forward the frame.

Ethernet A series of LAN standards defined by the IEEE, originally invented by Xerox Corporation and developed jointly by Xerox, Intel, and Digital Equipment Corporation.

Ethernet address A 48-bit (6-byte) binary number, usually written as a 12-digit hexadecimal number, used to identify Ethernet nodes in an Ethernet network. Ethernet frame headers list a destination and source address field, used by the Ethernet devices to deliver Ethernet frames to the correct destination.

Ethernet frame A term referring to an Ethernet data-link header and trailer, plus the data encapsulated between the header and trailer.

Ethernet Line Service (E-Line) A specific carrier/metro Ethernet service defined by MEF (MEF.net) that provides a point-to-point topology between two customer devices, much as if the two devices were connected using an Ethernet crossover cable.

Ethernet link A generic term for any physical link between two Ethernet nodes, no matter what type of cabling is used.

Ethernet over MPLS (EoMPLS) A term referring specifically to how a service provider can create an Ethernet WAN service using an MPLS network. More generally, a term referring to Ethernet WAN services.

Ethernet port A generic term for the opening on the side of any Ethernet node, typically in an Ethernet NIC or LAN switch, into which an Ethernet cable can be connected.

EtherType Jargon that shortens the term *Ethernet Type*, which refers to the Type field in the Ethernet header. The Type field identifies the type of packet encapsulated inside an Ethernet frame.

EUI-64 Literally, a standard for an extended unique identifier that is 64 bits long. Specifically for IPv6, a set of rules for forming a 64-bit identifier, used as the interface ID in IPv6 addresses, by starting with a 48-bit MAC address, inserting FFFE (hex) in the middle, and inverting the seventh bit.

extended ping An IOS command in which the **ping** command accepts many other options besides just the destination IP address.

extended service set (ESS) Multiple APs that are connected by a common switched infrastructure.

Extensible Authentication Protocol (EAP) A standardized authentication framework that is used by a variety of authentication methods

F

Fast Ethernet The common name for all the IEEE standards that send data at 100 megabits per second.

fiber-optic cable A type of cabling that uses glass fiber as a medium through which to transmit light.

filter Generally, a process or a device that screens network traffic for certain characteristics, such as source address, destination address, or protocol, and determines whether to forward or discard that traffic based on the established criteria.

firewall A device that forwards packets between the less secure and more secure parts of the network, applying rules that determine which packets are allowed to pass and which are not.

flash memory A type of read/write permanent memory that retains its contents even with no power applied to the memory, and uses no moving parts, making the memory less likely to fail over time.

floating static route A static IP route that uses a higher administrative distance than other routes, typically routes learned by a routing protocol. As a result, the router will not use the static route if the routing protocol route has been learned, but then use the static route if the routing protocol fails to learn the route.

flood/flooding The result of the LAN switch forwarding process for broadcasts and unknown unicast frames. Switches forward these frames out all interfaces, except the interface in which the frame arrived. Switches also flood multicasts by default, although this behavior can be changed.

forward To send a frame received in one interface out another interface, toward its ultimate destination.

forward delay An STP timer, defaulting to 15 seconds, used to dictate how long an interface stays in the listening state and the time spent in learning state. Also called the forward delay timer.

forward route From one host's perspective, the route over which a packet travels from that host to some other host.

forward secrecy A key exchange method used in WPA3 that prevents attackers from being able to use a discovered pre-shared key to unencrypt data that has already been transmitted over the air

forwarding state An STP and RSTP port state in which an interface operates unrestricted by STP.

frame A term referring to a data-link header and trailer, plus the data encapsulated between the header and trailer.

Frame Check Sequence A field in many data-link trailers used as part of the error-detection process.

full duplex Generically, any communication in which two communicating devices can concurrently send and receive data. In Ethernet LANs, the allowance for both devices to send and receive at the same time, allowed when both devices disable their CSMA/CD logic.

full state In OSPF, a neighbor state that implies that the two routers have exchanged the complete (full) contents of their respective LSDBs.

full update With IP routing protocols, the general concept that a routing protocol update lists all known routes.

fully adjacent In OSPF, a characterization of the state of a neighbor in which the two neighbors have reached the full state.

fully adjacent neighbor In OSPF, a neighbor with which the local router has also reached the OSPF full state, meaning that the two routers have exchanged their LSDBs directly with each other.

G

Galois/Counter Mode Protocol (GCMP) A strong encryption method used in the WPA3 wireless security model.

Gigabit Ethernet The common name for all the IEEE standards that send data at 1 gigabit per second.

global routing prefix An IPv6 prefix that defines an IPv6 address block made up of global unicast addresses, assigned to one organization, so that the organization has a block of globally unique IPv6 addresses to use in its network.

global unicast address A type of unicast IPv6 address that has been allocated from a range of public globally unique IP addresses, as registered through IANA/ICANN, its member agencies, and other registries or ISPs.

H

half duplex Generically, any communication in which only one device at a time can send data. In Ethernet LANs, the normal result of the CSMA/CD algorithm that enforces the rule that only one device should send at any point in time.

HDLC High-Level Data Link Control. A bit-oriented synchronous data-link layer protocol developed by the International Organization for Standardization (ISO).

header In computer networking, a set of bytes placed in front of some other data, encapsulating that data, as defined by a particular protocol.

Hello (Multiple definitions) 1) A protocol used by OSPF routers to discover, establish, and maintain neighbor relationships. 2) A protocol used by EIGRP routers to discover, establish, and maintain neighbor relationships. 3) In STP, refers to the name of the periodic message sourced by the root bridge in a spanning tree.

Hello BPDU The STP and RSTP message used for the majority of STP communications, listing the root's bridge ID, the sending device's bridge ID, and the sending device's cost with which to reach the root.

Hello Interval With OSPF and EIGRP, an interface timer that dictates how often the router should send Hello messages.

Hello timer In STP, the time interval at which the root switch should send Hello BPDUs.

history buffer In a Cisco router or switch, the function by which IOS keeps a list of commands that the user has used in this login session, both in EXEC mode and configuration mode. The user can then recall these commands for easier repeating or making small edits and issuing similar commands.

hop count The metric used by the RIP routing protocol. Each router in an IP route is considered a hop, so for example, if two other routers sit between a router and some subnet, that router would have a hop count of two for that route.

host Any device that uses an IP address.

host address The IP address assigned to a network card on a computer.

host part A term used to describe a part of an IPv4 address that is used to uniquely identify a host inside a subnet. The host part is identified by the bits of value 0 in the subnet mask.

host route A route with a /32 mask, which by virtue of this mask represents a route to a single host IP address.

hostname The alphanumeric name of an IP host.

hub A LAN device that provides a centralized connection point for LAN cabling, repeating any received electrical signal out all other ports, thereby creating a logical bus. Hubs do not interpret the electrical signals as a frame of bits, so hubs are considered to be Layer 1 devices.

I

IANA The Internet Assigned Numbers Authority (IANA). An organization that owns the rights to assign many operating numbers and facts about how the global Internet works, including public IPv4 and IPv6 addresses. *See also* ICANN.

ICANN The Internet Corporation for Assigned Names and Numbers. An organization appointed by IANA to oversee the distributed process of assigning public IPv4 and IPv6 addresses across the globe.

ICMP Internet Control Message Protocol. A TCP/IP network layer protocol that reports errors and provides other information relevant to IP packet processing.

ICMP echo reply One type of ICMP message, created specifically to be used as the message sent by the ping command to test connectivity in a network. The ping command expects to receive these messages from other hosts, after the ping command first sends an ICMP echo request message to the host.

ICMP echo request One type of ICMP message, created specifically to be used as the message sent by the ping command to test connectivity in a network. The ping command sends these messages to other hosts, expecting the other host to reply with an ICMP echo reply message.

IEEE Institute of Electrical and Electronics Engineers. A professional organization that develops communications and network standards, among other activities.

IEEE 802.1 AD The IEEE standard for the functional equivalent of the Cisco-proprietary EtherChannel.

IEEE 802.11 The IEEE base standard for wireless LANs.

IEEE 802.1Q The IEEE standard VLAN trunking protocol. 802.1Q includes the concept of a native VLAN, for which no VLAN header is added, and a 4-byte VLAN header is inserted after the original frame's Type/Length field.

IEEE 802.2 An IEEE LAN protocol that specifies an implementation of the LLC sublayer of the data-link layer.

IEEE 802.3 A set of IEEE LAN protocols that specifies the many variations of what is known today as an Ethernet LAN.

IEEE 802.3 AD The IEEE standard for the functional equivalent of the Cisco-proprietary EtherChannel.

IETF The Internet Engineering Task Force. The IETF serves as the primary organization that works directly to create new TCP/IP standards.

IGP *See* interior gateway protocol.

inactivity timer For switch MAC address tables, a timer associated with each entry that counts time upward from 0 and is reset to 0 each time a switch receives a frame with the same MAC address. The entries with the largest timers can be removed to make space for additional MAC address table entries.

independent basic service set (IBSS) An impromptu wireless network formed between two or more devices without an AP or a BSS; also known as an ad hoc network.

infrastructure mode The operating mode of an AP that is providing a BSS for wireless clients.

Integrated Services Router (ISR) Cisco's long-running term for several different model series of Enterprise-class routers, intended mostly for use as enterprise routers and some use as SOHO routers. ISR routers first serve as routers but, depending on the family or specific model, support all current types of WAN connections (private and Internet), LAN switching ports, Wireless APs, VPNs, and other integrated functions supported in a single device.

interface bandwidth In OSPF, the numerator in the calculation of an interface's default OSPF cost metric, calculated as the interface bandwidth divided by the reference bandwidth.

interface-local scope A concept in IPv6 for which packets sent to an address using this scope should not physically exit the interface, keeping the packet inside the sending host.

interior gateway protocol (IGP) A routing protocol designed to be used to exchange routing information inside a single autonomous system.

interior routing protocol A synonym of interior gateway protocol. *See* interior gateway protocol.

Internal Border Gateway Protocol (iBGP) The use of BGP between two routers in the same ASN, with different rules compared to External BGP (eBGP).

internal router In OSPF, a router with all interfaces in the same nonbackbone area.

Internetwork Operating System The operating system (OS) of Cisco routers and switches, which provides the majority of a router's or switch's features, with the hardware providing the remaining features.

Inter-Switch Link (ISL) The Cisco-proprietary VLAN trunking protocol that predated 802.IQ by many years. ISL defines a 26-byte header that encapsulates the original Ethernet frame.

IOS *See* Internetwork Operating System.

IP Internet Protocol. The network layer protocol in the TCP/IP stack, providing routing and logical addressing standards and services.

IP address (IP version 4) In IP version 4 (IPv4), a 32-bit address assigned to hosts using TCP/IP. Each address consists of a network number, an optional subnetwork number, and a host number. The network and subnetwork numbers together are used for routing, and the host number is used to address an individual host within the network or subnetwork.

IP address (IP version 6) In IP version 6 (IPv6), a 128-bit address assigned to hosts using TCP/IP. Addresses use different formats, commonly using a routing prefix, subnet, and interface ID, corresponding to the IPv4 network, subnet, and host parts of an address.

IP network *See* classful IP network.

IP packet An IP header, followed by the data encapsulated after the IP header, but specifically not including any headers and trailers for layers below the network layer.

IP routing table *See* routing table.

IP subnet Subdivisions of a Class A, B, or C network, as configured by a network administrator. Subnets allow a single Class A, B, or C network to be used instead of multiple networks, and still allow for a large number of groups of IP addresses, as is required for efficient IP routing.

IP version 4 Literally, the version of the Internet Protocol defined in an old RFC 791, standardized in 1980, and used as the basis of TCP/IP networks and the Internet for over 30 years.

IP version 6 A newer version of the Internet Protocol defined in RFC 2460, as well as many other RFCs, whose creation was motivated by the need to avoid the IPv4 address exhaustion problem.

IPv4 *See* IP version 4.

IPv4 address exhaustion The process by which the public IPv4 addresses, available to create the Internet, were consumed through the 1980s until today, with the expectation that eventually the world would run out of available IPv4 addresses.

IPv6 *See* IP version 6.

IPv6 address scope The concept of how far an IPv6 packet should be forwarded by hosts and routers in an IPv6 network. Includes interface-local, link-local, site-local, and organization-local scopes.

IPv6 administrative distance In Cisco routers, a means for one router to choose between multiple IPv6 routes to reach the same subnet when those routes were learned by different routing protocols. The lower the administrative distance, the better the source of the routing information.

IPv6 host route A route with a /128 mask, which by virtue of this mask represents a route to a single host IPv6 address.

IPv6 local route A route added to an IPv6 router's routing table for the router's interface IP address, with a /128 mask, which by virtue of this mask represents a route to only that router's IPv4 address.

IPv6 multicast scope The idea of how far away from the sending host an IPv6 multicast packet should be forwarded, as based on the value in the 4th hex digit of the multicast address.

IPv6 neighbor table The IPv6 equivalent of the ARP table. A table that lists IPv6 addresses of other hosts on the same link, along with their matching MAC addresses, as typically learned using Neighbor Discovery Protocol (NDP).

ISL Inter-Switch Link. A Cisco-proprietary protocol that maintains VLAN information as traffic flows between switches and routers.

ISO International Organization for Standardization. An international organization that is responsible for a wide range of standards, including many standards relevant to networking. The ISO developed the OSI reference model, a popular networking reference model.

K–L

keepalive A proprietary feature of Cisco routers in which the router sends messages on a periodic basis as a means of letting the neighboring router know that the first router is still alive and well.

known unicast frame An Ethernet frame whose destination MAC address is listed in a switch's MAC address table, so the switch will forward the frame out the one port associated with that entry in the MAC address table.

L2PDU Layer 2 protocol data unit. Often called a frame. The data compiled by a Layer 2 protocol, including Layer 2 header, encapsulated high-layer data, and Layer 2 trailer.

L3PDU Layer 3 protocol data unit. Often called a packet. The data compiled by a Layer 3 protocol, including Layer 3 headers and the encapsulated high-layer data, but not including lower-layer headers and trailers.

L4PDU Layer 4 protocol data unit. Often called a segment. The data compiled by a Layer 4 protocol, including Layer 4 headers and encapsulated high-layer data, but not including lower-layer headers and trailers.

LACP Link Aggregation Control Protocol is a messaging protocol defined by the IEEE 802.3ad standard that enables two neighboring devices to realize that they have multiple parallel links connecting to each other and then to decide which links can be combined into an EtherChannel.

Layer 2 EtherChannel (L2 EtherChannel) An EtherChannel that acts as a switched port (that is, not a routed port), and as such, is used by a switch's Layer 2 forwarding logic. As a result, the Layer 2 switch lists the Layer 2 EtherChannel in switch MAC address tables, and when forwarding a frame based on one of these MAC table entries, the switch balances traffic across the various ports in the Layer 2 EtherChannel.

Layer 3 EtherChannel (L3 EtherChannel) An EtherChannel that acts as a routed port (that is, not a switched port), and as such, is used by a switch's Layer 3 forwarding logic. As a result, the Layer 3 switch lists the Layer 3 EtherChannel in various routes in the switch's IP routing table, with the switch balancing traffic across the various ports in the Layer 3 EtherChannel.

Layer 3 protocol A protocol that has characteristics like OSI Layer 3, which defines logical addressing and routing. IPv4 and IPv6 are Layer 3 protocols.

Layer 3 switch *See* multilayer switch.

learning The process used by switches for discovering MAC addresses, and their relative location, by looking at the source MAC address of all frames received by a bridge or switch.

learning state In STP, a temporary port state in which the interface does not forward frames, but it can begin to learn MAC addresses from frames received on the interface.

leased line A serial communications circuit between two points, provided by some service provider, typically a telephone company (telco). Because the telco does not sell a physical cable between the two endpoints, instead charging a monthly fee for the ability to send bits between the two sites, the service is considered to be a leased service.

lightweight AP A wireless AP that performs real-time 802.11 functions to interface with wireless clients, while relying on a wireless LAN controller to handle all management functions.

Lightweight EAP (LEAP) A legacy Cisco proprietary wireless security method.

link state A classification of the underlying algorithm used in some routing protocols. Link-state protocols build a detailed database that lists links (subnets) and their state (up, down), from which the best routes can then be calculated.

link-local address A unicast IPv6 address that begins FE80, used on each IPv6-enabled interface, used for sending packets within the attached link by applying a link-local scope.

link-local multicast address A multicast IPv6 address that begins with FF02, with the fourth digit of 2 identifying the scope as link-local, to which devices apply a link-local scope.

link-local scope With IPv6 multicasts, a term that refers to the parts (scope) of the network to which a multicast packet can flow, with link-local referring to the fact that the packet stays on the subnet in which it originated.

link-state advertisement (LSA) In OSPF, the name of the data structure that resides inside the LSDB and describes in detail the various components in a network, including routers and links (subnets).

link-state database (LSDB) In OSPF, the data structure in RAM of a router that holds the various LSAs, with the collective LSAs representing the entire topology of the network.

Link-State Request An OSPF packet used to ask a neighboring router to send a particular LSA.

Link-State Update An OSPF packet used to send an LSA to a neighboring router.

listening state A temporary STP port state that occurs immediately when a blocking interface must be moved to a forwarding state. The switch times out MAC table entries during this state. It also ignores frames received on the interface and doesn't forward any frames out the interface.

LLC Logical Link Control. The higher of the two sublayers of the data-link layer defined by the IEEE. Synonymous with IEEE 802.2.

local broadcast IP address IPv4 address 255.255.255.255. A packet sent to this address is sent as a data-link broadcast, but only flows to hosts in the subnet into which it was originally sent. Routers do not forward these packets.

local mode The default mode of a Cisco lightweight AP that offers one or more functioning BSSs on a specific channel.

local route A route added to an IPv4 router's routing table for the router's interface IP address, with a /32 mask, which by virtue of this mask represents a route to only that router's IPv4 address.

local username A username (with matching password), configured on a router or switch. It is considered local because it exists on the router or switch, and not on a remote server.

logical address A generic reference to addresses as defined by Layer 3 protocols that do not have to be concerned with the physical details of the underlying physical media. Used mainly to contrast these addresses with data-link addresses, which are generically considered to be physical addresses because they differ based on the type of physical medium.

LSA *See* link-state advertisement.

LSDB *See* link-state database.

M

MAC Media Access Control. The lower of the two sublayers of the data-link layer defined by the IEEE. Synonymous with IEEE 802.3 for Ethernet LANs.

MAC address A standardized data-link layer address that is required for every device that connects to a LAN. Ethernet MAC addresses are 6 bytes long and are controlled by the IEEE. Also known as a hardware address, a MAC layer address, and a physical address.

MAC address table A table of forwarding information held by a Layer 2 switch, built dynamically by listening to incoming frames and used by the switch to match frames to make decisions about where to forward the frame.

MaxAge In STP, a timer that states how long a switch should wait when it no longer receives Hellos from the root switch before acting to reconverge the STP topology. Also called the MaxAge timer.

maximum paths In Cisco IOS, a reference to the number of equal cost routes (paths) to reach a single subnet that IOS will add to the IP routing table at the same time.

MD5 hash A specific mathematical algorithm intended for use in various security protocols. In the context of Cisco routers and switches, the devices store the MD5 hash of certain passwords, rather than the passwords themselves, in an effort to make the device more secure.

media access control (MAC) layer A low-level function performed as part of Layer 2; in wireless networks, this function can be divided between a wireless LAN controller and a lightweight AP to form a split-MAC architecture.

mesh network A network of APs used to cover a large area without the need for wired Ethernet cabling; client traffic is bridged from AP to AP over a backhaul network.

message integrity check (MIC) A cryptographic value computed from the contents of a data frame and used to detect tampering.

message of the day One type of login banner that can be defined on a Cisco router or switch.

metric A unit of measure used by routing protocol algorithms to determine the best route for traffic to use to reach a particular destination.

Mobility Express WLC deployment A wireless network design that places a WLC co-located with a lightweight AP.

Modified EUI-64 *See* EUI-64.

multiarea In OSPFv2 and OSPFv3, a design that uses multiple areas.

multicast IP address A class D IPv4 address. When used as a destination address in a packet, the routers collectively work to deliver copies of the one original packet to all hosts who have previously registered to receive packets sent to that particular multicast address.

multilayer switch A LAN switch that can also perform Layer 3 routing functions. The name comes from the fact that this device makes forwarding decisions based on logic from multiple OSI layers (Layers 2 and 3).

multimode fiber A type of fiber cable that works well with transmitters like LEDs that emit multiple angles of light into the core of the cable; to accommodate the multiple angles of incident, the cable has a larger core in comparison to single-mode fiber cables.

N

name resolution The process by which an IP host discovers the IP address associated with a hostname, often involving sending a DNS request to a DNS server, with the server supplying the IP address used by a host with the listed hostname.

name server A server connected to a network that resolves network names into network addresses.

NAT Network Address Translation. A mechanism for reducing the need for globally unique IP addresses. NAT allows an organization with addresses that are not globally unique to connect to the Internet, by translating those addresses into public addresses in the globally routable address space.

native VLAN The one VLAN ID on any 802.1Q VLAN trunk for which the trunk forwards frames without an 802.1Q header.

neighbor In routing protocols, another router with which a router decides to exchange routing information.

Neighbor Advertisement (NA) A message defined by the IPv6 Neighbor Discovery Protocol (NDP), used to declare to other neighbors a host's MAC address. Sometimes sent in response to a previously received NDP Neighbor Solicitation (NS) message.

Neighbor Discovery Protocol (NDP) A protocol that is part of the IPv6 protocol suite, used to discover and exchange information about devices on the same subnet (neighbors). In particular, it replaces the IPv4 ARP protocol.

Neighbor Solicitation (NS) A message defined by the IPv6 Neighbor Discovery Protocol (NDP), used to ask a neighbor to reply with a Neighbor Advertisement, which lists the neighbor's MAC address.

neighbor table For OSPF and EIGRP, a list of routers that have reached neighbor status.

network A collection of computers, printers, routers, switches, and other devices that can communicate with each other over some transmission medium.

network address *See* network number.

network broadcast address In IPv4, a special address in each classful network that can be used to broadcast a packet to all hosts in that same classful network. Numerically, the address has the same value as the network number in the network part of the address and all 255s in the host octets; for example, 10.255.255.255 is the network broadcast address for classful network 10.0.0.0.

network ID A number that identifies an IPv4 network, using a number in dotted-decimal notation (like IP addresses); a number that represents any single Class A, B, or C IP network.

network interface card (NIC) A computer card, sometimes an expansion card and sometimes integrated into the motherboard of the computer, that provides the electronics and other functions to connect to a computer network. Today, most NICs are specifically Ethernet NICs, and most have an RJ-45 port, the most common type of Ethernet port.

Network LSA In OSPF, a type of LSA that a designated router (DR) creates for the network (subnet) for which the DR is helping to distribute LSAs.

network number A number that uses dotted-decimal notation like IP addresses, but the number itself represents all hosts in a single Class A, B, or C IP network.

network part The portion of an IPv4 address that is either 1, 2, or 3 octets/bytes long, based on whether the address is in a Class A, B, or C network.

network route A route for a classful network.

networking model A generic term referring to any set of protocols and standards collected into a comprehensive grouping that, when followed by the devices in a network, allows all the devices to communicate. Examples include TCP/IP and OSI.

next-hop router In an IP route in a routing table, part of a routing table entry that refers to the next IP router (by IP address) that should receive packets that match the route.

NIC *See* network interface card.

nonoverlapping channels Successive channel numbers in a band that each have a frequency range that is narrow enough to not overlap the next channel above or below.

NVRAM Nonvolatile RAM. A type of random-access memory (RAM) that retains its contents when a unit is powered off.

O

open authentication An 802.11 authentication method that requires clients to associate with an AP without providing any credentials at all.

Organization-local scope A concept in IPv6 for which packets sent to an address using this scope should be forwarded by routers inside the organization but not over any links connected to other organizations or over links connected to the Internet.

OSI Open System Interconnection reference model. A network architectural model developed by the ISO. The model consists of seven layers, each of which specifies particular network functions, such as addressing, flow control, error control, encapsulation, and reliable message transfer.

OSPF Open Shortest Path First. A popular link-state IGP that uses a link-state database and the Shortest Path First (SPF) algorithm to calculate the best routes to reach each known subnet.

OSPF version 2 The version of the OSPF routing protocol that supports IPv4, and not IPv6, and has been commonly used for over 20 years.

OSPF version 3 The version of the OSPF routing protocol that originally supported only IPv6, and not IPv4, but now supports IPv4 through the use of address family configuration.

outgoing interface In an IP route in a routing table, part of a routing table entry that refers to the local interface out which the local router should forward packets that match the route.

overlapping subnets An (incorrect) IP subnet design condition in which one subnet's range of addresses includes addresses in the range of another subnet.

P

packet A logical grouping of bytes that includes the network layer header and encapsulated data, but specifically does not include any headers and trailers below the network layer.

PagP Port Aggregation Protocol (PAgP) is a messaging protocol defined by Cisco that enables two neighboring devices to realize that they have multiple parallel links connecting to each other and then to decide which links can be combined into an EtherChannel.

partial mesh A network topology in which more than two devices could physically communicate but, by choice, only a subset of the pairs of devices connected to the network is allowed to communicate directly.

passive interface With a routing protocol, a router interface for which the routing protocol is enabled on the interface, but for which the routing protocol does not send routing protocol messages out that interface.

patch cable An Ethernet cable, usually short, that connects from a device's Ethernet port to a wall plate or switch. With wiring inside a building, electricians prewire from the wiring closet to each cubicle or other location, with a patch cable connecting the short distance from the wall plate to the user device.

PDU Protocol data unit. An OSI term to refer generically to a grouping of information by a particular layer of the OSI model. More specifically, an LxPDU would imply the data and headers as defined by Layer x.

periodic update With routing protocols, the concept that the routing protocol advertises routes in a routing update on a regular periodic basis. This is typical of distance vector routing protocols.

personal mode Pre-shared key authentication as applied to WPA, WPA2, and WPA3.

ping An Internet Control Message Protocol (ICMP) echo message and its reply; ping often is used in IP networks to test the reachability of a network device.

pinout The documentation and implementation of which wires inside a cable connect to each pin position in any connector.

point-to-point bridge An AP configured to bridge a wired network to a companion bridge at the far end of a line-of-sight path.

port In TCP and UDP, a number that is used to uniquely identify the application process that either sent (source port) or should receive (destination port) data. In LAN switching, another term for *switch interface*.

PortChannel One term Cisco switches use to reference a bundle of links that are, in some respects, treated like a single link. Other similar terms include *EtherChannel* and *Channel-group*.

PortFast A switch STP feature in which a port is placed in an STP forwarding state as soon as the interface comes up, bypassing the listening and learning states. This feature is meant for ports connected to end-user devices.

Prefix (prefix ID) In both IPv4 and IPv6, this term refers to the number that identifies a group of IPv4 or IPv6 addresses, respectively. Another term for *subnet identifier*.

prefix length In IPv6, the number of bits in an IPv6 prefix.

prefix mask A term to describe an IPv4 subnet mask when represented as a slash (/) followed by a decimal number. The decimal number is the number of binary 1s in the mask.

prefix notation (IP version 4) A shorter way to write a subnet mask in which the number of binary 1s in the mask is simply written in decimal. For example, /24 denotes the subnet mask with 24 binary 1 bits in the subnet mask. The number of bits of value binary 1 in the mask is considered to be the prefix length.

primary root This term refers to the switch configured with the primary keyword on the **spanning-tree vlan x root {primary | secondary}** command. At time of configuration, this command causes the switch to choose a new priority setting that makes the switch become the root switch in the network.

private addresses IP addresses in several Class A, B, and C networks that are set aside for use inside private organizations. These addresses, as defined in RFC 1918, are not routable through the Internet.

private IP network Any of the IPv4 Class A, B, or C networks as defined by RFC 1918, intended for use inside a company but not used as public IP networks.

protected access credential (PAC) Special-purpose data that is used as an authentication credential in EAP-FAST.

Protected EAP (PEAP) An authentication method that uses a certificate on the AS for outer authentication and a TLS tunnel for inner authentication. Clients can provide their credentials through either MS-CHAPv2 or GTC.

Protected Management Frame (PMF) A service provided by WPA3 that protects a set of 802.11 robust management and action frames, to prevent spoofing of AP functions.

protocol data unit (PDU) A generic term referring to the header defined by some layer of a networking model, and the data encapsulated by the header (and possibly trailer) of that layer, but specifically not including any lower-layer headers and trailers.

Protocol Type field A field in a LAN header that identifies the type of header that follows the LAN header. Includes the DIX Ethernet Type field, the IEEE 802.2 DSAP field, and the SNAP protocol Type field.

public IP address An IP address that is part of a registered network number, as assigned by an Internet Assigned Numbers Authority (IANA) member agency, so that only the organization to which the address is registered is allowed to use the address. Routers in the Internet should have routes allowing them to forward packets to all the publicly registered IP addresses.

public IP network Any IPv4 Class A, B, or C network assigned for use by one organization only, so that the addresses in the network are unique across the Internet, allowing packets to be sent through the public Internet using the addresses.

Public Key Infrastructure (PKI) An enterprisewide system that generates and revokes digital certificates for client authentication.

PVST+ An STP option in Cisco switches that creates an STP instance per VLAN. Cisco proprietary.

Q–R

quartet A term used in this book, but not in other references, to refer to a set of four hex digits in an IPv6 address.

RADIUS server An authentication server used with 802.1x to authenticate wireless clients.

RAM Random-access memory. A type of volatile memory that can be read and written by a microprocessor.

Rapid PVST+ An STP option in Cisco switches that creates an RSTP instance per VLAN. Cisco proprietary.

Rapid Spanning Tree Protocol (RSTP) Defined in IEEE 802.lw. Defines an improved version of STP that converges much more quickly and consistently than STP (802.Id).

reference bandwidth In OSPF, a configurable value for the OSPF routing process, used by OSPF when calculating an interface's default OSPF cost metric, calculated as the interface's bandwidth divided by the reference bandwidth.

Regional Internet Registry An organization (five globally) that receives allocations of public IPv4 addresses from IANA and then manages that address space in their major geographic region, performing public address allocations to ISPs and assignments directly to companies that use the addresses.

repeater A device that repeats or retransmits signals it receives, effectively expanding the wireless coverage area.

resident subnet Each IP subnet contains a number of unicast IP addresses; that subnet is the resident subnet for each of those addresses—that is, the subnet in which those addresses reside.

reverse route From one host's perspective, for packets sent back to the host from another host, the route over which the packet travels.

RFC Request For Comments. A document used as the primary means for communicating information about the TCP/IP protocols. Some RFCs are designated by the Internet Architecture Board (IAB) as Internet standards, and others are informational. RFCs are available online from numerous sources, including http://www.rfc-editor.org.

RIP Routing Information Protocol. An interior gateway protocol (IGP) that uses distance vector logic and router hop count as the metric. RIP version 2 (RIPv2) replaced the older RIP version 1 (RIPv1), with RIPv2 providing more features, including support for VLSM.

RIR *See* Regional Internet Registry.

RJ-45 A popular type of cabling connector used for Ethernet cabling. It is similar to the RJ-11 connector used for telephone wiring in homes in the United States. RJ-45 allows the connection of eight wires.

roaming The process a wireless client uses to move from one AP to another as it changes location.

ROAS *See* Router-on-a-Stick.

ROM Read-only memory. A type of nonvolatile memory that can be read but not written to by the microprocessor.

ROMMON A shorter name for ROM Monitor, which is a low-level operating system that can be loaded into Cisco routers for several seldom-needed maintenance tasks, including password recovery and loading a new IOS when flash memory has been corrupted.

root bridge *See* root switch.

root cost The STP cost from a nonroot switch to reach the root switch, as the sum of all STP costs for all ports out which a frame would exit to reach the root.

root port In STP and RSTP, the one port on a nonroot switch in which the least-cost Hello is received. Switches put root ports in a forwarding state.

root switch In STP and RSTP, the switch that wins the election by virtue of having the lowest bridge ID and, as a result, sends periodic Hello BPDUs (default, 2 seconds).

routed port A port on a multilayer Cisco switch, configured with the no switchport command, that tells the switch to treat the port as if it were a Layer 3 port, like a router interface.

routed protocol A protocol that defines packets that can be routed by a router. Examples of routed protocols include IPv4 and IPv6.

Router Advertisement (RA) A message defined by the IPv6 Neighbor Discovery Protocol (NDP), used by routers to announce their willingness to act as an IPv6 router on a link. These can be sent in response to a previously received NDP Router Solicitation (RS) message.

router ID (RID) In EIGRP and OSPF, a 32-bit number, written in dotted-decimal notation, that uniquely identifies each router.

router LSA In OSPF, a type of LSA that a router creates to describe itself and the networks connected to it.

Router-on-a-Stick (ROAS) Jargon to refer to the Cisco router feature of using VLAN trunking on an Ethernet interface, which then allows the router to route packets that happen to enter the router on that trunk and then exit the router on that same trunk, just on a different VLAN.

Router Solicitation (RS) A message defined by the IPv6 Neighbor Discovery Protocol (NDP), used to ask any routers on the link to reply, identifying the router, plus other configuration settings (prefixes and prefix lengths).

routing protocol A set of messages and processes with which routers can exchange information about routes to reach subnets in a particular network. Examples of routing protocols include Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), and Routing Information Protocol (RIP).

routing table A list of routes in a router, with each route listing the destination subnet and mask, the router interface out which to forward packets destined to that subnet, and as needed, the next-hop router's IP address.

routing update A generic reference to any routing protocol's messages in which it sends routing information to a neighbor.

RSTP *See* Rapid Spanning Tree Protocol.

running-config file In Cisco IOS switches and routers, the name of the file that resides in RAM, holding the device's currently used configuration.

S

same-layer interaction The communication between two networking devices for the purposes of the functions defined at a particular layer of a networking model, with that communication happening by using a header defined by that layer of the model. The two devices set values in the header, send the header and encapsulated data, with the receiving devices interpreting the header to decide what action to take.

secondary root This term refers to the switch configured with the secondary keyword on the **spanning-tree vlan x root {primary | secondary}** command. At time of configuration, this command causes the switch to set its base priority to 28,762.

Secure Shell (SSH) A TCP/IP application layer protocol that supports terminal emulation between a client and server, using dynamic key exchange and encryption to keep the communications private.

segment In TCP, a term used to describe a TCP header and its encapsulated data (also called an L4PDU). Also in TCP, the process of accepting a large chunk of data from the application layer and breaking it into smaller pieces that fit into TCP segments. In Ethernet, a segment is either a single Ethernet cable or a single collision domain (no matter how many cables are used).

serial cable A type of cable with many different styles of connectors used to connect a router to an external CSU/DSU on a leased-line installation.

serial interface A type of interface on a router, used to connect to some types of WAN links, particularly leased lines and Frame Relay access links.

service set identifier (SSID) A text string that is used to identify a wireless network.

shared Ethernet An Ethernet that uses a hub, or even the original coaxial cabling, that results in the devices having to take turns sending data, sharing the available bandwidth.

shortest path first (SPF) algorithm The name of the algorithm used by link-state routing protocols to analyze the LSDB and find the least-cost routes from that router to each subnet.

Simultaneous Authentication of Equals (SAE) A strong authentication method used in WPA3 to authenticate wireless clients and APs and to prevent dictionary attacks for discovering pre-shared keys.

single-mode fiber A type of fiber cable that works well with transmitters like lasers that emit a single angle of light into the core of the cable, allowing for a smaller core in comparison to multimode fiber cables.

site-local scope A concept in IPv6 for which packets sent to an address using this scope should be forwarded by routers, but not forwarded over WAN links to other sites.

SOHO router A term to describe the general role of a router that exists as part of the enterprise network but resides at an employee's home or at a smaller business site, possibly with a short-term lease compared to larger enterprise sites. These sites typically have few devices, so it makes sense to use one device that integrates routing, switches, wireless, and other features into a single device (the SOHO router) and are more likely to justify Internet access as the primary WAN access method.

solicited-node multicast address A type of IPv6 multicast address, with link-local scope, used to send packets to all hosts in the subnet that share the same value in the last six hex digits of their unicast IPv6 addresses. Begins with FF02::1:FF00:0/104.

Spanning Tree Protocol (STP) A protocol defined by IEEE standard 802.ID. Allows switches and bridges to create a redundant LAN, with the protocol dynamically causing some ports to block traffic, so that the bridge/switch forwarding logic will not cause frames to loop indefinitely around the LAN.

split-MAC architecture A wireless AP strategy based around the idea that normal AP functions are split or divided between a wireless LAN controller and lightweight APs.

SSH *See* Secure Shell.

standard access list A list of IOS global configuration commands that can match only a packet's source IP address, for the purpose of deciding which packets to discard and which to allow through the router.

star topology A network topology in which endpoints on a network are connected to a common central device by point-to-point links.

startup-config file In Cisco IOS switches and routers, the name of the file that resides in NVRAM memory, holding the device's configuration that will be loaded into RAM as the running-config file when the device is next reloaded or powered on.

stateful DHCPv6 A term used in IPv6 to contrast with stateless DHCP. Stateful DHCP keeps track of which clients have been assigned which IPv6 addresses (state information).

stateless address autoconfiguration (SLAAC) A feature of IPv6 in which a host or router can be assigned an IPv6 unicast address without the need for a stateful DHCP server.

stateless DHCPv6 A term used in IPv6 to contrast with stateful DHCP. Stateless DHCP servers don't lease IPv6 addresses to clients. Instead, they supply other useful information, such as DNS server IP addresses, but with no need to track information about the clients (state information).

static access interface A LAN network design term, synonymous with the term *access interface*, but emphasizing that the port is assigned to one VLAN as a result of static configuration rather than through some dynamic process.

static route An IP route on a router created by the user configuring the details of the route on the local router.

station (STA) An 802.11 client device that is associated with a BSS.

STP Shielded twisted-pair. This type of cabling has a layer of shielded insulation to reduce electromagnetic interference (EMI).

straight-through cable In Ethernet, a cable that connects the wire on pin 1 on one end of the cable to pin 1 on the other end of the cable, pin 2 on one end to pin 2 on the other end, and so on.

subinterface One of the virtual interfaces on a single physical interface.

subnet Subdivisions of a Class A, B, or C network, as configured by a network administrator. Subnets allow a single Class A, B, or C network to be used instead of multiple networks, and still allow for a large number of groups of IP addresses, as is required for efficient IP routing.

subnet address *See* subnet number.

subnet broadcast address A special address in each IPv4 subnet, specifically the largest numeric address in the subnet, designed so that packets sent to this address should be delivered to all hosts in that subnet.

subnet ID (IPv4) *See* subnet number.

subnet ID (IPv6) The number that represents the IPv6 subnet. Also known as the IPv6 prefix, or more formally as the subnet-router anycast address.

subnet ID (prefix ID) *See* subnet number.

subnet mask A 32-bit number that numerically describes the format of an IP address, by representing the combined network and subnet bits in the address with mask bit values of 1, and representing the host bits in the address with mask bit values of 0.

subnet number In IPv4, a dotted-decimal number that represents all addresses in a single subnet. Numerically, the smallest value in the range of numbers in a subnet, reserved so that it cannot be used as a unicast IP address by a host.

subnet part In a subnetted IPv4 address, interpreted with classful addressing rules, one of three parts of the structure of an IP address, with the subnet part uniquely identifying different subnets of a classful IP network.

subnet router anycast address A special anycast address in each IPv6 subnet, reserved for use by routers as a way to send a packet to any router on the subnet. The address's value in each subnet is the same number as the subnet ID.

subnet zero An alternative term for *zero subnet*. *See* zero subnet.

subnetting The process of subdividing a Class A, B, or C network into smaller groups called subnets.

summary LSA In OSPFv2, a type of LSA, created by an Area Border Router (ABR), to describe a subnet in one area in the database of another area.

supplicant An 802.1x entity that exists as software on a client device and serves to request network access.

switch A network device that filters, forwards, and floods Ethernet frames based on the destination address of each frame.

switched Ethernet An Ethernet that uses a switch, and particularly not a hub, so that the devices connected to one switch port do not have to contend to use the bandwidth available on another port. This term contrasts with *shared Ethernet*, in which the devices must share bandwidth, whereas switched Ethernet provides much more capacity, as the devices do not have to share the available bandwidth.

switched port A port on a multilayer Cisco switch or a Layer 2 switch, configured with the normal default interface setting of switchport, that tells the switch to treat the port as if it were a Layer 2 port, resulting in the switch performing switch MAC learning, Layer 2 forwarding, and STP on that interface.

switched virtual interface (SVI) Another term for any VLAN interface in a Cisco switch. *See also* VLAN interface.

symmetric A feature of many Internet access technologies in which the downstream transmission rate is the same as the upstream transmission rate.

synchronous The imposition of time ordering on a bit stream. Practically, a device will try to use the same speed as another device on the other end of a serial link. However, by examining transitions between voltage states on the link, the device can notice slight variations in the speed on each end and can adjust its speed accordingly.

system ID extension The term for the formatting applied to the original 16-bit STP priority field to break it into a 4-bit priority field and a 12-bit VLAN ID field.

T

T1 A line from the telco that allows transmission of data at 1.544 Mbps, with the ability to treat the line as 24 different 64-kbps DS0 channels (plus 8 kbps of overhead).

TCP Transmission Control Protocol. A connection-oriented transport layer TCP/IP protocol that provides reliable data transmission.

TCP/IP Transmission Control Protocol/Internet Protocol. A common name for the suite of protocols developed by the U.S. Department of Defense in the 1970s to support the construction of worldwide internetworks. TCP and IP are the two best-known protocols in the suite.

telco A common abbreviation for *telephone company*.

Telnet The standard terminal-emulation application layer protocol in the TCP/IP protocol stack. Telnet is used for remote terminal connection, enabling users to log in to remote systems and use resources as if they were connected to a local system. Telnet is defined in RFC 854.

Temporal Key Integrity Protocol (TKIP) A wireless security scheme developed before 802.11i that provides a MIC for data integrity, a dynamic method for per-frame WEP encryption keys, and a 48-bit initialization vector. The MIC also includes a time stamp and the sender's MAC address

three-tier design *See* core design.

topology database The structured data that describes the network topology to a routing protocol. Link-state and balanced hybrid routing protocols use topology tables, from which they build the entries in the routing table.

trace Short for traceroute. A program available on many systems that traces the path that a packet takes to a destination. It is used mostly to troubleshoot routing problems between hosts.

traceroute A program available on many systems that traces the path that a packet takes to a destination. It is used mostly to debug routing problems between hosts.

trailer In computer networking, a set of bytes placed behind some other data, encapsulating that data, as defined by a particular protocol. Typically, only data-link layer protocols define trailers.

transceiver A term formed from the words *transmitter* and *receiver*. The hardware used to both send (transmit) energy over some communications medium (e.g., wires in a cable), as well as to process received energy signals to interpret as a series of 1s and 0s.

transparent bridge The name of a networking device that was a precursor to modern LAN switches. Bridges forward frames between LAN segments based on the destination MAC address. Transparent bridging is so named because the presence of bridges is transparent to network end nodes.

trunk In campus LANs, an Ethernet segment over which the devices add a VLAN header that identifies the VLAN in which the frame exists.

trunk interface A switch interface configured so that it operates using VLAN trunking (either 802.1Q or ISL).

trunking Also called *VLAN trunking*. A method (using either the Cisco ISL protocol or the IEEE 802.1Q protocol) to support multiple VLANs, allowing traffic from those VLANs to cross a single link.

trunking administrative mode The configured trunking setting on a Cisco switch interface, as configured with the switchport mode command.

trunking operational mode The current behavior of a Cisco switch interface for VLAN trunking.

twisted-pair Transmission medium consisting of two insulated wires, with the wires twisted around each other in a spiral. An electrical circuit flows over the wire pair, with the current in opposite directions on each wire, which significantly reduces the interference between the two wires.

two-tier design *See* collapsed core design.

U

UDP User Datagram Protocol. Connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery.

unicast address Generally, any address in networking that represents a single device or interface, instead of a group of addresses (as would be represented by a multicast or broadcast address).

unicast IP address An IP address that represents a single interface. In IPv4, these addresses come from the Class A, B, and C ranges.

unified WLC deployment A wireless network design that places a WLC centrally within a network topology.

unique local address A type of IPv6 unicast address meant as a replacement for IPv4 private addresses.

unknown unicast frame An Ethernet frame whose destination MAC address is not listed in a switch's MAC address table, so the switch must flood the frame.

up and up Jargon referring to the two interface states on a Cisco IOS router or switch (line status and protocol status), with the first "up" referring to the line status and the second "up" referring to the protocol status. An interface in this state should be able to pass data-link frames.

update timer The time interval that regulates how often a routing protocol sends its next periodic routing updates. Distance vector routing protocols send full routing updates every update interval.

user mode A mode of the user interface to a router or switch in which the user can type only nondisruptive EXEC commands, generally just to look at the current status, but not to change any operational settings.

UTP Unshielded twisted-pair. A type of cabling, standardized by the Telecommunications Industry Association (TIA), that holds twisted pairs of copper wires (typically four pair) and does not contain any shielding from outside interference.

V

variable-length subnet mask (VLSM) The capability to specify a different subnet mask for the same Class A, B, or C network number on different subnets. VLSM can help optimize available address space.

virtual LAN (VLAN) A group of devices, connected to one or more switches, with the devices grouped into a single broadcast domain through switch configuration. VLANs allow switch administrators to separate the devices connected to the switches into separate VLANs without requiring separate physical switches, gaining design advantages of separating the traffic without the expense of buying additional hardware.

virtual private network (VPN) The process of securing communication between two devices whose packets pass over some public and unsecured network, typically the Internet. VPNs encrypt packets so that the communication is private, and authenticate the identity of the endpoints.

VLAN *See* virtual LAN.

VLAN configuration database The name of the collective configuration of VLAN IDs and names on a Cisco switch.

VLAN interface A configuration concept inside Cisco switches, used as an interface between IOS running on the switch and a VLAN supported inside the switch, so that the switch can assign an IP address and send IP packets into that VLAN.

VLAN Trunking Protocol (VTP) A Cisco-proprietary messaging protocol used between Cisco switches to communicate configuration information about the existence of VLANs, including the VLAN ID and VLAN name.

voice VLAN A VLAN defined for use by IP Phones, with the Cisco switch notifying the phone about the voice VLAN ID so that the phone can use 802.1Q frames to support traffic for the phone and the attached PC (which uses a data VLAN).

VoIP Voice over IP. The transport of voice traffic inside IP packets over an IP network.

VTP *See* VLAN Trunking Protocol.

VTP client mode One of three VTP operational modes for a switch with which switches learn about VLAN numbers and names from other switches, but which does not allow the switch to be directly configured with VLAN information.

VTP server mode One of three VTP operational modes. Switches in server mode can configure VLANs, tell other switches about the changes, and learn about VLAN changes from other switches.

VTP transparent mode One of three VTP operational modes. Switches in transparent mode can configure VLANs, but they do not tell other switches about the changes, and they do not learn about VLAN changes from other switches.

W

WAN *See* wide-area network.

web server Software, running on a computer, that stores web pages and sends those web pages to web clients (web browsers) that request the web pages.

wide-area network (WAN) A part of a larger network that implements mostly OSI Layer 1 and 2 technology, connects sites that typically sit far apart, and uses a business model in which a consumer (individual or business) must lease the WAN from a service provider (often a telco).

Wi-Fi Alliance An organization formed by many companies in the wireless industry (an industry association) for the purpose of getting multivendor certified-compatible wireless products to market in a more timely fashion than would be possible by simply relying on standardization processes.

Wi-Fi Protected Access (WPA) The first version of a Wi-Fi Alliance standard that requires pre-shared key or 802.1x authentication, TKIP, and dynamic key management; based on parts of the 802.11i amendment before it was ratified.

wildcard mask The mask used in Cisco IOS ACL commands and OSPF and EIGRP network commands.

window Represents the number of bytes that can be sent without receiving an acknowledgment.

Wired Equivalent Privacy (WEP) An 802.11 authentication and encryption method that requires clients and APs to use a common WEP key.

wired LAN A local-area network (LAN) that physically transmits bits using cables, often the wires inside cables. A term for local-area networks that use cables, emphasizing the fact that the LAN transmits data using wires (in cables) instead of wireless radio waves. *See also* wireless LAN.

wireless LAN A local-area network (LAN) that physically transmits bits using radio waves. The name "wireless" compares these LANs to more traditional "wired" LANs, which are LANs that use cables (which often have copper wires inside).

wireless LAN Controller (WLC) A device that cooperates with wireless lightweight access points (LWAP) to create a wireless LAN by performing some control functions for each LWAP and forwarding data between each LWAP and the wired LAN.

WLAN client A wireless device that wants to gain access to a wireless access point for the purpose of communicating with other wireless devices or other devices connected to the wired internetwork.

workgroup bridge (WGB) An AP that is configured to bridge between a wired device and a wireless network. The WGB acts as a wireless client.

WPA Version 2 (WPA2) The second version of a Wi-Fi Alliance standard that requires pre-shared key or 802.1x authentication, TKIP or CCMP, and dynamic encryption key management; based on the complete 802.11i amendment after its ratification.

WPA Version 3 (WPA3) The third version of a Wi-Fi Alliance standard introduced in 2018 that requires pre-shared key or 802.1x authentication, GCMP, SAE, and forward secrecy.

Z

zero subnet For every classful IPv4 network that is subnetted, the one subnet whose subnet number has all binary 0s in the subnet part of the number. In decimal, the zero subnet can be easily identified because it is the same number as the classful network number.

Index

Symbols

Numbers

A

B

C

D

E

F

G

H

J - K

L

M

N

O

P

Q - R

S

T

U

V

W - X - Y - Z

REGISTER YOUR PRODUCT at CiscoPress.com/register

Access Additional Benefits and SAVE 35% on Your Next Purchase

- Download available product updates.
- Access bonus material when applicable.
- Receive exclusive offers on new editions and related products. (Just check the box to hear from us when setting up your account.)
- Get a coupon for 35% for your next purchase, valid for 30 days. Your code will be available in your Cisco Press cart. (You will also find it in the Manage Codes section of your account page.)

Registration benefits vary by product. Benefits will be listed on your account page under Registered Products.

CiscoPress.com – Learning Solutions for Self-Paced Study, Enterprise, and the Classroom

Cisco Press is the Cisco Systems authorized book publisher of Cisco networking technology, Cisco certification self-study, and Cisco Networking Academy Program materials.

At CiscoPress.com you can

- Shop our books, eBooks, software, and video training.
- Take advantage of our special offers and promotions (ciscopress.com/promotions).
- Sign up for special offers and content newsletters (ciscopress.com/newsletters).
- Read free articles, exam profiles, and blogs by information technology experts.
- Access thousands of free chapters and video lessons.

Connect with Cisco Press – Visit CiscoPress.com/community

Learn about Cisco Press community events and programs.

Cisco Press

APPENDIX D

Practice for Chapter 12: Analyzing Classful IPv4 Networks

Practice Problems

The practice problems in this appendix require that you determine a few basic facts about a network, given an IP address and an assumption that subnetting is not used in that network. To do so, refer to the processes described in Chapter 12 of *CCNA 200-301 Official Cert Guide, Volume 1*.

NOTE You may also elect to do this same set of practice problems using the "Practice Exercise: Analyzing Classful IPv4 Networks" application on the companion website.

In particular, for the upcoming list of IP addresses, you should identify the following information:

- Class of the address
- Number of octets in the network part of the address
- Number of octets in the host part of the address
- Network number
- Network broadcast address

Find all these facts for the following IP addresses:

1. 10.55.44.3
2. 128.77.6.7
3. 192.168.76.54
4. 190.190.190.190
5. 9.1.1.1
6. 200.1.1.1
7. 201.1.77.5
8. 101.1.77.5
9. 119.67.99.240
10. 219.240.66.98

Answers

The process to answer these problems is relatively basic, so this section reviews the overall process and then lists the answers to problems 1–10.

The process starts by examining the first octet of the IP address:

- If the first octet of the IP address is a number between 1 and 126, inclusive, the address is a Class A address.
- If the first octet of the IP address is a number between 128 and 191, inclusive, the address is a Class B address.
- If the first octet of the IP address is a number between 192 and 223, inclusive, the address is a Class C address.

When no subnetting is used:

- Class A addresses have one octet in the network part of the address and three octets in the host part.
- Class B addresses have two octets each in the network and host part.
- Class C addresses have three octets in the network part and one octet in the host part.

After determining the class and the number of network octets, you can easily find the network number and network broadcast address. To find the network number, copy the network octets of the IP address and write down 0s for the host octets. To find the network broadcast address, copy the network octets of the IP address and write down 255s for the host octets.

Table D-1 lists all ten problems and their respective answers.

Table D-1 Answers to Problems

IP Address	Class	Number of Network Octets	Number of Host Octets	Network Number	Network Broadcast Address
10.55.44.3	A	1	3	10.0.0.0	10.255.255.255
128.77.6.7	B	2	2	128.77.0.0	128.77.255.255
192.168.76.54	C	3	1	192.168.76.0	192.168.76.255
190.190.190.190	B	2	2	190.190.0.0	190.190.255.255
9.1.1.1	A	1	3	9.0.0.0	9.255.255.255
200.1.1.1	C	3	1	200.1.1.0	200.1.1.255
201.1.77.55	C	3	1	201.1.77.0	201.1.77.255
101.1.77.55	A	1	3	101.0.0.0	101.255.255.255
119.67.99.240	A	1	3	119.0.0.0	119.255.255.255
219.240.66.98	C	3	1	219.240.66.0	219.240.66.255

APPENDIX E

Practice for Chapter 13: Analyzing Subnet Masks

This appendix begins with 23 mask conversion problems, followed by the matching answers and explanations. After that, the appendix lists 10 mask analysis problems, with the matching answers to follow.

NOTE You may also perform this same set of practice problems using the "Analyzing Subnet Masks" and "Mask Conversion" applications on the companion website.

Mask Conversion Problems

The problems in this appendix require you to convert dotted-decimal subnet masks to prefix format and vice versa. To do so, feel free to use the processes described in Chapter 13 of *CCNA 200-301 Official Cert Guide, Volume 1*.

Many people use the information in Table E-1 when converting masks. The table lists the nine dotted-decimal notation (DDN) mask values, the binary equivalent, and the number of binary 1s in the binary equivalent.

Table E-1 Nine Possible Values in One Octet of a Subnet Mask

Binary Mask Octet	DDN Mask Octet	Number of Binary 1s
00000000	0	0
10000000	128	1
11000000	192	2
11100000	224	3
11110000	240	4
11111000	248	5
11111100	252	6
11111110	254	7
11111111	255	8

Convert each DDN mask to prefix format and vice versa:

1. 255.240.0.0
2. 255.255.192.0
3. 255.255.255.224
4. 255.254.0.0.

5. 255.255.248.0
6. /30
7. /25
8. /11
9. /22
10. /24
11. 255.0.0.0
12. /29
13. /9
14. 255.192.0.0
15. 255.255.255.240
16. /26
17. /13
18. 255.255.254.0
19. 255.252.0.0
20. /20
21. /16
22. 255.255.224.0
23. 255.255.128.0

Answers to Mask Conversion Problems

Mask Conversion Problem 1: Answer

The answer is /12.

The binary process for converting the mask from dotted-decimal format to prefix format is relatively simple. The only hard part is converting the dotted-decimal number to binary. For reference, the process is as follows:

Step 1. Convert the dotted-decimal mask to binary.

Step 2. Count the number of binary 1s in the 32-bit binary mask; this is the value of the prefix notation mask.

For problem 1, mask 255.240.0.0 converts to the following:

11111111 11110000 00000000 00000000

You can see from the binary number that it contains 12 binary 1s, so the prefix format of the mask will be /12.

You can find the same answer without converting decimal to binary if you have memorized the nine DDN mask values, and the corresponding number of binary 1s in each, as listed earlier in Table E-1. Follow these steps:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 4 because the second mask octet of 240 includes four binary 1s.

Step 4. The resulting prefix is /12.

Mask Conversion Problem 2: Answer

The answer is /18.

For problem 2, mask 255.255.192.0 converts to the following:

11111111 11111111 11000000 00000000

You can see from the binary number that it contains 18 binary 1s, so the prefix format of the mask will be /18.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 2 because the third mask octet of 192 includes two binary 1s.

Step 5. The resulting prefix is /18.

Mask Conversion Problem 3: Answer

The answer is /27.

For problem 3, mask 255.255.255.224 converts to the following:

11111111 11111111 11111111 11100000

You can see from the binary number that it contains 27 binary 1s, so the prefix format of the mask will be /27.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 8 because the third mask octet of 255 includes eight binary 1s.

Step 5. (4th octet) Add 3 because the fourth mask octet of 224 includes three binary 1s.

Step 6. The resulting prefix is /27.

Mask Conversion Problem 4: Answer

The answer is /15.

For problem 4, mask 255.254.0.0 converts to the following:

11111111 11111110 00000000 00000000

You can see from the binary number that it contains 15 binary 1s, so the prefix format of the mask will be /15.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 7 because the second mask octet of 254 includes seven binary 1s.

Step 4. The resulting prefix is /15.

Mask Conversion Problem 5: Answer

The answer is /21.

For problem 5, mask 255.255.248.0 converts to the following:

11111111 11111111 11111000 00000000

You can see from the binary number that it contains 21 binary 1s, so the prefix format of the mask will be /21.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 5 because the third mask octet of 248 includes five binary 1s.

Step 5. The resulting prefix is /21.

Mask Conversion Problem 6: Answer

The answer is 255.255.255.252.

The binary process for converting the prefix version of the mask to dotted-decimal is straightforward, but again requires some binary math. For reference, the process runs like this:

Step 1. Write down x binary 1s, where x is the value listed in the prefix version of the mask.

Step 2. Write down binary 0s after the binary 1s until the combined 1s and 0s form a 32-bit number.

Step 3. Convert this binary number, 8 bits at a time, to decimal, to create a dotted-decimal number; this value is the dotted-decimal version of the subnet mask. (Refer to Table E-1, which lists the binary and decimal equivalents.)

For problem 6, with a prefix of /30, you start at Step 1 by writing down 30 binary 1s, as shown here:

11111111 11111111 11111111 111111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111111 11111**100**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 7: Answer

The answer is 255.255.255.128.

For problem 7, with a prefix of /25, you start at Step 1 by writing down 25 binary 1s, as shown here:

11111111 11111111 11111111 1

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111111 **1**0000000

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 8: Answer

The answer is 255.224.0.0.

For problem 8, with a prefix of /11, you start at Step 1 by writing down 11 binary 1s, as shown here:

11111111 111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 111**00000 00000000 00000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 9: Answer

The answer is 255.255.252.0.

For problem 9, with a prefix of /22, you start at Step 1 by writing down 22 binary 1s, as shown here:

11111111 11111111 111111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111**100 00000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 10: Answer

The answer is 255.255.255.0.

For problem 10, with a prefix of /24, you start at Step 1 by writing down 24 binary 1s, as shown here:

11111111 11111111 11111111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111111 **00000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 11: Answer

The answer is /8.

For problem 11, mask 255.0.0.0 converts to the following:

11111111 00000000 00000000 00000000

You can see from the binary number that it contains 8 binary 1s, so the prefix format of the mask will be /8.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 0 for the other octets because each mask octet of 0 includes zero binary 1s.

Step 4. The resulting prefix is /8.

Mask Conversion Problem 12: Answer

The answer is 255.255.255.248.

For problem 12, with a prefix of /29, you start at Step 1 by writing down 29 binary 1s, as shown here:

11111111 11111111 11111111 11111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111111 11111**000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 13: Answer

The answer is 255.128.0.0.

For problem 13, with a prefix of /9, you start at Step 1 by writing down 9 binary 1s, as shown here:

11111111 1

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 10000000 00000000 00000000

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 14: Answer

The answer is /10.

For problem 14, mask 255.192.0.0 converts to the following:

11111111 11000000 00000000 00000000

You can see from the binary number that it contains 10 binary 1s, so the prefix format of the mask will be /10.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 2 because the second mask octet of 192 includes two binary 1s.

Step 4. The resulting prefix is /10.

Mask Conversion Problem 15: Answer

The answer is /28.

For problem 15, mask 255.255.255.240 converts to the following:

11111111 11111111 11111111 11110000

You can see from the binary number that it contains 28 binary 1s, so the prefix format of the mask will be /28.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 8 because the third mask octet of 255 includes eight binary 1s.

Step 5. (4th octet) Add 4 because the fourth mask octet of 240 includes four binary 1s.

Step 6. The resulting prefix is /28.

Mask Conversion Problem 16: Answer

The answer is 255.255.255.192.

For problem 16, with a prefix of /26, you start at Step 1 by writing down 26 binary 1s, as shown here:

11111111 11111111 11111111 11

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 11111111 11**000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 17: Answer

The answer is 255.248.0.0.

For problem 17, with a prefix of /13, you start at Step 1 by writing down 13 binary 1s, as shown here:

11111111 11111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111**000 00000000 00000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 18: Answer

The answer is /23.

For problem 18, mask 255.255.254.0 converts to the following:

11111111 11111111 11111110 00000000

You can see from the binary number that it contains 23 binary 1s, so the prefix format of the mask will be /23.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 7 because the third mask octet of 254 includes seven binary 1s.

Step 5. The resulting prefix is /23.

Mask Conversion Problem 19: Answer

The answer is /14.

For problem 19, mask 255.252.0.0 converts to the following:

11111111 11111100 00000000 00000000

You can see from the binary number that it contains 14 binary 1s, so the prefix format of the mask will be /14.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 6 because the second mask octet of 252 includes six binary 1s.

Step 4. The resulting prefix is /14.

Mask Conversion Problem 20: Answer

The answer is 255.255.240.0.

For problem 20, with a prefix of /20, you start at Step 1 by writing down 20 binary 1s, as shown here:

11111111 11111111 1111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 **1111**0000 00000000

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 21: Answer

The answer is 255.255.0.0.

For problem 21, with a prefix of /16, you start at Step 1 by writing down 16 binary 1s, as shown here:

11111111 11111111

At Step 2, you add binary 0s until you have 32 total bits, as shown next:

11111111 11111111 **00000000 00000000**

The only remaining work is to convert this 32-bit number to decimal, remembering that the conversion works with 8 bits at a time.

Mask Conversion Problem 22: Answer

The answer is /19.

For problem 22, mask 255.255.224.0 converts to the following:

11111111 11111111 11100000 00000000

You can see from the binary number that it contains 19 binary 1s, so the prefix format of the mask will be /19.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 3 because the third mask octet of 224 includes three binary 1s.

Step 5. The resulting prefix is /19.

Mask Conversion Problem 23: Answer

The answer is /17.

For problem 23, mask 255.255.128.0 converts to the following:

11111111 11111111 10000000 00000000

You can see from the binary number that it contains 17 binary 1s, so the prefix format of the mask will be /17.

If you memorized the number of binary 1s represented by each DDN mask value, you can possibly work faster with the following logic:

Step 1. Start with a prefix value of 0.

Step 2. (1st octet) Add 8 because the first mask octet of 255 includes eight binary 1s.

Step 3. (2nd octet) Add 8 because the second mask octet of 255 includes eight binary 1s.

Step 4. (3rd octet) Add 1 because the third mask octet of 128 includes one binary 1.

Step 5. The resulting prefix is /17.

Mask Analysis Problems

This appendix lists problems that require you to analyze an existing IP address and mask to determine the number of network, subnet, and host bits. From that, you should calculate the number of subnets possible when using the listed mask in the class of network shown in the problem, as well as the number of possible host addresses in each subnet.

To find this information, you can use the processes explained in Chapter 13 of *CCNA 200-301 Official Cert Guide, Volume 1*. When doing the problems, Table E-1, earlier in this appendix, which lists all possible DDN mask values, can be useful.

Each row of Table E-2 lists an IP address and mask. For each row, complete the table. Note that for the purposes of this exercise you can assume that the two special subnets in each network, the zero subnet and broadcast subnet, are allowed to be used.

Table E-2 Mask Analysis Problems

Problem Number	Problem	Network Bits	Subnet Bits	Host Bits	Number of Subnets in Network	Number of Hosts per Subnet
1	10.66.5.99, 255.255.254.0					
2	172.16.203.42, 255.255.252.0					
3	192.168.55.55, 255.255.255.224					
4	10.22.55.87/30					
5	172.30.40.166/26					
6	192.168.203.18/29					
7	200.11.88.211, 255.255.255.240					
8	128.1.211.33, 255.255.255.128					
9	9.211.45.65/21					
10	223.224.225.226/25					

Answers to Mask Analysis Problems

Table E-3 includes the answers to problems 1–10. The paragraphs following the table provide the explanations of each answer.

Table E-3 Answers to Problems in This Appendix

Problem Number	Problem	Network Bits	Subnet Bits	Host Bits	Number of Subnets in Network	Number of Hosts per Subnet
1	10.66.5.99, 255.255.254.0	8	15	9	$2^{15} = 32{,}768$	$2^9 - 2 = 510$
2	172.16.203.42, 255.255.252.0	16	6	10	$2^6 = 64$	$2^{10} - 2 = 1022$
3	192.168.55.55, 255.255.255.224	24	3	5	$2^3 = 8$	$2^5 - 2 = 30$
4	10.22.55.87/30	8	22	2	$2^{22} = 4{,}194{,}304$	$2^2 - 2 = 2$
5	172.30.40.166/26	16	10	6	$2^{10} = 1024$	$2^6 - 2 = 62$
6	192.168.203.18/29	24	5	3	$2^5 = 32$	$2^3 - 2 = 6$
7	200.11.88.211, 255.255.255.240	24	4	4	$2^4 = 16$	$2^4 - 2 = 14$
8	128.1.211.33, 255.255.255.128	16	9	7	$2^9 = 512$	$2^7 - 2 = 126$
9	9.211.45.65/21	8	13	11	$2^{13} = 8192$	$2^{11} - 2 = 2046$
10	223.224.225.226/25	24	1	7	$2^1 = 2$	$2^7 - 2 = 126$

Mask Analysis Problem 1: Answer

Address 10.66.5.99 is in Class A network 10.0.0.0, meaning that 8 network bits exist. Mask 255.255.254.0 converts to prefix /23, because the first 2 octets of value 255 represent 8 binary 1s, and the 254 in the third octet represents 7 binary 1s, for a total of 23 binary 1s. Therefore, the number of host bits is 32 – 23 = 9, leaving 15 subnet bits (32 – 8 network bits – 9 host bits = 15 subnet bits). The number of subnets in this Class A network, using mask 255.255.254.0, is $2^{15} = 32{,}768$. The number of hosts per subnet is $2^9 - 2 = 510$.

Mask Analysis Problem 2: Answer

Address 172.16.203.42, mask 255.255.252.0, is in Class B network 172.16.0.0, meaning that 16 network bits exist. Mask 255.255.252.0 converts to prefix /22, because the first 2 octets of value 255 represent 8 binary 1s, and the 252 in the third octet represents 6 binary 1s, for a total of 22 binary 1s. Therefore, the number of host bits is 32 – 22 = 10, leaving 6 subnet bits (32 – 16 network bits – 10 host bits = 6 subnet bits). The number of subnets in this Class B network, using mask 255.255.252.0, is $2^6 = 64$. The number of hosts per subnet is $2^{10} - 2 = 1022$.

Mask Analysis Problem 3: Answer

Address 192.168.55.55 is in Class C network 192.168.55.0, meaning that 24 network bits exist. Mask 255.255.255.224 converts to prefix /27, because the first 3 octets of value 255 represent 8 binary 1s, and the 224 in the fourth octet represents 3 binary 1s, for a total of 27 binary 1s. Therefore, the number of host bits is 32 – 27 = 5, leaving 3 subnet bits (32 – 24 network bits – 5 host bits = 3 subnet bits). The number of subnets in this Class C network, using mask 255.255.255.224, is $2^3 = 8$. The number of hosts per subnet is $2^5 – 2 = 30$.

Mask Analysis Problem 4: Answer

Address 10.22.55.87 is in Class A network 10.0.0.0, meaning that 8 network bits exist. The prefix format mask of /30 lets you calculate the number of host bits as 32 – prefix length (in this case, 32 – 30 = 2). This leaves 22 subnet bits (32 – 8 network bits – 2 host bits = 22 subnet bits). The number of subnets in this Class A network, using mask 255.255.255.252, is $2^{22} = 4{,}194{,}304$. The number of hosts per subnet is $2^2 – 2 = 2$. (Note that this mask is popularly used on serial links, which need only two IP addresses in a subnet.)

Mask Analysis Problem 5: Answer

Address 172.30.40.166 is in Class B network 172.30.0.0, meaning that 16 network bits exist. The prefix format mask of /26 lets you calculate the number of host bits as 32 – prefix length (in this case, 32 – 26 = 6). This leaves 10 subnet bits (32 – 16 network bits – 6 host bits = 10 subnet bits). The number of subnets in this Class B network, using mask /26, is $2^{10} = 1024$. The number of hosts per subnet is $2^6 – 2 = 62$.

Mask Analysis Problem 6: Answer

Address 192.168.203.18 is in Class C network 192.168.203.0, meaning that 24 network bits exist. The prefix format mask of /29 lets you calculate the number of host bits as 32 – prefix length (in this case, 32 – 29 = 3). This leaves 5 subnet bits, because 32 – 24 network bits – 3 host bits = 5 subnet bits. The number of subnets in this Class C network, using mask /29, is $2^5 = 32$. The number of hosts per subnet is $2^3 – 2 = 6$.

Mask Analysis Problem 7: Answer

Address 200.11.88.211 is in Class C network 200.11.88.0, meaning that 24 network bits exist. Mask 255.255.255.240 converts to prefix /28, because the first three octets of value 255 represent 8 binary 1s, and the 240 in the fourth octet represents 4 binary 1s, for a total of 28 binary 1s. This leaves 4 subnet bits (32 – 24 network bits – 4 host bits = 4 subnet bits). The number of subnets in this Class C network, using mask /28, is $2^4 = 16$. The number of hosts per subnet is $2^4 – 2 = 14$.

Mask Analysis Problem 8: Answer

Address 128.1.211.33, mask 255.255.255.128, is in Class B network 128.1.0.0, meaning that 16 network bits exist. Mask 255.255.255.128 converts to prefix /25, because the first 3 octets of value 255 represent 8 binary 1s, and the 128 in the fourth octet represents 1 binary 1, for a total of 25 binary 1s. Therefore, the number of host bits is 32 – 25 = 7, leaving 9 subnet bits (32 – 16 network bits – 7 host bits = 9 subnet bits). The number of subnets in this Class B network, using mask 255.255.255.128, is $2^9 = 512$. The number of hosts per subnet is $2^7 – 2 = 126$.

Mask Analysis Problem 9: Answer

Address 9.211.45.65 is in Class A network 10.0.0.0, meaning that 8 network bits exist. The prefix format mask of /21 lets you calculate the number of host bits as 32 – prefix length (in this case, 32 – 21 = 11). This leaves 13 subnet bits (32 – 8 network bits – 11 host bits = 13 subnet bits). The number of subnets in this Class A network, using mask /21, is 2^{13} = 8192. The number of hosts per subnet is 2^{11} – 2 = 2046.

Mask Analysis Problem 10: Answer

Address 223.224.225.226 is in Class C network 223.224.225.0, meaning that 24 network bits exist. The prefix format mask of /25 lets you calculate the number of host bits as 32 – prefix length (in this case, 32 – 25 = 7). This leaves 1 subnet bit (32 – 24 network bits – 7 host bits = 1 subnet bit). The number of subnets in this Class C network, using mask /25, is 2^1 = 2. The number of hosts per subnet is 2^7 – 2 = 126.

APPENDIX F

Practice for Chapter 14: Analyzing Existing Subnets

Practice Problems

This appendix lists practice problems related to Chapter 14, "Analyzing Existing Subnets." Each problem asks you to find a variety of information about the subnet in which an IP address resides. Each problem supplies an IP address and a subnet mask, from which you should find the following information:

- Subnet number
- Subnet broadcast address
- Range of valid IP addresses in this network

To find these facts, you can use any of the processes explained in Chapter 14.

In addition, these same problems can be used to review the concepts in Chapter 13, "Analyzing Subnet Masks." To use these same problems for practice related to Chapter 13, simply find the following information for each of the problems:

- Size of the network part of the address
- Size of the subnet part of the address
- Size of the host part of the address
- Number of hosts per subnet
- Number of subnets in this network

Feel free to either ignore or use the opportunity for more practice related to analyzing subnet masks.

Solve for the following problems:

1. 10.180.10.18, mask 255.192.0.0
2. 10.200.10.18, mask 255.224.0.0
3. 10.100.18.18, mask 255.240.0.0
4. 10.100.18.18, mask 255.248.0.0
5. 10.150.200.200, mask 255.252.0.0
6. 10.150.200.200, mask 255.254.0.0
7. 10.220.100.18, mask 255.255.0.0
8. 10.220.100.18, mask 255.255.128.0
9. 172.31.100.100, mask 255.255.192.0

10. 172.31.100.100, mask 255.255.224.0
11. 172.31.200.10, mask 255.255.240.0
12. 172.31.200.10, mask 255.255.248.0
13. 172.31.50.50, mask 255.255.252.0
14. 172.31.50.50, mask 255.255.254.0
15. 172.31.140.14, mask 255.255.255.0
16. 172.31.140.14, mask 255.255.255.128
17. 192.168.15.150, mask 255.255.255.192
18. 192.168.15.150, mask 255.255.255.224
19. 192.168.100.100, mask 255.255.255.240
20. 192.168.100.100, mask 255.255.255.248
21. 192.168.15.230, mask 255.255.255.252
22. 10.1.1.1, mask 255.248.0.0
23. 172.16.1.200, mask 255.255.240.0
24. 172.16.0.200, mask 255.255.255.192
25. 10.1.1.1, mask 255.0.0.0

Answers

This section includes the answers to the 25 problems listed in this appendix. The answer section for each problem explains how to use the process outlined in Chapter 14 to find the answers. Also, refer to Chapter 13 for details on how to find information about analyzing the subnet mask.

Answer to Problem 1

The answers begin with the analysis of the three parts of the address, the number of hosts per subnet, and the number of subnets of this network using the stated mask, as outlined in Table F-1. The binary math for subnet and broadcast address calculation follows. The answer finishes with the easier mental calculations for the range of IP addresses in the subnet.

Table F-1 Question 1: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.180.10.18	—
Mask	255.192.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	22	Always defined as number of binary 0s in mask
Number of subnet bits	2	32 – (network size + host size)
Number of subnets	$2^2 = 4$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{22} - 2 = 4{,}194{,}302$	$2^{\text{number-of-host-bits}} - 2$

Table F-2 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-2 Question 1: Binary Calculation of Subnet and Broadcast Addresses

Address	10.180.10.18	00001010 10**110100 00001010 00010010**
Mask	255.192.0.0	11111111 11**000000 00000000 00000000**
AND result (subnet number)	10.128.0.0	00001010 10**000000 00000000 00000000**
Change host to 1s (broadcast address)	10.191.255.255	00001010 10**111111 11111111 11111111**

To get the first valid IP address, just add 1 to the subnet number; to get the last valid IP address, just subtract 1 from the broadcast address. In this case:

10.128.0.1 through 10.191.255.254

10.128.0.0 + 1 = 10.128.0.1

10.191.255.255 – 1 = 10.191.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. The key parts of the process are as follows:

- The interesting octet is the octet for which the mask's value is not a decimal 0 or 255.
- The magic number is calculated as the value of the IP address's interesting octet, subtracted from 256.
- The subnet number can be found by copying the IP address octets to the left of the interesting octet, by writing down 0s for octets to the right of the interesting octet, and by finding the multiple of the magic number closest to, but not larger than, the IP address's value in that same octet.
- The broadcast address can be similarly found by copying the subnet number's octets to the left of the interesting octet, by writing 255s for octets to the right of the interesting octet, and by taking the subnet number's value in the interesting octet, adding the magic number, and subtracting 1.

Table F-3 shows the work for this problem, with some explanation of the work following the table. Refer to Chapter 14 for the detailed processes.

Table F-3 Question 1: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4	Comments
Mask	255	192	0	0	
Address	10	180	10	18	
Subnet Number	10	128	0	0	Magic number = 256 – 192 = 64
First Address	10	128	0	1	Add 1 to last octet of subnet
Last Address	10	191	255	254	Subtract 1 from last octet of broadcast
Broadcast	10	191	255	255	128 + 64 – 1 = 191

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 192 = 64 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 128 is the multiple of 64 that is closest to 180 but not higher than 180. So, the second octet of the subnet number is 128.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 128 + 64 – 1 = 191.

Answer to Problem 2

Table F-4 Question 2: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.200.10.18	—
Mask	255.224.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	21	Always defined as number of binary 0s in mask
Number of subnet bits	3	32 – (network size + host size)
Number of subnets	$2^3 = 8$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{21} – 2 = 2{,}097{,}150$	$2^{\text{number-of-host-bits}} – 2$

Table F-5 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-5 Question 2: Binary Calculation of Subnet and Broadcast Addresses

Address	10.200.10.18	00001010 11001000 00001010 00010010
Mask	255.224.0.0	11111111 11100000 00000000 00000000
AND result (subnet number)	10.192.0.0	00001010 11000000 00000000 00000000
Change host to 1s (broadcast address)	10.223.255.255	00001010 11011111 11111111 11111111

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.192.0.1 through 10.223.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-6 shows the work for this problem, with some explanation of the work following the table.

Table F-6 Question 2: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4	Comments
Mask	255	224	0	0	
Address	10	200	10	18	
Subnet Number	10	192	0	0	Magic number = 256 – 224 = 32
First Address	10	192	0	1	Add 1 to last octet of subnet
Last Address	10	223	255	254	Subtract 1 from last octet of broadcast
Broadcast	10	223	255	255	192 + 32 – 1 = 223

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 224 = 32 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 192 is the multiple of 32 that is closest to 200 but not higher than 200. So, the second octet of the subnet number is 192.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 192 + 32 – 1 = 223.

Answer to Problem 3

Table F-7 Question 3: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.100.18.18	—
Mask	255.240.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	20	Always defined as number of binary 0s in mask
Number of subnet bits	4	32 – (network size + host size)
Number of subnets	$2^4 = 16$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{20} – 2 = 1{,}048{,}574$	$2^{\text{number-of-host-bits}} – 2$

Table F-8 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-8 Question 3: Binary Calculation of Subnet and Broadcast Addresses

Address	10.100.18.18	`00001010 0110`**`0100 00010010 00010010`**
Mask	255.240.0.0	`11111111 1111`**`0000 00000000 00000000`**
AND result (subnet number)	10.96.0.0	`00001010 0110`**`0000 00000000 00000000`**
Change host to 1s (broadcast address)	10.111.255.255	`00001010 0110`**`1111 11111111 11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.96.0.1 through 10.111.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-9 shows the work for this problem, with some explanation of the work following the table.

Table F-9 Question 3: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4	Comments
Mask	255	240	0	0	—
Address	10	100	18	18	—
Subnet Number	10	96	0	0	Magic number = 256 – 240 = 16
First Address	10	96	0	1	Add 1 to last octet of subnet
Last Address	10	111	255	254	Subtract 1 from last octet of broadcast
Broadcast	10	111	255	255	96 + 16 – 1 = 111

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 240 = 16 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 96 is the multiple of 16 that is closest to 100 but not higher than 100. So, the second octet of the subnet number is 96.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 96 + 16 – 1 = 111.

Answer to Problem 4

Table F-10 Question 4: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.100.18.18	—
Mask	255.248.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	19	Always defined as number of binary 0s in mask
Number of subnet bits	5	32 – (network size + host size)
Number of subnets	$2^5 = 32$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{19} – 2 = 524{,}286$	$2^{\text{number-of-host-bits}} – 2$

Table F-11 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-11 Question 4: Binary Calculation of Subnet and Broadcast Addresses

Address	10.100.18.18	`00001010 01100`**`100 00010010 00010010`**
Mask	255.248.0.0	`11111111 11111`**`000 00000000 00000000`**
AND result (subnet number)	10.96.0.0	`00001010 01100`**`000 00000000 00000000`**
Change host to 1s (broadcast address)	10.103.255.255	`00001010 01100`**`111 11111111 11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.96.0.1 through 10.103.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-12 shows the work for this problem, with some explanation of the work following the table.

F

Table F-12 Question 4: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4	Comments
Mask	255	248	0	0	—
Address	10	100	18	18	—
Subnet Number	10	96	0	0	Magic number = 256 – 248 = 8
First Address	10	96	0	1	Add 1 to last octet of subnet
Last Address	10	103	255	254	Subtract 1 from last octet of broadcast
Broadcast	10	103	255	255	96 + 8 – 1 = 103

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 248 = 8 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 96 is the multiple of 8 that is closest to 100 but not higher than 100. So, the second octet of the subnet number is 96.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 96 + 8 – 1 = 103.

Answer to Problem 5

Table F-13 Question 5: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.150.200.200	—
Mask	255.252.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	18	Always defined as number of binary 0s in mask
Number of subnet bits	6	32 – (network size + host size)
Number of subnets	$2^6 = 64$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{18} – 2 = 262{,}142$	$2^{\text{number-of-host-bits}} – 2$

Table F-14 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-14 Question 5: Binary Calculation of Subnet and Broadcast Addresses

Address	10.150.200.200	00001010 10010110 11001000 11001000
Mask	255.252.0.0	11111111 11111100 00000000 00000000
AND result (subnet number)	10.148.0.0	00001010 10010100 00000000 00000000
Change host to 1s (broadcast address)	10.151.255.255	00001010 10010111 11111111 11111111

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.148.0.1 through 10.151.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-15 shows the work for this problem, with some explanation of the work following the table.

Table F-15 Question 5: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4	Comments
Mask	255	252	0	0	—
Address	10	150	200	200	—
Subnet Number	10	148	0	0	Magic number = 256 – 252 = 4
First Address	10	148	0	1	Add 1 to last octet of subnet
Last Address	10	151	255	254	Subtract 1 from last octet of broadcast
Broadcast	10	151	255	255	148 + 4 – 1 = 151

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 252 = 4 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 148 is the multiple of 4 that is closest to 150 but not higher than 150. So, the second octet of the subnet number is 148.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 148 + 4 – 1 = 151.

Answer to Problem 6

Table F-16 Question 6: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.150.200.200	—
Mask	255.254.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	17	Always defined as number of binary 0s in mask
Number of subnet bits	7	32 – (network size + host size)
Number of subnets	$2^7 = 128$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{17} – 2 = 131{,}070$	$2^{\text{number-of-host-bits}} – 2$

Table F-17 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-17 Question 6: Binary Calculation of Subnet and Broadcast Addresses

Address	10.150.200.200	`00001010 10010110 11001000 11001000`
Mask	255.254.0.0	`11111111 11111110 00000000 00000000`
AND result (subnet number)	10.150.0.0	`00001010 10010110 00000000 00000000`
Change host to 1s (broadcast address)	10.151.255.255	`00001010 10010111 11111111 11111111`

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.150.0.1 through 10.151.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-18 shows the work for this problem, with some explanation of the work following the table.

Table F-18 Question 6: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	254	0	0
Address	10	150	200	200
Subnet Number	10	150	0	0
First Valid Address	10	150	0	1
Last Valid Address	10	151	255	254
Broadcast	10	151	255	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 254 = 2 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 150 is the multiple of 2 that is closest to 150 but not higher than 150. So, the second octet of the subnet number is 150.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 150 + 2 – 1 = 151.

Answer to Problem 7

Table F-19 Question 7: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.220.100.18	—
Mask	255.255.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	16	Always defined as number of binary 0s in mask
Number of subnet bits	8	32 – (network size + host size)
Number of subnets	$2^8 = 256$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{16} – 2 = 65{,}534$	$2^{\text{number-of-host-bits}} – 2$

Table F-20 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-20 Question 7: Binary Calculation of Subnet and Broadcast Addresses

Address	10.220.100.18	`00001010 11011100 01100100 00010010`
Mask	255.255.0.0	`11111111 11111111 00000000 00000000`
AND result (subnet number)	10.220.0.0	`00001010 11011100 00000000 00000000`
Change host to 1s (broadcast address)	10.220.255.255	`00001010 11011100 11111111 11111111`

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.220.0.1 through 10.220.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-21 shows the work for this problem.

Table F-21 Question 7: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	0	0
Address	10	220	100	18
Subnet Number	10	220	0	0
First Valid Address	10	220	0	1
Last Valid Address	10	220	255	254
Broadcast	10	220	255	255

This subnetting scheme uses an easy mask because all the octets are a 0 or a 255. No math tricks are needed.

Answer to Problem 8

Table F-22 Question 8: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.220.100.18	—
Mask	255.255.128.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	15	Always defined as number of binary 0s in mask
Number of subnet bits	9	32 – (network size + host size)
Number of subnets	$2^9 = 512$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{15} – 2 = 32{,}766$	$2^{\text{number-of-host-bits}} – 2$

Table F-23 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-23 Question 8: Binary Calculation of Subnet and Broadcast Addresses

Address	10.220.100.18	`00001010 11011100 01100100 00010010`
Mask	255.255.128.0	`11111111 11111111 10000000 00000000`
AND result (subnet number)	10.220.0.0	`00001010 11011100 00000000 00000000`
Change host to 1s (broadcast address)	10.220.127.255	`00001010 11011100 01111111 11111111`

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.220.0.1 through 10.220.127.254

Table F-24 shows the work for this problem, with some explanation of the work following the table. Refer to Chapter 14 for the detailed processes.

Table F-24 Question 8: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	128	0
Address	10	220	100	18
Subnet Number	10	220	0	0
First Address	10	220	0	1
Last Address	10	220	127	254
Broadcast	10	220	127	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 128 = 128 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 0 is the multiple of 128 that is closest to 100 but not higher than 100. So, the third octet of the subnet number is 0.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 0 + 128 – 1 = 127.

This example tends to confuse people, because a mask with 128 in it gives you subnet numbers that just do not seem to look right. Table F-25 gives you the answers for the first several subnets, just to make sure that you are clear about the subnets when using this mask with a Class A network.

Table F-25 Question 8: First Four Subnets

	Zero Subnet	2nd Subnet	3rd Subnet	4th Subnet
Subnet	10.0.0.0	10.0.128.0	10.1.0.0	10.1.128.0
First Address	10.0.0.1	10.0.128.1	10.1.0.1	10.1.128.1
Last Address	10.0.127.254	10.0.255.254	10.1.127.254	10.1.255.254
Broadcast	10.0.127.255	10.0.255.255	10.1.127.255	10.1.255.255

Answer to Problem 9

Table F-26 Question 9: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.100.100	—
Mask	255.255.192.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	14	Always defined as number of binary 0s in mask
Number of subnet bits	2	32 – (network size + host size)
Number of subnets	$2^2 = 4$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{14} - 2 = 16{,}382$	$2^{\text{number-of-host-bits}} - 2$

Table F-27 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-27 Question 9: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.100.100	`10101100 00011111 01`**`100100 01100100`**
Mask	255.255.192.0	`11111111 11111111 11`**`000000 00000000`**
AND result (subnet number)	172.31.64.0	`10101100 00011111 01`**`000000 00000000`**
Change host to 1s (broadcast address)	172.31.127.255	`10101100 00011111 01`**`111111 11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.64.1 through 172.31.127.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-28 shows the work for this problem, with some explanation of the work following the table.

Table F-28 Question 9: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	192	0
Address	172	31	100	100
Subnet Number	172	31	64	0
First Valid Address	172	31	64	1
Last Valid Address	172	31	127	254
Broadcast	172	31	127	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 192 = 64 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 64 is the multiple of 64 that is closest to 100 but not higher than 100. So, the third octet of the subnet number is 64.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 64 + 64 – 1 = 127.

Answer to Problem 10

Table F-29 Question 10: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.100.100	—
Mask	255.255.224.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	13	Always defined as number of binary 0s in mask
Number of subnet bits	3	32 – (network size + host size)
Number of subnets	$2^3 = 8$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{13} – 2 = 8190$	$2^{\text{number-of-host-bits}} – 2$

Table F-30 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-30 Question 10: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.100.100	10101100 00011111 011**00100 01100100**
Mask	255.255.224.0	11111111 11111111 111**00000 00000000**
AND result (subnet number)	172.31.96.0	10101100 00011111 011**00000 00000000**
Change host to 1s (broadcast address)	172.31.127.255	10101100 00011111 011**11111 11111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.96.1 through 172.31.127.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-31 shows the work for this problem, with some explanation of the work following the table.

Table F-31 Question 10: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	224	0
Address	172	31	100	100
Subnet Number	172	31	96	0
First Valid Address	172	31	96	1
Last Valid Address	172	31	127	254
Broadcast	172	31	127	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 224 = 32 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 96 is the multiple of 32 that is closest to 100 but not higher than 100. So, the third octet of the subnet number is 96.

The second part of this process calculates the subnet broadcast address, with the tricky parts, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 96 + 32 – 1 = 127.

Answer to Problem 11

Table F-32 Question 11: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.200.10	—
Mask	255.255.240.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	12	Always defined as number of binary 0s in mask
Number of subnet bits	4	32 – (network size + host size)
Number of subnets	$2^4 = 16$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{12} – 2 = 4094$	$2^{\text{number-of-host-bits}} – 2$

Table F-33 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-33 Question 11: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.200.10	10101100 00011111 1100**1000 00001010**
Mask	255.255.240.0	11111111 11111111 1111**0000 00000000**
AND result (subnet number)	172.31.192.0	10101100 00011111 1100**0000 00000000**
Change host to 1s (broadcast address)	172.31.207.255	10101100 00011111 1100**1111 11111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.192.1 through 172.31.207.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-34 shows the work for this problem, with some explanation of the work following the table.

Table F-34 Question 11: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	240	0
Address	172	31	200	10
Subnet Number	172	31	192	0
First Valid Address	172	31	192	1
Last Valid Address	172	31	207	254
Broadcast	172	31	207	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 240 = 16 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 192 is the multiple of 16 that is closest to 200 but not higher than 200. So, the third octet of the subnet number is 192.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 192 + 16 – 1 = 207.

Answer to Problem 12

Table F-35 Question 12: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.200.10	—
Mask	255.255.248.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	11	Always defined as number of binary 0s in mask
Number of subnet bits	5	32 – (network size + host size)
Number of subnets	$2^5 = 32$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{11} – 2 = 2046$	$2^{\text{number-of-host-bits}} – 2$

Table F-36 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-36 Question 12: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.200.10	10101100 00011111 11001**000 00001010**
Mask	255.255.248.0	11111111 11111111 11111**000 00000000**
AND result (subnet number)	172.31.200.0	10101100 00011111 11001**000 00000000**
Change host to 1s (broadcast address)	172.31.207.255	10101100 00011111 11001**111 11111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.200.1 through 172.31.207.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-37 shows the work for this problem, with some explanation of the work following the table.

Table F-37 Question 12: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	248	0
Address	172	31	200	10
Subnet Number	172	31	200	0
First Valid Address	172	31	200	1
Last Valid Address	172	31	207	254
Broadcast	172	31	207	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 248 = 8 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 200 is the multiple of 8 that is closest to 200 but not higher than 200. So, the third octet of the subnet number is 200.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 200 + 8 – 1 = 207.

Answer to Problem 13

Table F-38 Question 13: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.50.50	—
Mask	255.255.252.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	10	Always defined as number of binary 0s in mask
Number of subnet bits	6	32 – (network size + host size)
Number of subnets	$2^6 = 64$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{10} – 2 = 1022$	$2^{\text{number-of-host-bits}} – 2$

Table F-39 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-39 Question 13: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.50.50	10101100 00011111 00110010 00110010
Mask	255.255.252.0	11111111 11111111 11111100 00000000
AND result (subnet number)	172.31.48.0	10101100 00011111 00110000 00000000
Change host to 1s (broadcast address)	172.31.51.255	10101100 00011111 00110011 11111111

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.48.1 through 172.31.51.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-40 shows the work for this problem, with some explanation of the work following the table.

F

Table F-40 Question 13: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	252	0
Address	172	31	50	50
Subnet Number	172	31	48	0
First Valid Address	172	31	48	1
Last Valid Address	172	31	51	254
Broadcast	172	31	51	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 252 = 4 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 48 is the multiple of 4 that is closest to 50 but not higher than 50. So, the third octet of the subnet number is 48.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 48 + 4 – 1 = 51.

Answer to Problem 14

Table F-41 Question 14: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.50.50	—
Mask	255.255.254.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	9	Always defined as number of binary 0s in mask
Number of subnet bits	7	32 – (network size + host size)
Number of subnets	$2^7 = 128$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^9 – 2 = 510$	$2^{\text{number-of-host-bits}} – 2$

Table F-42 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-42 Question 14: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.50.50	10101100 00011111 00110010 00110010
Mask	255.255.254.0	11111111 11111111 11111110 00000000
AND result (subnet number)	172.31.50.0	10101100 00011111 00110010 00000000
Change host to 1s (broadcast address)	172.31.51.255	10101100 00011111 00110011 11111111

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.50.1 through 172.31.51.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-43 shows the work for this problem, with some explanation of the work following the table.

Table F-43 Question 14: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	254	0
Address	172	31	50	50
Subnet Number	172	31	50	0
First Valid Address	172	31	50	1
Last Valid Address	172	31	51	254
Broadcast	172	31	51	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 254 = 2 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 50 is the multiple of 2 that is closest to 50 but not higher than 50. So, the third octet of the subnet number is 50.

The second part of this process calculates the subnet broadcast address with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 50 + 2 – 1 = 51.

Answer to Problem 15

Table F-44 Question 15: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.140.14	—
Mask	255.255.255.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	8	Always defined as number of binary 0s in mask
Number of subnet bits	8	32 – (network size + host size)
Number of subnets	$2^8 = 256$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^8 – 2 = 254$	$2^{\text{number-of-host-bits}} – 2$

Table F-45 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-45 Question 15: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.140.14	`10101100 00011111 10001100` **`00001110`**
Mask	255.255.255.0	`11111111 11111111 11111111` **`00000000`**
AND result (subnet number)	172.31.140.0	`10101100 00011111 10001100` **`00000000`**
Change host to 1s (broadcast address)	172.31.140.255	`10101100 00011111 10001100` **`11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.140.1 through 172.31.140.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-46 shows the work for this problem.

Table F-46 Question 15: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	0
Address	172	31	140	14
Subnet Number	172	31	140	0
First Valid Address	172	31	140	1
Last Valid Address	172	31	140	254
Broadcast	172	31	140	255

This subnetting scheme uses an easy mask because all the octets are a 0 or a 255. No math tricks are needed.

Answer to Problem 16

Table F-47 Question 16: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.31.140.14	—
Mask	255.255.255.128	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	7	Always defined as number of binary 0s in mask
Number of subnet bits	9	32 – (network size + host size)
Number of subnets	$2^9 = 512$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^7 - 2 = 126$	$2^{\text{number-of-host-bits}} - 2$

Table F-48 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-48 Question 16: Binary Calculation of Subnet and Broadcast Addresses

Address	172.31.140.14	`10101100 00011111 10001100 0`**`0001110`**
Mask	255.255.255.128	`11111111 11111111 11111111 1`**`0000000`**
AND result (subnet number)	172.31.140.0	`10101100 00011111 10001100 0`**`0000000`**
Change host to 1s (broadcast address)	172.31.140.127	`10101100 00011111 10001100 0`**`1111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.31.140.1 through 172.31.140.126

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-49 shows the work for this problem, with some explanation of the work following the table.

Table F-49 Question 16: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	128
Address	172	31	140	14
Subnet Number	172	31	140	0
First Valid Address	172	31	140	1
Last Valid Address	172	31	140	126
Broadcast	172	31	140	127

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 128 = 128 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 0 is the multiple of 128 that is closest to 14 but not higher than 14. So, the fourth octet of the subnet number is 0.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 0 + 128 – 1 = 127.

Answer to Problem 17

Table F-50 Question 17: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	192.168.15.150	—
Mask	255.255.255.192	—
Number of network bits	24	Always defined by Class A, B, C
Number of host bits	6	Always defined as number of binary 0s in mask
Number of subnet bits	2	32 – (network size + host size)
Number of subnets	$2^2 = 4$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^6 – 2 = 62$	$2^{\text{number-of-host-bits}} – 2$

Table F-51 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-51 Question 17: Binary Calculation of Subnet and Broadcast Addresses

Address	192.168.15.150	`11000000 10101000 00001111 10`**`010110`**
Mask	255.255.255.192	`11111111 11111111 11111111 11`**`000000`**
AND result (subnet number)	192.168.15.128	`11000000 10101000 00001111 10`**`000000`**
Change host to 1s (broadcast address)	192.168.15.191	`11000000 10101000 00001111 10`**`111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

192.168.15.129 through 192.168.15.190

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-52 shows the work for this problem, with some explanation of the work following the table.

Table F-52 Question 17: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	192
Address	192	168	15	150
Subnet Number	192	168	15	128
First Valid Address	192	168	15	129
Last Valid Address	192	168	15	190
Broadcast	192	168	15	191

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 192 = 64 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 128 is the multiple of 64 that is closest to 150 but not higher than 150. So, the fourth octet of the subnet number is 128.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 128 + 64 – 1 = 191.

Answer to Problem 18

Table F-53 Question 18: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	192.168.15.150	—
Mask	255.255.255.224	—
Number of network bits	24	Always defined by Class A, B, C
Number of host bits	5	Always defined as number of binary 0s in mask
Number of subnet bits	3	32 – (network size + host size)
Number of subnets	$2^3 = 8$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^5 – 2 = 30$	$2^{\text{number-of-host-bits}} – 2$

Table F-54 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-54 Question 18: Binary Calculation of Subnet and Broadcast Addresses

Address	192.168.15.150	11000000 10101000 00001111 100**10110**
Mask	255.255.255.224	11111111 11111111 11111111 111**00000**
AND result (subnet number)	192.168.15.128	11000000 10101000 00001111 100**00000**
Change host to 1s (broadcast address)	192.168.15.159	11000000 10101000 00001111 100**11111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

192.168.15.129 through 192.168.15.158

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-55 shows the work for this problem, with some explanation of the work following the table.

Table F-55 Question 18: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	224
Address	192	168	15	150
Subnet Number	192	168	15	128
First Valid Address	192	168	15	129
Last Valid Address	192	168	15	158
Broadcast	192	168	15	159

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 224 = 32 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 128 is the multiple of 32 that is closest to 150 but not higher than 150. So, the fourth octet of the subnet number is 128.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 128 + 32 – 1 = 159.

Answer to Problem 19

Table F-56 Question 19: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	192.168.100.100	—
Mask	255.255.255.240	—
Number of network bits	24	Always defined by Class A, B, C
Number of host bits	4	Always defined as number of binary 0s in mask
Number of subnet bits	4	32 – (network size + host size)
Number of subnets	$2^4 = 16$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^4 – 2 = 14$	$2^{\text{number-of-host-bits}} – 2$

Table F-57 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-57 Question 19: Binary Calculation of Subnet and Broadcast Addresses

Address	192.168.100.100	`11000000 10101000 01100100 0110`**`0100`**
Mask	255.255.255.240	`11111111 11111111 11111111 1111`**`0000`**
AND result (subnet number)	192.168.100.96	`11000000 10101000 01100100 0110`**`0000`**
Change host to 1s (broadcast address)	192.168.100.111	`11000000 10101000 01100100 0110`**`1111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

192.168.100.97 through 192.168.100.110

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-58 shows the work for this problem, with some explanation of the work following the table.

Table F-58 Question 19: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	240
Address	192	168	100	100
Subnet Number	192	168	100	96
First Valid Address	192	168	100	97
Last Valid Address	192	168	100	110
Broadcast	192	168	100	111

F

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 240 = 16 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 96 is the multiple of 16 that is closest to 100 but not higher than 100. So, the fourth octet of the subnet number is 96.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 96 + 16 – 1 = 111.

Answer to Problem 20

Table F-59 Question 20: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	192.168.100.100	—
Mask	255.255.255.248	—
Number of network bits	24	Always defined by Class A, B, C
Number of host bits	3	Always defined as number of binary 0s in mask
Number of subnet bits	5	32 – (network size + host size)
Number of subnets	$2^5 = 32$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^3 – 2 = 6$	$2^{\text{number-of-host-bits}} – 2$

Table F-60 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-60 Question 20: Binary Calculation of Subnet and Broadcast Addresses

Address	192.168.100.100	11000000 10101000 01100100 01100**100**
Mask	255.255.255.248	11111111 11111111 11111111 11111**000**
AND result (subnet number)	192.168.100.96	11000000 10101000 01100100 01100**000**
Change host to 1s (broadcast address)	192.168.100.103	11000000 10101000 01100100 01100**111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

192.168.100.97 through 192.168.100.102

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-61 shows the work for this problem, with some explanation of the work following the table.

Table F-61 Question 20: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	248
Address	192	168	100	100
Subnet Number	192	168	100	96
First Valid Address	192	168	100	97
Last Valid Address	192	168	100	102
Broadcast	192	168	100	103

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 248 = 8 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 96 is the multiple of 8 that is closest to 100 but not higher than 100. So, the fourth octet of the subnet number is 96.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 96 + 8 – 1 = 103.

Answer to Problem 21

Table F-62 Question 21: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	192.168.15.230	—
Mask	255.255.255.252	—
Number of network bits	24	Always defined by Class A, B, C
Number of host bits	2	Always defined as number of binary 0s in mask
Number of subnet bits	6	32 – (network size + host size)
Number of subnets	$2^6 = 64$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^2 – 2 = 2$	$2^{\text{number-of-host-bits}} – 2$

Table F-63 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-63 Question 21: Binary Calculation of Subnet and Broadcast Addresses

Address	192.168.15.230	11000000 10101000 00001111 11100**10**
Mask	255.255.255.252	11111111 11111111 11111111 11111**00**
AND result (subnet number)	192.168.15.228	11000000 10101000 00001111 11100**00**
Change host to 1s (broadcast address)	192.168.15.231	11000000 10101000 00001111 11100**11**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

192.168.15.229 through 192.168.15.230

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-64 shows the work for this problem, with some explanation of the work following the table.

Table F-64 Question 21: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	252
Address	192	168	15	230
Subnet Number	192	168	15	228
First Valid Address	192	168	15	229
Last Valid Address	192	168	15	230
Broadcast	192	168	15	231

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 252 = 4 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 228 is the multiple of 4 that is closest to 230 but not higher than 230. So, the fourth octet of the subnet number is 228.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 228 + 4 – 1 = 231.

Answer to Problem 22

Table F-65 Question 22: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.1.1.1	—
Mask	255.248.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	19	Always defined as number of binary 0s in mask
Number of subnet bits	5	32 – (network size + host size)
Number of subnets	$2^5 = 32$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{19} - 2 = 524{,}286$	$2^{\text{number-of-host-bits}} - 2$

Table F-66 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-66 Question 22: Binary Calculation of Subnet and Broadcast Addresses

Address	10.1.1.1	`00001010 00000`**`001 00000001 00000001`**
Mask	255.248.0.0	`11111111 11111`**`000 00000000 00000000`**
AND result (subnet number)	10.0.0.0	`00001010 00000`**`000 00000000 00000000`**
Change host to 1s (broadcast address)	10.7.255.255	`00001010 00000`**`111 11111111 11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.0.0.1 through 10.7.255.254

Take a closer look at the subnet part of the subnet address, as shown in bold here: 0000 1010 **0000 0**000 0000 0000 0000 0000. The subnet part of the address is all binary 0s, making this subnet a zero subnet.

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-67 shows the work for this problem, with some explanation of the work following the table.

Table F-67 Question 22: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	248	0	0
Address	10	1	1	1
Subnet Number	10	0	0	0
First Valid Address	10	0	0	1
Last Valid Address	10	7	255	254
Broadcast	10	7	255	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The second octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 248 = 8 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 0 is the multiple of 8 that is closest to 1 but not higher than 1. So, the second octet of the subnet number is 0.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 0 + 8 – 1 = 7.

Answer to Problem 23

Table F-68 Question 23: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.16.1.200	—
Mask	255.255.240.0	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	12	Always defined as number of binary 0s in mask
Number of subnet bits	4	32 – (network size + host size)
Number of subnets	$2^4 = 16$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{12} – 2 = 4094$	$2^{\text{number-of-host-bits}} – 2$

Table F-69 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-69 Question 23: Binary Calculation of Subnet and Broadcast Addresses

Address	172.16.1.200	10101100 00010000 0000**0001 11001000**
Mask	255.255.240.0	11111111 11111111 1111**0000 00000000**
AND result (subnet number)	172.16.0.0	10101100 00010000 0000**0000 00000000**
Change host to 1s (broadcast address)	172.16.15.255	10101100 00010000 0000**1111 11111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.16.0.1 through 172.16.15.254

Take a closer look at the subnet part of the subnet address, as shown in bold here: 1010 1100 0001 0000 **0000** 0000 0000 0000. The subnet part of the address is all binary 0s, making this subnet a zero subnet.

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-70 shows the work for this problem, with some explanation of the work following the table.

Table F-70 Question 23: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	240	0
Address	172	16	1	200
Subnet Number	172	16	0	0
First Valid Address	172	16	0	1
Last Valid Address	172	16	15	254
Broadcast	172	16	15	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The third octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 240 = 16 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 0 is the multiple of 16 that is closest to 1 but not higher than 1. So, the third octet of the subnet number is 0.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 0 + 16 – 1 = 15.

Answer to Problem 24

Table F-71 Question 24: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	172.16.0.200	—
Mask	255.255.255.192	—
Number of network bits	16	Always defined by Class A, B, C
Number of host bits	6	Always defined as number of binary 0s in mask
Number of subnet bits	10	32 – (network size + host size)
Number of subnets	$2^{10} = 1024$	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^6 – 2 = 62$	$2^{\text{number-of-host-bits}} – 2$

Table F-72 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-72 Question 24: Binary Calculation of Subnet and Broadcast Addresses

Address	172.16.0.200	10101100 00010000 00000000 11**001000**
Mask	255.255.255.192	11111111 11111111 11111111 11**000000**
AND result (subnet number)	172.16.0.192	10101100 00010000 00000000 11**000000**
Change host to 1s (broadcast address)	172.16.0.255	10101100 00010000 00000000 11**111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

172.16.0.193 through 172.16.0.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-73 shows the work for this problem, with some explanation of the work following the table.

Table F-73 Question 24: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	255	255	192
Address	172	16	0	200
Subnet Number	172	16	0	192
First Valid Address	172	16	0	193
Last Valid Address	172	16	0	254
Broadcast	172	16	0	255

This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is "interesting" in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 192 = 64 in this case (256 – mask's value in the interesting octet). The subnet number's value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address's value in the interesting octet. In this case, 192 is the multiple of 64 that is closest to 200 but not higher than 200. So, the fourth octet of the subnet number is 192.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the "interesting" octet. Take the subnet number's value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address's value in the interesting octet. In this case, it is 192 + 64 – 1 = 255.

You can easily forget that the subnet part of this address, when using this mask, actually covers all the third octet as well as 2 bits of the fourth octet. For example, the valid subnet numbers in order are listed here:

172.16.0.0 (zero subnet)
172.16.0.64
172.16.0.128
172.16.0.192
172.16.1.0
172.16.1.64
172.16.1.128
172.16.1.192
172.16.2.0
172.16.2.64
172.16.2.128
172.16.2.192
172.16.3.0
172.16.3.64
172.16.3.128
172.16.3.192

And so on.

Answer to Problem 25

Congratulations! You made it through the extra practice in this appendix! Here is an easy one to complete your review—one with no subnetting at all.

Table F-74 Question 25: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.1.1.1	—
Mask	255.0.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	24	Always defined as number of binary 0s in mask
Number of subnet bits	0	32 – (network size + host size)
Number of subnets	0	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{24} - 2 = 16{,}777{,}214$	$2^{\text{number-of-host-bits}} - 2$

Table F-75 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-75 Question 25: Binary Calculation of Subnet and Broadcast Addresses

Address	10.1.1.1	`00001010` **`00000001 00000001 00000001`**
Mask	255.0.0.0	`11111111` **`00000000 00000000 00000000`**
AND result (subnet number)	10.0.0.0	`00001010` **`00000000 00000000 00000000`**
Change host to 1s (broadcast address)	10.255.255.255	`00001010` **`11111111 11111111 11111111`**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.0.0.1 through 10.255.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-76 shows the work for this problem.

Table F-76 Question 25: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	0	0	0
Address	10	1	1	1
Network Number	10	0	0	0
First Valid Address	10	0	0	1
Last Valid Address	10	255	255	254
Broadcast	10	255	255	255

APPENDIX G

Practice for Chapter 22: Fundamentals of IP Version 6

This appendix provides extra practice problems for two topics discussed in Chapter 22, "Fundamentals of IP Version 6," of the book. The first problems let you convert from a full 32-digit IPv6 address to its abbreviated form, or to do the reverse. The second set of problems begins with IPv6 addresses and prefix lengths, asking you to determine the IPv6 prefix (subnet).

Address Abbreviating and Expanding Problems

Chapter 22 discusses some reasons why you may need to be able to mentally convert from the full 32-digit IPv6 address to the abbreviated form, or vice versa. The practice problems in this section simply provide more opportunities to practice.

Table G-1 lists some practice problems, with the full 32-digit IPv6 address on the left and the best abbreviation on the right. The table gives you either the expanded or abbreviated address, and you need to supply the opposite value. The answers sit at the end of the appendix, in the section "Answers to Address Abbreviating and Expanding Problems."

Table G-1 IPv6 Address Abbreviation and Expansion Practice

	Full	Abbreviation
1	2987:BA11:B011:B00A:1000:0001:F001:F003	
2		3100::1010:D00D:D000:D00B:B00D
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	
4		FDDF:8080:880:1001:0:FF:FE01:507
5	32CC:0000:0000:000D:210F:0000:0000:0000	
6		2100:E:E0::E00
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	
8		3799:9F9F:F000:0:FFFF::1
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	
10		3194::1:0:0:101
11	2001:0DB8:0000:0000:0001:0000:0002:0100	
12		2001:DB8::10:A000
13	3330:0000:0000:0100:0000:0002:0000:0003	
14		FD00::1000:2000:0:1:20
15	FD11:1000:0100:0010:0001:0000:1000:0100	
16		2000::2

Calculating the IPv6 Prefix Problems

Routers take the interface IPv6 address configuration and add a connected IPv6 route to the IPv6 routing table, for the IPv6 prefix (subnet) connected to that interface. This section provides some practice problems so that you can do the same math and predict the prefix value that the router will add to the routing table.

Table G-2 lists practice problems that all use the same prefix length (/64), which is the most common prefix length you see. Table G-3 that follows lists additional practice problems, with prefix lengths other than /64.

Table G-2 Finding the IPv6 Prefix When Using a /64 Prefix Length

	Address (Assume a /64 Prefix Length)	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003	
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	
5	32CC:0000:0000:000D:210F:0000:0000:0000	
6	2100:000E:00E0:0000:0000:0000:0000:0E00	
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	
10	3194:0000:0000:0000:0001:0000:0000:0101	
11	2001:0DB8:0000:0000:0001:0000:0002:0100	
12	2001:0DB8:0000:0000:0000:0000:0010:A000	
13	3330:0000:0000:0100:0000:0002:0000:0003	
14	FD00:0000:0000:1000:2000:0000:0001:0020	
15	FD11:1000:0100:0010:0001:0000:1000:0100	
16	2000:0000:0000:0000:0000:0000:0000:0002	

Table G-3 Finding the IPv6 Prefix Using a Prefix Length Other Than /64

	Address	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003 /60	
2	3100:0000:0000:1010:D00D:D000:D00B:B00D /56	
3	FD00:0001:0001:0001:0200:00FF:FE00:0001 /52	
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507 /48	
5	32CC:0000:0000:000D:210F:0000:0000:0000 /44	
6	2100:000E:00E0:0000:0000:0000:0000:0E00 /60	
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C /56	
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001 /52	
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A /48	
10	3194:0000:0000:0000:0001:0000:0000:0101 /44	

Answers to Address Abbreviating and Expanding Problems

Table G-4 lists the answers to the problems listed earlier in Table G-1.

Table G-4 Answers: IPv6 Address Abbreviation and Expansion Practice

	Full	Abbreviation
1	2987:BA11:B011:B00A:1000:0001:F001:F003	2987:BA11:B011:B00A:1000:1:F001:F003
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	3100::1010:D00D:D000:D00B:B00D
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	FD00:1:1:1:200:FF:FE00:1
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	FDDF:8080:880:1001:0:FF:FE01:507
5	32CC:0000:0000:000D:210F:0000:0000:0000	32CC:0:0:D:210F::
6	2100:000E:00E0:0000:0000:0000:0000:0E00	2100:E:E0::E00
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	3A11:CA00::FF:FECC:C
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	3799:9F9F:F000:0:FFFF::1
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	2A2A::2A2A
10	3194:0000:0000:0000:0001:0000:0000:0101	3194::1:0:0:101
11	2001:0DB8:0000:0000:0001:0000:0002:0100	2001:DB8::1:0:2:100
12	2001:0DB8:0000:0000:0000:0000:0010:A000	2001:DB8::10:A000
13	3330:0000:0000:0100:0000:0002:0000:0003	3330::100:0:2:0:3
14	FD00:0000:0000:1000:2000:0000:0001:0020	FD00:.1000:2000:0:1:20
15	FD11:1000:0100:0010:0001:0000:1000:0100	FD11:1000:100:10:1:0:1000:100
16	2000:0000:0000:0000:0000:0000:0000:0002	2000::2

Answers to Calculating IPv6 Prefix Problems

Tables G-5 and G-6 list the answers to the problems listed earlier in Tables G-2 and G-3.

Table G-5 Answers: Finding the IPv6 Prefix, with a /64 Prefix Length

	Address (Assume a /64 Prefix Length)	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003	2987:BA11:B011:B00A::/64
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	3100:0:0:1010::/64
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	FD00:1:1:1::/64
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	FDDF:8080:880:1001::/64
5	32CC:0000:0000:000D:210F:0000:0000:0000	32CC:0:0:D::/64
6	2100:000E:00E0:0000:0000:0000:0000:0E00	2100:E:E0::/64
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	3A11:CA00::/64
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	3799:9F9F:F000::/64
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	2A2A::/64
10	3194:0000:0000:0000:0001:0000:0000:0101	3194::/64
11	2001:0DB8:0000:0000:0001:0000:0002:0100	2001:DB8::/64
12	2001:0DB8:0000:0000:0000:0000:0010:A000	2001:DB8::/64
13	3330:0000:0000:0100:0000:0002:0000:0003	3330:0:0:100::/64
14	FD00:0000:0000:1000:2000:0000:0001:0020	FD00:0:0:1000::/64
15	FD11:1000:0100:0010:0001:0000:1000:0100	FD11:1000:100:10::/64
16	2000:0000:0000:0000:0000:0000:0000:0002	2000::/64

Table G-6 Answers: Finding the IPv6 Prefix, with Other Prefix Lengths

	Address	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003 /60	2987:BA11:B011:B000::/60
2	3100:0000:0000:1010:D00D:D000:D00B:B00D /56	3100:0:0:1000::/56
3	FD00:0001:0001:0001:0200:00FF:FE00:0001 /52	FD00:1:1::/52
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507 /48	FDDF:8080:880::/48
5	32CC:0000:0000:000D:210F:0000:0000:0000 /44	32CC::/44
6	2100:000E:00E0:0000:0000:0000:0000:0E00 /60	2100:E:E0::/60
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C /56	3A11:CA00::/56
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001 /52	3799:9F9F:F000::/52
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A /48	2A2A::/48
10	3194:0000:0000:0000:0001:0000:0000:0101 /44	3194::/44

APPENDIX H

Practice for Chapter 24: Implementing IPv6 Addressing on Routers

This appendix provides practice problems for two types of addresses: unicast addresses formed with the EUI-64 feature and solicited node multicast addresses. With EUI-64, you take the 64-bit (16 hex digit) prefix and a MAC address, manipulate the MAC address into a 64-bit value, and use those 64 bits as the interface ID. Solicited node multicast addresses are formed from a standard 26 hex digit prefix, combined with the same last 6 hex digits as the unicast address.

EUI-64 and Solicited Node Multicast Problems

Table H-1 lists some practice problems. Each problem lists a prefix and a MAC address. Then, in Table H-2, record your answers for the unicast IPv6 address, assuming that EUI-64 rules are used. Also in Table H-2, list the solicited node multicast address associated with your calculated unicast address.

For each answer, use the best abbreviation, instead of a full 32-digit address.

The answers sit at the end of the appendix, in Table H-3.

Table H-1 IPv6 EUI-64 Unicast and Solicited Node Multicast Problems

	Prefix	MAC Address
1	2987:BA11:B011:B00A::/64	0000.1234.5678
2	3100:0000:0000:1010::/64	1234.5678.9ABC
3	FD00:0001:0001:0001::/64	0400.AAAA.0001
4	FDDF:8080:0880:1001::/64	0611.BABA.DADA
5	32CC:0000:0000:000D::/64	0000.0000.0001
6	2100:000E:00E0:0000::/64	0505.0505.0707
7	3A11:CA00:0000:0000::/64	0A0A.B0B0.0C0C
8	3799:9F9F:F000:0000::/64	F00F.0005.0041
9	2A2A:0000:0000:0000::/64	0200.0101.0101
10	3194:0000:0000:0000::/64	0C0C.000C.00CC

Table H-2 Blank Answer Table for Problems in Table H-1

	Unicast Address Using EUI-64	Solicited Node Multicast Address
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Answers to EUI-64 and Solicited Node Multicast Problems

Table H-3 lists the answers to the problems listed earlier in Table H-1.

Table H-3 Answers to Problems in Table H-1

	Unicast Address Using EUI-64	Solicited Node Multicast Address
1	2987:BA11:B011:B00A:200:12FF:FE34:5678	FF02::01:FF34.5678
2	3100::1010:1034:56FF:FE78:9ABC	FF02::01:FF78.9ABC
3	FD00:1:1:1:600:AAFF:FEAA:1	FF02::01:FFAA:1
4	FDDF:8080:880:1001:411:BAFF:FEBA:DADA	FF02::01:FFBA:DADA
5	32CC::D:200:FF:FE00:1	FF02::01:FF00:1
6	2100:E:E0:0:705:5FF:FE05:707	FF02::01:FF05:707
7	3A11:CA00::80A:B0FF:FEB0:C0C	FF02::01:FFB0:C0C
8	3799:9F9F:F000:0:F20F:FF:FE05:41	FF02::01:FF05:41
9	2A2A::1FF:FE01:101	FF02::01:FF01:101
10	3194::E0C:FF:FE0C:CC	FF02::01:FF0C:CC

Appendix I

Study Planner

Practice Test	Reading	Task

Element	Task	Goal Date	First Date Completed	Second Date Completed (Optional)	Notes
Introduction	Read Introduction				
Your Study Plan	Read Your Study Plan				
1. Introduction to TCP/IP Networking	Read Foundation Topics				
1. Introduction to TCP/IP Networking	Review Key Topics using the book or companion website				
1. Introduction to TCP/IP Networking	Define Key Terms using the book or companion website				
1. Introduction to TCP/IP Networking	Repeat DIKTA questions using the book or PTP exam engine				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
2. Fundamentals of Ethernet LANs	Read Foundation Topics				
2. Fundamentals of Ethernet LANs	Review Key Topics using the book or companion website				
2. Fundamentals of Ethernet LANs	Define Key Terms using the book or companion website				
2. Fundamentals of Ethernet LANs	Repeat DIKTA questions using the book or PTP exam engine				
2. Fundamentals of Ethernet LANs	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
3. Fundamentals of WANs and IP Routing	Read Foundation Topics				
3. Fundamentals of WANs and IP Routing	Review Key Topics using the book or companion website				

3. Fundamentals of WANs and IP Routing	Define Key Terms using the book or companion website				
3. Fundamentals of WANs and IP Routing	Repeat DIKTA questions using the book or PTP exam engine				
3. Fundamentals of WANs and IP Routing	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part I. Introduction to Networking	Complete all exercises in Part I Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
4. Using the Command-Line Interface	Read Foundation Topics				
4. Using the Command-Line Interface	Review Key Topics using the book or companion website				
4. Using the Command-Line Interface	Define Key Terms using the book or companion website				
4. Using the Command-Line Interface	Repeat DIKTA questions using the book or PTP exam engine				
4. Using the Command-Line Interface	Review the command tables				
4. Using the Command-Line Interface	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
5. Analyzing Ethernet LAN Switching	Read Foundation Topics				
5. Analyzing Ethernet LAN Switching	Review Key Topics using the book or companion website				
5. Analyzing Ethernet LAN Switching	Define Key Terms using the book or companion website				
5. Analyzing Ethernet LAN Switching	Repeat DIKTA questions using the book or PTP exam engine				
5. Analyzing Ethernet LAN Switching	Do labs listed for this chapter using the Sim Lite app				
5. Analyzing Ethernet LAN Switching	Review the command tables				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
6. Configuring Basic Switch Management	Read Foundation Topics				

6. Configuring Basic Switch Management	Review Key Topics using the book or companion website				
6. Configuring Basic Switch Management	Define Key Terms using the book or companion website				
6. Configuring Basic Switch Management	Repeat DIKTA questions using the book or PTP exam engine				
6. Configuring Basic Switch Management	Complete config checklists in this chapter using the companion website				
6. Configuring Basic Switch Management	Do labs listed for this chapter using the Sim Lite app				
6. Configuring Basic Switch Management	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
7. Configuring and Verifying Switch Interfaces	Read Foundation Topics				
7. Configuring and Verifying Switch Interfaces	Review Key Topics using the book or companion website				
7. Configuring and Verifying Switch Interfaces	Define Key Terms using the book or companion website				
7. Configuring and Verifying Switch Interfaces	Repeat DIKTA questions using the book or PTP exam engine				
7. Configuring and Verifying Switch Interfaces	Review command tables for this chapter				
7. Configuring and Verifying Switch Interfaces	Complete all memory tables in this chapter using the companion website				
7. Configuring and Verifying Switch Interfaces	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part II. Implementing Ethernet LANs	Complete all exercises in Part II Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
8. Implementing Ethernet Virtual LANs	Read Foundation Topics				
8. Implementing Ethernet Virtual LANs	Review Key Topics using the book or companion website				
8. Implementing Ethernet Virtual LANs	Define Key Terms using the book or companion website				
8. Implementing Ethernet Virtual LANs	Repeat DIKTA questions using the book or PTP exam engine				

8. Implementing Ethernet Virtual LANs	Complete config checklists in this chapter using the companion website				
8. Implementing Ethernet Virtual LANs	Review command tables for this chapter				
8. Implementing Ethernet Virtual LANs	Complete all memory tables in this chapter using the companion website				
8. Implementing Ethernet Virtual LANs	Do labs listed for this chapter using the Sim Lite app				
8. Implementing Ethernet Virtual LANs	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
9. Spanning Tree Protocol Concepts	Read Foundation Topics				
9. Spanning Tree Protocol Concepts	Review Key Topics using the book or companion website				
9. Spanning Tree Protocol Concepts	Define Key Terms using the book or companion website				
9. Spanning Tree Protocol Concepts	Repeat DIKTA questions using the book or PTP exam engine				
9. Spanning Tree Protocol Concepts	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
10. RSTP and EtherChannel Configuration	Read Foundation Topics				
10. RSTP and EtherChannel Configuration	Review Key Topics using the book or companion website				
10. RSTP and EtherChannel Configuration	Define Key Terms using the book or companion website				
10. RSTP and EtherChannel Configuration	Repeat DIKTA questions using the book or PTP exam engine				
10. RSTP and EtherChannel Configuration	Complete config checklists in this chapter using the companion website				
10. RSTP and EtherChannel Configuration	Review command tables for this chapter				
10. RSTP and EtherChannel Configuration	Complete all memory tables in this chapter using the companion website				
10. RSTP and EtherChannel Configuration	Do labs listed for this chapter using the Sim Lite app				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part III. Implementing VLANs and STP	Complete all exercises in Part III Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
11. Perspectives on IPv4 Subnetting	Read Foundation Topics				
11. Perspectives on IPv4 Subnetting	Review Key Topics using the book or companion website				
11. Perspectives on IPv4 Subnetting	Define Key Terms using the book or companion website				
11. Perspectives on IPv4 Subnetting	Repeat DIKTA questions using the book or PTP exam engine				
11. Perspectives on IPv4 Subnetting	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
12. Analyzing Classful IPv4 Networks	Read Foundation Topics				
12. Analyzing Classful IPv4 Networks	Review Key Topics using the book or companion website				
12. Analyzing Classful IPv4 Networks	Define Key Terms using the book or companion website				
12. Analyzing Classful IPv4 Networks	Repeat DIKTA questions using the book or PTP exam engine				
12. Analyzing Classful IPv4 Networks	Complete all memory tables in this chapter using the companion website				
12. Analyzing Classful IPv4 Networks	Practice analyzing classful IPv4 networks using Appendix D on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
13. Analyzing Subnet Masks	Read Foundation Topics				
13. Analyzing Subnet Masks	Review Key Topics using the book or companion website				
13. Analyzing Subnet Masks	Define Key Terms using the book or companion website				
13. Analyzing Subnet Masks	Repeat DIKTA questions using the book or PTP exam engine				

13. Analyzing Subnet Masks	Complete all memory tables in this chapter using the companion website				
13. Analyzing Subnet Masks	Practice analyzing subnet masks using Appendix E on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
14. Analyzing Existing Subnets	Read Foundation Topics				
14. Analyzing Existing Subnets	Review Key Topics using the book or companion website				
14. Analyzing Existing Subnets	Define Key Terms using the book or companion website				
14. Analyzing Existing Subnets	Repeat DIKTA questions using the book or PTP exam engine				
14. Analyzing Existing Subnets	Complete all memory tables in this chapter using the companion website				
14. Analyzing Existing Subnets	Practice mask analysis using Appendix F on the companion website				
14. Analyzing Existing Subnets	Practice analyzing existing subnets using Appendix F on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part IV. IPv4 Addressing	Complete all exercises in Part IV Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
15. Operating Cisco Routers	Read Foundation Topics				
15. Operating Cisco Routers	Review Key Topics using the book or companion website				
15. Operating Cisco Routers	Define Key Terms using the book or companion website				
15. Operating Cisco Routers	Repeat DIKTA questions using the book or PTP exam engine				
15. Operating Cisco Routers	Review command tables for this chapter				
15. Operating Cisco Routers	Complete all memory tables in this chapter using the companion website				
15. Operating Cisco Routers	Do labs listed for this chapter using the Sim Lite app				
15. Operating Cisco Routers	Watch video for this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
16. Configuring IPv4 Addresses and Static Routes	Read Foundation Topics				
16. Configuring IPv4 Addresses and Static Routes	Review Key Topics using the book or companion website				
16. Configuring IPv4 Addresses and Static Routes	Define Key Terms using the book or companion website				
16. Configuring IPv4 Addresses and Static Routes	Repeat DIKTA questions using the book or PTP exam engine				
16. Configuring IPv4 Addresses and Static Routes	Review command tables for this chapter				
16. Configuring IPv4 Addresses and Static Routes	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
17. IP Routing in the LAN	Read Foundation Topics				
17. IP Routing in the LAN	Review Key Topics using the book or companion website				
17. IP Routing in the LAN	Define Key Terms using the book or companion website				
17. IP Routing in the LAN	Repeat DIKTA questions using the book or PTP exam engine				
17. IP Routing in the LAN	Complete config checklists in this chapter using the companion website				
17. IP Routing in the LAN	Review command tables for this chapter				
17. IP Routing in the LAN	Do labs listed for this chapter using the Sim Lite app				
17. IP Routing in the LAN	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
18. Troubleshooting IPv4 Routing	Read Foundation Topics				
18. Troubleshooting IPv4 Routing	Review Key Topics using the book or companion website				
18. Troubleshooting IPv4 Routing	Define Key Terms using the book or companion website				
18. Troubleshooting IPv4 Routing	Watch video for this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part V. IPv4 Routing	Complete all exercises in Part V Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
19. Understanding OSPF Concepts	Read Foundation Topics				
19. Understanding OSPF Concepts	Review Key Topics using the book or companion website				
19. Understanding OSPF Concepts	Define Key Terms using the book or companion website				
19. Understanding OSPF Concepts	Repeat DIKTA questions using the book or PTP exam engine				
19. Understanding OSPF Concepts	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
20. Implementing OSPF	Read Foundation Topics				
20. Implementing OSPF	Review Key Topics using the book or companion website				
20. Implementing OSPF	Define Key Terms using the book or companion website				
20. Implementing OSPF	Repeat DIKTA questions using the book or PTP exam engine				
20. Implementing OSPF	Complete config checklists in this chapter using the companion website				
20. Implementing OSPF	Review command tables for this chapter				
20. Implementing OSPF	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
21. OSPF Network Types and Neighbors	Read Foundation Topics				
21. OSPF Network Types and Neighbors	Review Key Topics using the book or companion website				
21. OSPF Network Types and Neighbors	Repeat DIKTA questions using the book or PTP exam engine				

21. OSPF Network Types and Neighbors	Complete all memory tables in this chapter using the companion website				
21. OSPF Network Types and Neighbors	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VI. OSPF	Complete all exercises in Part VI Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
22. Fundamentals of IP Version 6	Read Foundation Topics				
22. Fundamentals of IP Version 6	Review Key Topics using the book or companion website				
22. Fundamentals of IP Version 6	Define Key Terms using the book or companion website				
22. Fundamentals of IP Version 6	Repeat DIKTA questions using the book or PTP exam engine				
22. Fundamentals of IP Version 6	Review command tables for this chapter				
22. Fundamentals of IP Version 6	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
23. IPv6 Addressing and Subnetting	Read Foundation Topics				
23. IPv6 Addressing and Subnetting	Review Key Topics using the book or companion website				
23. IPv6 Addressing and Subnetting	Define Key Terms using the book or companion website				
23. IPv6 Addressing and Subnetting	Repeat DIKTA questions using the book or PTP exam engine				
23. IPv6 Addressing and Subnetting	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
24. Implementing IPv6 Addressing on Routers	Read Foundation Topics				
24. Implementing IPv6 Addressing on Routers	Review Key Topics using the book or companion website				
24. Implementing IPv6 Addressing on Routers	Define Key Terms using the book or companion website				

24. Implementing IPv6 Addressing on Routers	Repeat DIKTA questions using the book or PTP exam engine				
24. Implementing IPv6 Addressing on Routers	Review command tables for this chapter				
24. Implementing IPv6 Addressing on Routers	Complete all memory tables in this chapter using the companion website				
24. Implementing IPv6 Addressing on Routers	Do labs listed for this chapter using the Sim Lite app				
24. Implementing IPv6 Addressing on Routers	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
25. Implementing IPv6 Routing	Read Foundation Topics				
25. Implementing IPv6 Routing	Review Key Topics using the book or companion website				
25. Implementing IPv6 Routing	Define Key Terms using the book or companion website				
25. Implementing IPv6 Routing	Repeat DIKTA questions using the book or PTP exam engine				
25. Implementing IPv6 Routing	Review command tables for this chapter				
25. Implementing IPv6 Routing	Complete all memory tables in this chapter using the companion website				
25. Implementing IPv6 Routing	Do labs listed for this chapter using the author's blog site				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VII. IP Version 6	Complete all exercises in Part VII Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
26. Fundamentals of Wireless Networks	Read Foundation Topics				
26. Fundamentals of Wireless Networks	Review Key Topics using the book or companion website				
26. Fundamentals of Wireless Networks	Define Key Terms using the book or companion website				
26. Fundamentals of Wireless Networks	Repeat DIKTA questions using the book or PTP exam engine				
26. Fundamentals of Wireless Networks	Complete all memory tables in this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
27. Analyzing Cisco Wireless Architectures	Read Foundation Topics				
27. Analyzing Cisco Wireless Architectures	Review Key Topics using the book or companion website				
27. Analyzing Cisco Wireless Architectures	Define Key Terms using the book or companion website				
27. Analyzing Cisco Wireless Architectures	Repeat DIKTA questions using the book or PTP exam engine				
27. Analyzing Cisco Wireless Architectures	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
28. Securing Wireless Networks	Read Foundation Topics				
28. Securing Wireless Networks	Review Key Topics using the book or companion website				
28. Securing Wireless Networks	Define Key Terms using the book or companion website				
28. Securing Wireless Networks	Repeat DIKTA questions using the book or PTP exam engine				
28. Securing Wireless Networks	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
29. Building a Wireless LAN	Read Foundation Topics				
29. Building a Wireless LAN	Review Key Topics using the book or companion website				
29. Building a Wireless LAN	Define Key Terms using the book or companion website				
29. Building a Wireless LAN	Repeat DIKTA questions using the book or PTP exam engine				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VIII. Wireless LANs	Complete all exercises in Part VII Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				

Final Review	Take practice test in study mode for all Book Questions in practice test software				
Final Review	Review all Key Topics in all chapters or in the Key Topics App using the companion website				
Final Review	Review all Key Terms in all chapters or using the Key Terms Flashcards on the companion website				
Final Review	Complete all memory tables for all chapters using the companion website				
Final Review	Take practice test in practice exam mode using Exam Bank #1 questions for all chapters				
Final Review	Take practice test in practice exam mode using Exam Bank #2 questions for all chapters				

APPENDIX J

Topics from Previous Editions

Cisco changes the exams, renaming the exams on occasion, and changing the exam numbers every time it changes the exam with a new blueprint, even with a few name changes over the years. As a result, the current CCNA 200-301 exam serves as the eighth separate version of CCNA in its 20-plus year history. At every change to the exams, we create new editions of the books to match the new exam.

We base the books' contents on Cisco's exam topics; that is, the book attempts to cover the topics Cisco lists as exam topics. However, the book authoring process does create some challenges, particularly with the balance of what to include in the books and what to leave out.

For instance, when comparing a new exam to the old, I found Cisco had removed some topics—and I might want to keep the content in the book. There are a few reasons why. Sometimes I just expect that some readers will still want to read about that technology. Also, more than a few schools use these books as textbooks, and keeping some of the older-but-still-relevant topics can be a help. And keeping the old material available on each book's companion website takes only a little extra work, so we do just that.

Some of the older topics that I choose to keep on the companion website are small, so I collect them into this appendix. Other topics happen to have been an entire chapter in a previous edition of the books, so we include those topics each as a separate appendix. Regardless, the material exists here in this appendix, and in the appendices that follow, for your use if you have a need. But do not feel like you have to read this appendix for the current exam.

The topics in this appendix are as follows:

- IPv4 Address Types
- Bandwidth and Clock Rate on Serial Interfaces
- Using traceroute to Isolate the Problem to Two Routers
- Troubleshooting Static IPv6 Routes
- Default Routes with SLAAC on Router Interfaces

NOTE The content under the heading "IPv4 Address Types" was most recently published for the 100-105 Exam in 2016, in Chapter 20 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

IPv4 Address Types

The IPv4 address space includes three major categories of addresses: unicast, broadcast, and multicast. For the current exam, Cisco lists one exam topic that asks you to compare and contrast these address types. To help you make those comparisons, this section explains multicast addressing, while pulling together the key ideas about unicast and broadcast IP addresses that have already been introduced, to pull the ideas together.

You may be wondering why this topic about IPv4 address types sits at the end of a chapter about DHCP and IP networking on hosts. Honestly, I could have put this topic in several chapters. The main reason it is here is that you have already seen the IP broadcast addresses in action, including the 255.255.255.255 local broadcast as shown in this chapter.

Review of Unicast (Class A, B, and C) IP Addresses

Unicast IP addresses are those Class A, B, and C IP addresses assigned to hosts, router interfaces, and other networking devices. Because most discussions about IP addressing refer to unicast IP addresses, most of us just refer to them as IP addresses, and leave out the word *unicast*.

Just to be complete and define the concept, unicast addresses identify one interface on one device to IP. Just like your postal address gives the post office an address to use to send letters to your one specific house or apartment, a unicast IP address gives the IP network an address to use to send packets to one specific host. However, with IP, instead of addressing the device, unicast addresses identify individual interfaces. For example:

- A router with four LAN interfaces, and two WAN interfaces, has six unicast addresses, each in a different subnet, one for each interface.
- A PC with both an Ethernet network interface card (NIC) and a wireless NIC would have two unicast IPv4 addresses, one for each interface.

IP Broadcast Addresses

Broadcast IPv4 addresses give IP a way to send one packet that the network delivers to multiple hosts. IPv4 defines several types of broadcast addresses, with each type being used to reach a different set of hosts. These different broadcast IP addresses give different overhead protocols like DHCP the ability to efficiently reach all hosts in a specific part of the network. The following list reviews the three IP broadcast address types:

Key Topic

Local broadcast address: 255.255.255.255. Used to send a packet on a local subnet, knowing that routers will not forward the packet as is. Also called a *limited broadcast*.

Subnet broadcast address: One reserved address for each subnet, namely the numerically highest number in the subnet, as discussed in Chapter 13, "Analyzing Subnet Masks." A packet sent to a subnet broadcast address can be routed to the router connected to that

subnet, and then sent as a data-link broadcast to all hosts in that one subnet. Also called an *all-hosts broadcast* to emphasize that all hosts in a subnet are reached, and also called a *directed broadcast*.

Network broadcast address: One reserved address for each classful network, namely the numerically highest number in the network. Used to send one packet to all hosts in that one network. Also called an *all-subnets broadcast*, referring to the fact that the packet reaches all subnets in a network.

This chapter has already shown how a local broadcast works, sending the message over the same subnet in which it was first transmitted, but no further. However, the other two types are a little more interesting.

Subnet and network broadcasts provide a way to send packets to all hosts in a subnet or network (respectively) while reducing waste. For instance, with a subnet broadcast, routers forward the packet just like any other IP packet going to that subnet. When that packet arrives at the router connected to that subnet, the last router then encapsulates the packet in a LAN broadcast, so that all hosts receive a copy. Figure J-1 shows the idea.

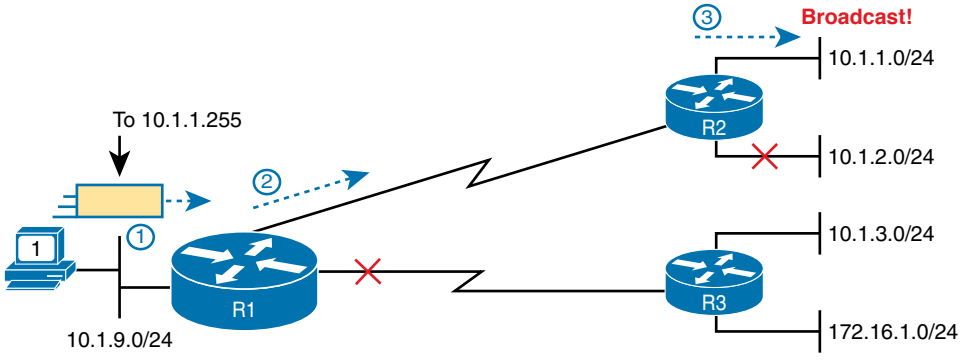


Figure J-1 *Example of a Subnet Broadcast to 10.1.1.255*

The figure shows two key points. R1 does not flood or broadcast the frame to all other routers, instead routing it to the next router (R2 in this case) so that the packet reaches subnet 10.1.1.0/24. R2, connected to subnet 10.1.1.0/24, forwards the packet onto the LAN, but encapsulates the packet in an Ethernet broadcast frame, so that it reaches all hosts in the subnet.

The figure shows the intended use of the subnet broadcast address; however, it presents a security issue today. Many attacks start with a ping to subnet broadcast addresses, hoping to get many hosts to reply. Cisco changed the IOS default many years ago to disable the forwarding of subnet broadcasts onto a connected subnet (that is, it disables Step 3 in Figure J-1). That default setting is based on the **no ip directed-broadcast** interface subcommand.

A network broadcast packet (a packet with a network broadcast address as the destination) works in a similar way. To reach all subnets, however, the routers create copies of the packet and flood it so it reaches all subnets inside the classful network. On any LAN interfaces, the packet is forwarded in a LAN broadcast, just as shown in Step 3 of Figure J-1.

IPv4 Multicast Addresses (Class D Addresses)

Multicast IP addresses and the related protocols help solve a similar problem as compared to broadcast addresses, but mainly for applications, and without the same security issues experienced by broadcast addresses. To see how it works, consider this example. A video application may be designed to show live video feeds. If 10 people at the same remote site in the same subnet want to watch the same video at the same time, the application could be designed so that the application sent the same video data 10 times, once to each client in the same subnet. An application designed to use Class D multicast addresses could send 1 packet, which the routers would route across the WAN, and then deliver a copy to all 10 hosts in the destination subnet.

When using multicast, all the hosts still use their individual unicast IP address for their normal traffic, while also using the same multicast IPv4 address for the multicast application. Any server or client that happens to use an application designed to take advantage of IP multicast then also uses the Class D multicast addresses that the application chooses to use. You can think of a Class D address more as a multicast group—in fact, it is often called that—because hosts join the group so that they can receive the packets sent by the multicast application.

Class D addresses begin with a first octet of between 224 and 239, with some ranges reserved for various purposes. Much of the Class D address space is set aside for a company to deploy one of these multicast applications, and then pick an address from the Class D range, and configure it to be used by a multicast application.

As an example, imagine the video application uses Class D address 226.1.1.1. Figure J-2 illustrates the process by which the application at the server on the left sends one multicast packet with destination address 226.1.1.1. Note that for this process to work, the hosts with * beside them registered with their local routers to notify the routers that the host wants to receive packets destined to multicast address 226.1.1.1. When the action in this figure begins, the routers collectively know which subnets have hosts that want a copy of multicasts sent to 226.1.1.1, and which subnets do not.

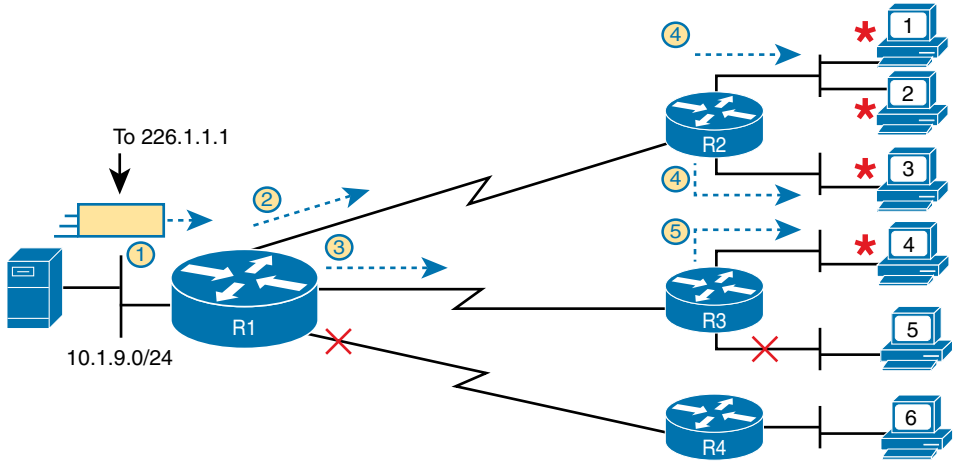


Figure J-2 *Example of a Multicast Packet Flow for Three Registered Hosts*

Following the steps in the figure:

1. The server on the left generates and sends a multicast packet.
2. Router R1 replicates the packet to send a copy to both R2…
3. …and to R3. R1 does not replicate and send a copy to R4, because there are no hosts near R4 listening for packets sent to 226.1.1.1.
4. R2 processes the multicast packet received from R1, and because of the earlier host registration process, R2 knows that at least one host off both its LAN interfaces are listening for packets sent to 226.1.1.1. R2 therefore forwards a copy of the packet out each of its LAN interfaces.
5. R3 receives the multicast packet from R1, and uses the same kind of logic as R2. However, R3 knows from the earlier host registration process that only one of its LAN interfaces connects to a subnet with hosts listening for packets sent to 226.1.1.1, so R3 forwards a copy of the packet out that one interface only.

As you can see from this example, the server sent one packet and the routers replicated the packet so it reached all the correct locations in the network.

As another comparison between unicast and multicast addresses, note that multicast addresses may be used as destination IP addresses only, whereas unicast addresses may be used as both the destination and source address. For instance, consider the packets in the example shown in Figure J-2. All those packets flow from one host, so the packet uses a unicast IP address of that host's unicast IP address.

Finally, to complete one more comparison between unicast IP addressing and multicast IP addressing, think about that last hop router in the example shown in Figure J-1. If a router such as R2 or R3 had forwarded a unicast IP packet, the router would look in its ARP cache to find the unicast IP address for the destination in that connected subnets, and the associated unicast MAC address. That will not work when forwarding a multicast packet with a multicast (Class D) destination IP address.

To encapsulate a multicast IP packet over an Ethernet LAN, IP multicast calculates the destination MAC address with a simple process. The process copies the last 23 bits of the IP address behind a reserved 25-bit prefix to form the 48-bit destination MAC address. The resulting MAC address, called a multicast MAC address, begins with hex 01005E. So, the multicast IP packet, encapsulated in the multicast Ethernet frame, is forwarded out the router interface onto the LAN. At that point, the switches take one of the following approaches to forwarding the frame so that all hosts who want a copy of the frame get a copy:

- Flood the multicast frame as if it were a broadcast
- Use other Ethernet multicast features that flood the frame only to those same devices that registered to receive a copy

If you feel like these few pages probably left out some detail; indeed, several books have been written about IP multicast all to itself. The topic is indeed large. For this book's purposes, know the main comparison points with unicast addressing. Multicast addressing gives applications that need to communicate the same data at the same time to multiple hosts a much more efficient way to do that. If the application is written to make use of IP multicast,

the application can consume much less traffic in the network, as compared to using unicast IP addresses and sending every host a copy of the packet.

Comparing and Contrasting IP Address Types

The last few pages reviewed unicast and broadcast addresses, and explained the core concepts behind IP multicast addresses. Table J-1 summarizes the key comparison points mentioned throughout this section for convenient study.

Table J-1 Comparisons of Unicast, Broadcast, and Multicast IP Addresses

	Unicast	Broadcast	Multicast
Primarily used for data sent by the most common user apps (web, email, chat, and so on)	Yes	No	No
Assigned to hosts with DHCP	Yes	No	No
Uses Class A, B, and C addresses	Yes	No	No
Primarily used by overhead protocols (DHCP, ARP) to send one message to more than one device	No	Yes	No
Used as destination IP address only	No	Yes	Yes
Primarily used by applications that send the same data at the same time to multiple clients	No	No	Yes
Uses Class D addresses	No	No	Yes

NOTE The content under the heading "Bandwidth and Clock Rate on Serial Interfaces" was most recently published for the 100-105 Exam in 2016, in Chapter 17 of the *CCENT/CCNA ICND1 100-105 Official Cert Guide*.

Bandwidth and Clock Rate on Serial Interfaces

WAN serial links can run at a wide variety of speeds. To deal with the wide range of speeds, routers physically slave themselves to the speed as dictated by the CSU/DSU through a process called *clocking*. As a result, routers can use serial links without the need for additional configuration or autonegotiation to sense the serial link's speed. The CSU/DSU knows the speed, the CSU/DSU sends clock pulses over the cable to the router, and the router reacts to the clocking signal.

To build a serial link in a home lab, the routers can use serial interface cards that normally use an external CSU/DSU, and make a serial link, without requiring the expense of two CSU/DSUs. Figure J-3 shows the concept. To make it work, the link uses two serial cables—one a DTE cable and the other a DCE cable—which swap the transmit and receive pair on the cables.

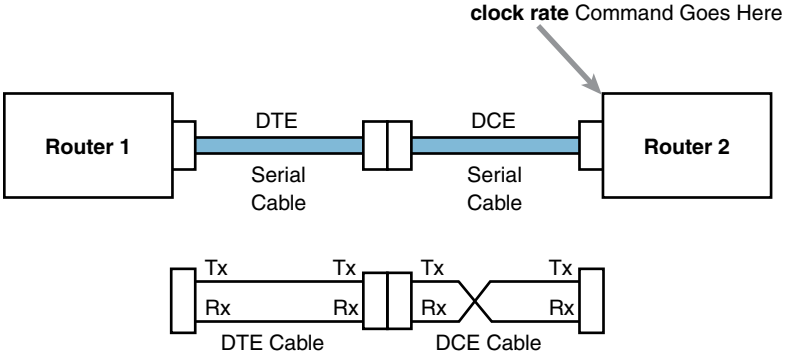


Figure J-3 *Serial Link in Lab*

Using the correct cabling works, as long as you add one command: the **clock rate** interface subcommand. This command tells that router the speed at which to transmit bits on a serial link like the one shown in Figure J-3. The **clock rate** command is not needed on real serial links, because the CSU/DSU provides the clocking. When you create a serial link in the lab using cables, without any real CSU/DSUs on the link, the router with the DCE cable must supply that clocking function, and the **clock rate** command tells the router to provide it.

NOTE Newer router IOS versions automatically add a default **clock rate 2000000** command on serial interfaces that have a DCE cable connected to them. While helpful, this speed might be too high for some types of back-to-back serial cables, so consider using a lower speed in lab.

Example J-1 shows the configuration of the **clock rate** command. The end of the example verifies that this router can use the **clock rate** command with the **show controllers** command. This command confirms that R1 has a V.35 DCE cable connected.

Example J-1 *Router R1 Configuration with the* **clock rate** *Command*

```
R1# show running-config
! lines omitted for brevity
interface Serial0/0/0
   ip address 172.16.4.1 255.255.255.0
   clock rate 2000000
!
interface Serial0/0/1
   ip address 172.16.5.1 255.255.255.0
   clock rate 128000

! lines omitted for brevity

R1# show controllers serial 0/0/1
Interface Serial0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 128000
idb at 0x8169BB20, driver data structure at 0x816A35E4
! Lines omitted for brevity
```

NOTE The **clock rate** command does not allow just any speed to be configured. However, the list of speeds does vary from router to router.

Some people confuse the router **bandwidth** command with the **clock rate** command. The **clock rate** command sets the actual Layer 1 speed used on the link, if no CSU/DSU is used, as just described. Conversely, every router interface has a bandwidth setting, either by default or configured. The bandwidth of the interface is the documented speed of the interface, which does not have to match the actual Layer 1 speed used on the interface.

That bandwidth setting does not impact how fast the interface transmits data. Instead, routers use the interface bandwidth setting as both documentation and as input to some other processes. For instance, the Open Shortest Path First (OSPF) and Enhanced Interior Gateway Routing Protocol (EIGRP) routing protocols base their routing protocol metrics on the bandwidth by default.

Example J-2 highlights the bandwidth setting on Router R1's S0/0/1 interface, as configured in the previous example. In that previous example, the **clock rate 128000** command sets the clock rate to 128 kbps, but it leaves the **bandwidth** command unset. As a result, IOS uses the default serial bandwidth setting of 1544, which means 1544 kbps—which is the speed of a T1 serial link.

Example J-2 *Router Bandwidth Settings*

```
R1# show interfaces s0/0/1
Serial0/0/1 is up, line protocol is up
   Hardware is WIC MBRD Serial
   Description: link to R3
   Internet address is 10.1.13.1/24
   MTU 1500 bytes, BW 1544 Kbit/sec, DLY 20000 usec,
      reliability 255/255, txload 1/255, rxload 1/255
   Encapsulation HDLC, loopback not set
```

The common mistake people make is to know about clock rate, but mistakenly think that the bandwidth setting is just another term for "clock rate." It is not. Follow these rules to find these two interface settings:

To see the clock rate, look for the **clock rate** interface subcommand in the configuration, or use the **show controllers serial** *number* command (as shown in Example J-1.)

To see the bandwidth setting on an interface, look for the **bandwidth** interface subcommand in the configuration, or use the **show interfaces** [*type number*] command (as shown in Example J-2).

Note that using default bandwidth settings on most router interfaces makes sense, with the exception of serial interfaces. IOS defaults to a bandwidth of 1544 (meaning 1544 kbps, or 1.544 Mbps) for serial interfaces, regardless of the speed dictated by the provider or by a **clock rate** command in the lab. Most engineers set the bandwidth to match the actual speed, for example, using the **bandwidth 128** interface subcommand on a link running at 128 kbps. On Ethernet 10/100 or 10/100/1000 interfaces, the router knows the speed used, and dynamically sets the Ethernet interface's bandwidth to match.

NOTE The content under the heading "Using traceroute to Isolate the Problem to Two Routers" was most recently published for the 100-105 Exam in 2016, in Chapter 23 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Using traceroute to Isolate the Problem to Two Routers

One of the best features of the **traceroute** command, as compared to ping, is that when it does not complete it gives an immediate clue as to where to look next. With ping, when the ping fails, the next step is usually to use more **ping** commands. With traceroute, it tells you what router to try to connect and look at the routes and in which direction.

NOTE As a reminder, this book uses the term *forward route* for routes that send the packets sent by the **ping** or **traceroute** command, and *reverse route* for the packets sent back.

When a problem exists, a **traceroute** command results in a partial list of routers. Then the command either finishes with an incomplete list or it runs until the user must stop the command. In either case, the output does not list all routers in the end-to-end route, because of the underlying problem.

NOTE In addition, the **traceroute** command may not finish even though the network has no problems. Routers and firewalls may filter the messages sent by the **traceroute** command, or the TTL Exceeded messages, which would prevent the display of portions or all or part of the path.

The last router listed in the output of a **traceroute** command's output tells us where to look next to isolate the problem, as follows:

- Connect to the CLI of the last router listed, to look at forward route issues.
- Connect to the CLI of the next router that should have been listed, to look for reverse route issues.

To see why, consider an example based on the internetwork in Figure J-4. In this case, R1 uses an extended traceroute to host 5.5.5.5, with source IP address 1.1.1.1. This command's output lists router 2.2.2.2, then 3.3.3.3, and then the command cannot complete.

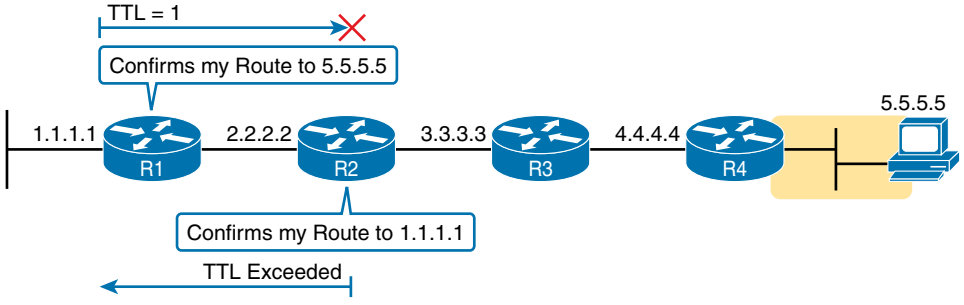


Figure J-4 *Messages That Cause the* **traceroute** *Command to List 2.2.2.2*

First, Figure J-4 focuses on the first line of output: the line that lists first-hop router 2.2.2.2.

The figure shows the TTL=1 message at the top and the TTL Exceeded message back on the bottom. This first pair of messages in the figure must have worked, because without them, the **traceroute** command on R1 cannot have learned about a router with address 2.2.2.2. The first (top) message required R1 to have a route for 5.5.5.5, which sent the packets to R2 next. The TTL Exceeded message required that R2 have a route that matched address 1.1.1.1, to send the packets back to R1's LAN IP address.

Next, Figure J-5 focuses on the messages that allow the second line of output on R1's sample **traceroute** command: the line that correctly lists 3.3.3.3 as the next router in the route.

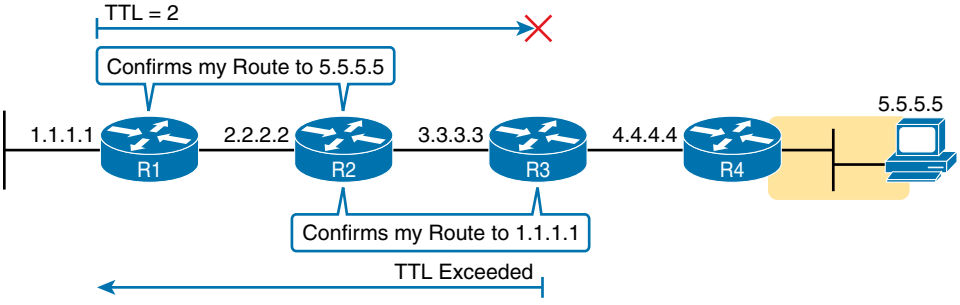


Figure J-5 *Messages That Cause the* **traceroute** *Command to List 3.3.3.3*

Following the same logic, the traceroute output lists 3.3.3.3 because the messages in Figure J-5 must have worked. For these messages to flow, the routes listed in Figure J-4 must exist, plus new routes listed in 18-15. Specifically, the TTL=2 packet at the top requires R2 to have a route for 5.5.5.5, which sends the packets to R3 next. The TTL Exceeded message requires that R3 have a route that matches address 1.1.1.1, to send the packets back toward R1's LAN IP address.

In this example, the **traceroute 5.5.5.5** command does not list any routers beyond 2.2.2.2 and 3.3.3.3 However, based on the figures, it is clear that 4.4.4.4 should be the next IP address listed. To help isolate the problem further, why might the next messages—the message with TTL=3 and the response—fail?

Figure J-6 points out the routing issues that can cause this command to not be able to list 4.4.4.4 as the next router. First, R3 must have a forward route matching destination 5.5.5.5 and forwarding the packet to Router R4. The return message requires a reverse route matching destination 1.1.1.1 and forwarding the packet back to Router R3.

In conclusion, for this example, if a routing problem prevents the **traceroute** command from working, the problem exists in one of two places: the forward route to 5.5.5.5 on Router R3, or the reverse route to 1.1.1.1 on R4.

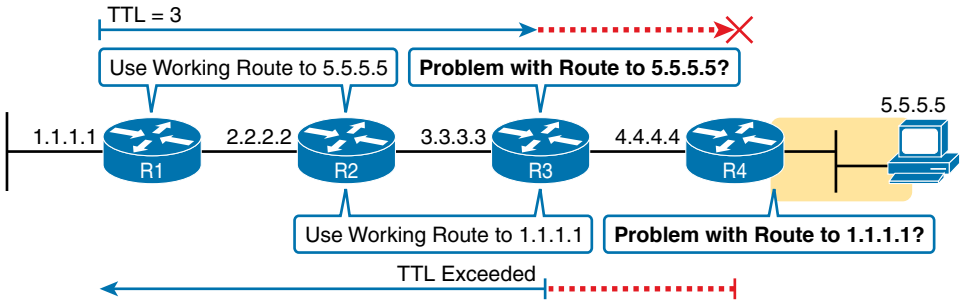


Figure J-6 *Issues That Could Prevent* **traceroute** *from Listing 4.4.4.4*

NOTE The content under the heading "Troubleshooting Static IPv6 Routes" was most recently published in Chapter 32 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Troubleshooting Static IPv6 Routes

This last part of the chapter looks at troubleshooting IPv6 static routes, reviewing many of the same troubleshooting rules applied to IPv4 static routes, while focusing on the details specific to IPv6.

This topic breaks static route troubleshooting into two perspectives: the route is in the routing table but is incorrect and cases in which the route is not in the routing table.

Troubleshooting Incorrect Static Routes That Appear in the IPv6 Routing Table

A static route is only as good as the input typed into the **ipv6 route** command. IOS checks the syntax of the command, of course. However, IOS cannot tell if you choose the incorrect outgoing interface, incorrect next-hop address, or incorrect prefix/prefix-length in a static route. If the parameters pass the syntax checks, IOS places the **ipv6 route** command into the running-config file. Then, if no other problem exists (as discussed at the next heading), IOS puts the route into the IP routing table—even though the route may not work because of the poorly chosen parameters.

For instance, an exam question might show a figure with Router R1 having an address of 2001:1:1:1::1 and neighboring Router R2 with an address of 2001:1:1:1::2. If R1 lists a static route with the command **ipv6 route 3333::/64 2001:1:1:1::1**, the command would be accepted by IOS with correct syntax, but it would not be effective as a route. R1 cannot use its own IPv6 address as a next-hop address. IOS does not prevent the configuration of the command, however; it allows the command and adds the route to the IPv6 routing table, but the route cannot possibly forward packets correctly.

When you see an exam question that has static routes, and you see them in the output of **show ipv6 route**, remember that the routes may have incorrect parameters. Check for these types of mistakes:

Step 1. Prefix/Length: Does the **ipv6 route** command reference the correct subnet ID (prefix) and mask (prefix length)?

Step 2. If using a next-hop IPv6 address that is a link-local address:

- **A.** Is the link-local address an address on the correct neighboring router? (It should be an address on another router on a shared link.)
- **B.** Does the **ipv6 route** command also refer to the correct outgoing interface on the local router?

Step 3. If using a next-hop IPv6 address that is a global unicast or unique local address, is the address the correct unicast address of the neighboring router?

Step 4. If referencing an outgoing interface, does the **ipv6 route** command reference the interface on the local router (that is, the same router where the static route is configured)?

This troubleshooting checklist works through the various cases in which IOS would accept the configuration of the static IPv6 route, but the route would not work because of the incorrect parameters in context. It helps to see a few examples. Figure J-7 shows a sample network to use for the examples; all the examples focus on routes added to Router R1, for the subnet on the far right.

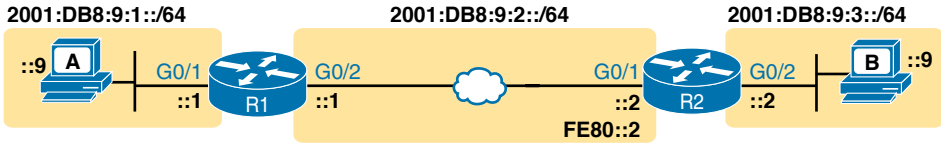


Figure J-7 *Sample Topology for Incorrect IPv6 Route Examples*

Example J-3 shows five **ipv6 route** commands. All have correct syntax, but all have one incorrect value; that is, the route will not work because of the types of problems in the troubleshooting checklist. Look for the short comment at the end of each configuration command to see why each is incorrect.

Example J-3 **ipv6 route** *Commands with Correct Syntax but Incorrect Ideas*

```
ipv6 route 2001:DB8:9:33::/64 2001:DB8:9:2::2 ! Step 1: Wrong prefix
ipv6 route 2001:DB8:9:3::/64 G0/2 FE80::AAA9 ! Step 2A: Wrong neighbor link local
ipv6 route 2001:DB8:9:3::/64 FE80::2 ! Step 2B: Missing outgoing interface
ipv6 route 2001:DB8:9:3::/64 2001:DB8:9:2::1 ! Step 3: Wrong neighbor address
ipv6 route 2001:DB8:9:3::/64 G0/1 FE80::2 ! Step 4: Wrong interface on R1
```

All these incorrect examples have correct syntax and would be added to R1's IPv6 routing table if configured on R1. However, all have flaws. Working through the examples in order:

Step 1. The prefix (2001:DB8:9:33::) has a typo in the fourth quartet (33 instead of 3).

Step 2A. The figure shows R2's G0/1 with link-local address FE80::2, but the command uses FE80::AAA9.

Step 2B. The command uses the correct link-local address on R2's address on the common link (FE80::2 per the figure), but it omits the outgoing interface of R1's G0/2 interface. (See the next example for more detail.)

Step 3. The figure shows the subnet in the center as 2001:DB8:9:2::/64, with R1 using the ::1 address and R2 using ::2. For the fourth command, R1's command should use R2's address 2001:DB8:9:2::2, but it uses R1's own 2001:DB8:9:2::1 address instead.

Step 4. As a command on R1, the outgoing interface references R1's own interfaces. R1's G0/1 is the interface on the left, whereas R1 should use its G0/2 interface on the right when forwarding packets to subnet 2001:DB8:9:3::/64.

The key takeaway for this section is to know that a route in the IPv6 routing table may be incorrect due to poor choices for the parameters. The parameters should always include the

neighboring router's IPv6 addresses, but the local router's interface type/number, and in all cases, the correct prefix/length. The fact that a route is in the IPv6 routing table, particularly a static route, does not mean it is a correct route.

Note that of the five example commands in Example J-3, IOS would accept all of them except the third one. IOS can notice the case of omitting the outgoing interface if the next-hop address is a link-local address. Example J-4 shows a sample of the error message from IOS.

Example J-4 *IOS Rejects the* **ipv6 route** *Command with Link-Local and No Outgoing Interface*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ipv6 route 2001:DB8:9:3::/64 FE80::2
% Interface has to be specified for a link-local nexthop
R1(config)# ^Z
R1#
R1# show running-config | include ipv6 route
R1#
```

The Static Route Does Not Appear in the IPv6 Routing Table

The preceding few pages focused on IPv6 static routes that show up in the IPv6 routing table but unfortunately have incorrect parameters. The next page looks at IPv6 routes that have correct parameters, but IOS does not place them into the IPv6 routing table.

When you add an **ipv6 route** command to the configuration, and the syntax is correct, IOS considers that route to be added to the IPv6 routing table. IOS makes the following checks before adding the route; note that IOS uses this same kind of logic for IPv4 static routes:

Key Topic

- For **ipv6 route** commands that list an outgoing interface, that interface must be in an up/up state.
- For **ipv6 route** commands that list a global unicast or unique local next-hop IP address (that is, not a link-local address), the local router must have a route to reach that next-hop address.
- If another IPv6 route exists for that exact same prefix/prefix-length, the static route must have a better (lower) administrative distance.

For example, Router R1, again from Figure J-7, has been configured with IPv6 addresses. Example J-5 shows the addition of an **ipv6 route** command for remote subnet 2001:DB8:9:3::/64, but with incorrect next-hop address 2001:DB8:9:3::2. That address is on R2, but it is the address on the far side of R2, on R2's G0/2 interface.

Example J-5 *No Route for Next-Hop IPv6 Address in Static Route*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ipv6 route 2001:DB8:9:3::/64 2001:DB8:9:3::2
R1(config)# ^Z
R1# show ipv6 route
IPv6 Routing Table - default - 5 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
      IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
      ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
      RL - RPL, O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1
      OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
      la - LISP alt, lr - LISP site-registrations, ld - LISP dyn-eid
       a - Application
C 2001:DB8:9:1::/64 [0/0]
     via GigabitEthernet0/1, directly connected
L 2001:DB8:9:1::1/128 [0/0]
     via GigabitEthernet0/1, receive
C 2001:DB8:9:2::/64 [0/0]
     via GigabitEthernet0/2, directly connected
L 2001:DB8:9:2::1/128 [0/0]
     via GigabitEthernet0/2, receive
L FF00::/8 [0/0]
     via Null0, receive
```

NOTE The content under the heading "Default Routes with SLAAC on Router Interfaces" was most recently published in Chapter 32 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Default Routes with SLAAC on Router Interfaces

Routers can use DHCP on their own interface and learn their IP address, mask, and even a default IPv4 route. In particular, that process can be useful on a router that connects to the Internet. The enterprise router uses DHCP as a client, learning its own IPv4 address with DHCP and adding a default route pointing to the ISP's router as the next-hop IPv4 address.

Routers can accomplish the same goals with IPv6, just with a few different protocols and methods. As with IPv4, the IPv6 enterprise router can dynamically learn its IPv6 address and dynamically create a default IPv6 route to the ISP's router. This section shows the details, with the enterprise router using SLAAC to learn its address and the information needed to create a default route.

First, the enterprise router that connects to the ISP, like Router R1 in Figure J-8, requires the configuration of the interface subcommand **ipv6 address autoconfig default**. This command tells the router that, on that interface, use SLAAC to build its own IPv6 address. R1 would act like any host that uses SLAAC, as shown in Step 2 of the figure, and send an NDP RS message over the link. As noted at Step 3, the ISP router would send back an RA message, announcing router ISP1's IPv6 address and the IPv6 prefix used on the link.

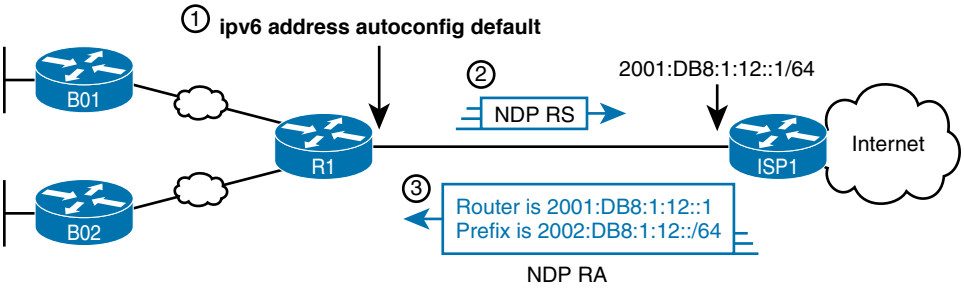


Figure J-8 *Enterprise Router Using SLAAC to Build IPv6 Address and Default IPv6 Route*

When R1 receives the NDP RA message, it does the following:

Interface address: Builds its own interface IPv6 address using the SLAAC process, based on the prefix in the RA.

Local /128 Route: Adds a local (/128) IPv6 route for the address, as it would for any interface IPv6 address.

Connected Route for Prefix: Adds a connected (/64) route for the prefix learned in the NDP RA message.

Default route: R1 adds a default route, to destination ::/0, with the next-hop address of ISP's link-local address, as learned in the RA sent by router ISP1.

Note that the router can be configured to add this default route or not. As shown in the figure, the router builds a default route. Using the **ipv6 address autoconfig** subcommand without the **default** keyword causes the router to build its address with SLAAC but not add a default route.

Example J-6 shows the three IPv6 routes on Router R1 just mentioned in the list. In particular, note the codes for the connected route and the default route; both codes begin with ND, meaning the route was learned with NDP. In particular, as highlighted in the legend part of the output, *ND* refers to an NDP-learned default route, and *NDp* refers to an NDP-learned prefix (as listed in the NDP RA message in Figure J-9 in this case). Note also that these same two routes have an administrative distance of 2, which is the default administrative distance of IPv6 routes learned with NDP.

Example J-6 *Learning an Address and Default Static Route with DHCP*

```
R1# show ipv6 route
IPv6 Routing Table - default - 4 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
ND ::/0 [2/0]
     via FE80::22FF:FE22:2222, Serial0/0/0
NDp 2001:DB8:1:12::/64 [2/0]
     via Serial0/0/0, directly connected
L 2001:DB8:1:12:32F7:DFF:FE29:8560/128 [0/0]
    via Serial0/0/0, receive
! lines omitted for brevity
```

APPENDIX K

Analyzing Ethernet LAN Designs

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 10 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

Ethernet defines what happens on each Ethernet link, but the more interesting and more detailed work happens on the devices connected to those links: the network interface cards (NIC) inside devices and the LAN switches. This chapter takes the Ethernet LAN basics introduced in Chapter 2, "Fundamentals of Ethernet LANs," and dives deeply into many aspects of a modern Ethernet LAN, while focusing on the primary device used to create these LANs: LAN switches.

This chapter breaks down the discussion of Ethernet and LAN switching into two sections. The first major section looks at the logic used by LAN switches when forwarding Ethernet frames, along with the related terminology. The second section considers design and implementation issues, as if you were building a new Ethernet LAN in a building or campus. This second section considers design issues, including using switches for different purposes, when to choose different types of Ethernet links, and how to take advantage of Ethernet autonegotiation.

Foundation Topics

Analyzing Collision Domains and Broadcast Domains

Ethernet devices, and the logic they use, have a big impact on why engineers design modern LANs in a certain way. Some of the terms used to describe key design features come from far back in the history of Ethernet, and because of their age, the meaning of each term may or may not be so obvious to someone learning Ethernet today. This first section of the chapter looks at two of these older terms in particular: collision domain and broadcast domain. And to understand these terms and apply them to modern Ethernet LANs, this section needs to work back through the history of Ethernet a bit, to put some perspective on the meaning behind these terms.

Ethernet Collision Domains

The term *collision domain* comes from the far back history of Ethernet LANs. To be honest, sometimes people new to Ethernet can get a little confused about what this term really means in the context of a modern Ethernet LAN, in part because modern Ethernet LANs, done properly, can completely prevent collisions. So to fully understand collision domains, we must first start with a bit of Ethernet history. This next section of the chapter looks at a few of the historical Ethernet devices, for the purpose of defining a collision domain, and then closing with some comments about how the term applies in a modern Ethernet LAN that uses switches.

10BASE-T with Hub

10BASE-T, introduced in 1990, significantly changed the design of Ethernet LANs, more like the designs seen today. 10BASE-T introduced the cabling model similar to today's Ethernet LANs, with each device connecting to a centralized device using an unshielded twisted-pair (UTP) cable. However, 10BASE-T did not originally use LAN switches; instead, the early 10BASE-T networks used a device called an *Ethernet hub*. (The technology required to build even a basic LAN switch was not yet available at that time.)

Although both a hub and a switch use the same cabling star topology, an Ethernet hub does not forward traffic like a switch. Ethernet hubs use physical layer processing to forward data. A hub does not interpret the incoming electrical signal as an Ethernet frame, look at the source and destination MAC address, and so on. Basically, a hub acts like a repeater, just with lots of ports. When a repeater receives an incoming electrical signal, it immediately *forwards a regenerated signal out all the other ports except the incoming port*. Physically, the hub just sends out a cleaner version of the same incoming electrical signal, as shown in Figure K-1, with Larry's signal being repeated out the two ports on the right.

Key Topic

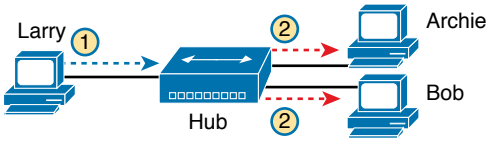


Figure K-1 *10BASE-T (with a Hub): The Hub Repeats Out All Other Ports*

Because of the physical layer operation used by the hub, the devices attached to the network must use carrier sense multiple access with collision detection (CSMA/CD) to take turns (as introduced at the end of Chapter 2). Note that the hub itself does not use CSMA/CD logic; the hub always receives an electrical signal and starts repeating a (regenerated) signal out all other ports, with no thought of CSMA/CD. So, although a hub's logic works well to make sure all devices get a copy of the original frame, that same logic causes frames to collide. Figure K-2 demonstrates that effect, when the two devices on the right side of the figure send a frame at the same time, and the hub physically transmits both electrical signals out the port to the left (toward Larry).

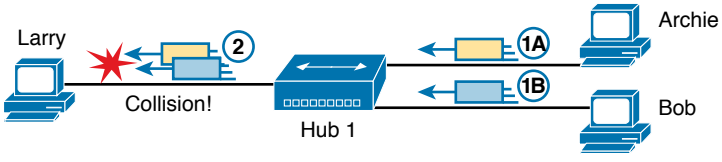


Figure K-2 *Hub Operation Causing a Collision*

Because a hub makes no attempt to prevent collisions, the devices connected to it all sit within the same collision domain. A *collision domain* is the set of NICs and device ports for which if they sent a frame at the same time, the frames would collide. In Figures K-1 and K-2, all three PCs are in the same collision domain, as well as the hub. Summarizing the key points about hubs:

Key Topic

- The hub acts a multiport repeater, blindly regenerating and repeating any incoming electrical signal out all other ports, even ignoring CSMA/CD rules.
- When two or more devices send at the same time, the hub's actions cause an electrical collision, making both signals corrupt.
- The connected devices must take turns by using carrier sense multiple access with collision detection (CSMA/CD) logic, so the devices share the bandwidth.
- Hubs create a physical star topology.

Ethernet Transparent Bridges

From a design perspective, the introduction of 10BASE-T was a great improvement over the earlier types of Ethernet. It reduced cabling costs and cable installation costs, and improved the availability percentages of the network. But sitting here today, thinking of a LAN in which all devices basically have to wait their turn may seem like a performance issue, and it was. If Ethernet could be improved to allow multiple devices to send at the same time without causing a collision, Ethernet performance could be improved.

The first method to allow multiple devices to send at the same time was Ethernet transparent bridges. Ethernet *transparent bridges*, or simply *bridges*, made these improvements:

- Bridges sat between hubs and divided the network into multiple *collision domains*.
- Bridges increase the capacity of the entire Ethernet, because each collision domain is basically a separate instance of CSMA/CD, so each collision domain can have one sender at a time.

Figure K-3 shows the effect of building a LAN with two hubs, each separated by a bridge. The resulting two collision domains each support at most 10 Mbps of traffic each, compared to at most 10 Mbps if a single hub were used.

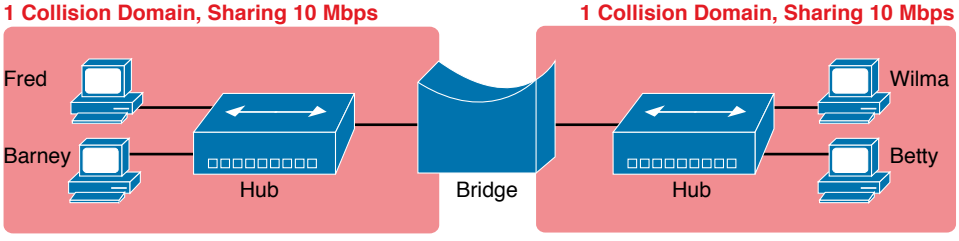


Figure K-3 *Bridge Creates Two Collision Domains and Two Shared Ethernets*

Bridges create multiple collision domains as a side effect of their forwarding logic. A bridge makes forwarding decisions just like a modern LAN switch; in fact, bridges were the predecessors of the modern LAN switch. Like switches, bridges hold Ethernet frames in memory, waiting to send out the outgoing interface based on CSMA/CD rules. In other cases, the bridge does not even need to forward the frame. For instance, if Fred sends a frame destined to Barney's MAC address, then the bridge would never forward frames from the left to the right.

Ethernet Switches and Collision Domains

LAN switches perform the same basic core functions as bridges but at much faster speeds and with many enhanced features. Like bridges, switches segment a LAN into separate collision domains, each with its own capacity. And if the network does not have a hub, each single link in a modern LAN is considered its own collision domain, even if no collisions can actually occur in that case.

For example, Figure K-4 shows a simple LAN with a switch and four PCs. The switch creates four collision domains, with the ability to send at 100 Mbps in this case on each of the four links. And with no hubs, each link can run at full duplex, doubling the capacity of each link.

Key Topic

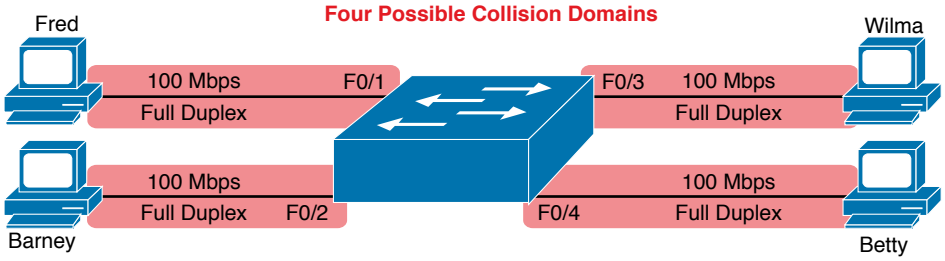


Figure K-4 *Switch Creates Four Collision Domains and Four Ethernet Segments*

Now take a step back for a moment and think about some facts about modern Ethernet LANs. Today, you build Ethernet LANs with Ethernet switches, not with Ethernet hubs or bridges. The switches connect to each other. And every single link is a separate collision domain.

As strange as it sounds, each of those collision domains in a modern LAN may also never have a collision. Any link that uses full duplex—that is, both devices on the link use full duplex—does not have collisions. In fact, running with full duplex is basically this idea: No collisions can occur between a switch and a single device, so we can turn off CSMA/CD by running full duplex.

NOTE The routers in a network design also create separate collision domains, because frames entering or exiting one router LAN interface do not collide with frames on another of the router's LAN interfaces.

The Impact of Collisions on LAN Design

So, what is the useful takeaway from this discussion about collision domains? A long time ago, collisions were normal in Ethernet, so analyzing an Ethernet design to determine where the collision domains were was useful. On the other end of the spectrum, a modern campus LAN that uses only switches (and no hubs or transparent bridges), and full duplex on all links, has no collisions at all. So does the collision domain term still matter today? And do we need to think about collisions even still today?

In a word, the term collision domain still matters, and collisions still matter, in that network engineers need to be ready to understand and troubleshoot exceptions. Whenever a port that could use full duplex (therefore avoiding collisions) happens to use half duplex—by incorrect configuration, by the result of autonegotiation, or any other reason—collisions can now occur. In those cases, engineers need to be able identify the collision domain.

Summarizing the key points about collision domains:

Key Topic

- LAN switches place each separate interface into a separate collision domain.
- LAN bridges, which use the same logic as switches, placed each interface into a separate collision domain.
- Routers place each LAN interface into a separate collision domain. (The term collision domain does not apply to WAN interfaces.)
- LAN hubs do not place each interface into a separate collision domain.

- A modern LAN, with all LAN switches and routers, with full duplex on each link, would not have collisions at all.
- In a modern LAN with all switches and routers, even though full duplex removes collisions, think of each Ethernet link as a separate collision domain when the need to troubleshoot arises.

Figure K-5 shows an example with a design that includes hubs, bridges, switches, and routers—a design that you would not use today, but it makes a good backdrop to remind us about which devices create separate collision domains.

Key Topic

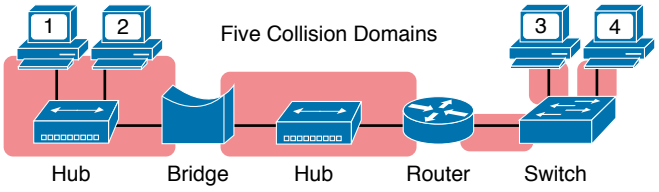
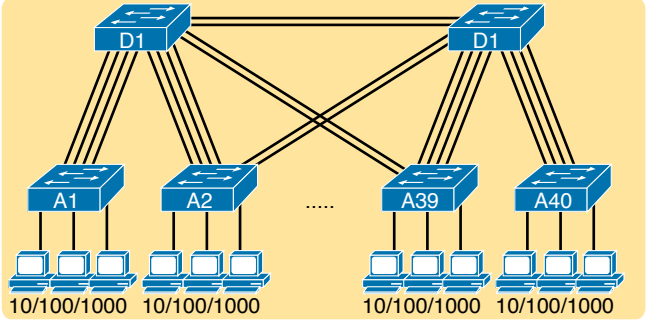


Figure K-5 *Example of a Hub Not Creating Multiple Collision Domains, While Others Do*

Ethernet Broadcast Domains

Take any Ethernet LAN, and pick any device. Then think of that device sending an Ethernet broadcast. An Ethernet *broadcast domain* is the set of devices to which that broadcast is delivered.

To begin, think about a modern LAN for a moment, and where a broadcast frame flows. Imagine that all the switches still used the switch default to put each interface into VLAN 1. As a result, a broadcast sent by any one device would be flooded to all devices connected to all switches (except for the device that sent the original frame). For instance, in Figure K-6, under the assumption that all ports are still assigned to VLAN 1, a broadcast would flow to all the devices shown in the figure.



One VLAN!

Figure K-6 *A Single Large Broadcast Domain*

Of all the common networking devices discussed in this book, only a router does not forward a LAN broadcast. Hubs of course forward broadcasts, because hubs do not even think about the electrical signal as an Ethernet frame. Bridges and switches use the same forwarding logic, flooding LAN broadcasts. Routers, as a side effect of their routing logic, do not forward Ethernet broadcast frames, so they separate a network into separate broadcast domains. Figure K-7 collects those thoughts into a single example.

Key Topic

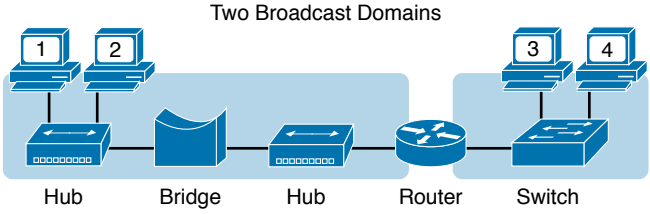


Figure K-7 *Broadcast Domains Separated by a Router*

By definition, broadcasts sent by a device in one broadcast domain are not forwarded to devices in another broadcast domain. In this example, there are two broadcast domains. The router does not forward a LAN broadcast sent by a PC on the left to the network segment on the right.

Virtual LANs

Routers create multiple broadcast domains mostly as a side effect of how IP routing works. While a network designer might set about to use more router interfaces for the purpose of making a larger number of smaller broadcast domains, that plan quickly consumes router interfaces. But a better tool exists, one that is integrated into LAN switches and consumes no additional ports: virtual LANs (VLAN).

By far, VLANs give the network designer the best tool for designing the right number of broadcast domains, of the right size, with the right devices in each. To appreciate how VLANs do that, you must first think about one specific definition of what a LAN is:

> A LAN consists of all devices in the same broadcast domain.

With VLANs, a switch configuration places each port into a specific VLAN. The switches create multiple broadcast domains by putting some interfaces into one VLAN and other interfaces into other VLANs. The switch forwarding logic does not forward frames from a port in one VLAN out a port into another VLAN—so the switch separates the LAN into separate broadcast domains. Instead, routers must forward packets between the VLANs by using routing logic. So, instead of all ports on a switch forming a single broadcast domain, the switch separates them into many, based on configuration.

For perspective, think about how you would create two different broadcast domains with switches if the switches had no concept of VLANs. Without any knowledge of VLANs, a switch would receive a frame on one port and flood it out all the rest of its ports. Therefore, to make two broadcast domains, two switches would be used—one for each broadcast domain, as shown in Figure K-8.

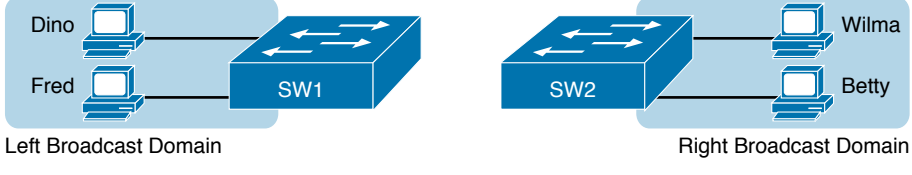


Figure K-8 *Sample Network with Two Broadcast Domains and No VLANs*

Alternatively, with a switch that understands VLANs, you can create multiple broadcast domains using a single switch. All you do is put some ports in one VLAN and some in the other. (The Cisco Catalyst switch interface subcommand to do so is **switchport access vlan 2**, for instance, to place a port into VLAN 2.) Figure K-9 shows the same two broadcast domains as in Figure K-8, now implemented as two different VLANs on a single switch.

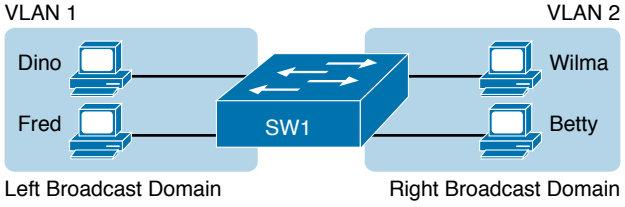


Figure K-9 *Sample Network with Two VLANs Using One Switch*

This section briefly introduces the concept of VLANs, but Chapter 11, "Implementing Ethernet Virtual LANs," discusses VLANs in more depth, including the details of how to configure VLANs in campus LANs.

The Impact of Broadcast Domains on LAN Design

Modern LAN designs try to avoid collisions, because collisions make performance worse. There is no benefit to keeping collisions in the network. However, a LAN design cannot remove broadcasts, because broadcast frames play an important role in many protocols. So when thinking about broadcast domains, the choices are more about tradeoffs rather than designing to remove broadcasts.

For just one perspective, just think about the size of a broadcast domain—that is, the number of devices in the same broadcast domain. A small number of large broadcast domains can lead to poor performance for the devices in that broadcast domain. However, moving in the opposite direction, to making a large number of broadcast domains each with just a few devices, leads to other problems.

Consider the idea of a too-large broadcast domain for a moment. When a host receives a broadcast, the host must process the received frame. All hosts need to send some broadcasts to function properly, so when a broadcast arrives, the NIC must interrupt the computer's CPU to give the incoming message to the CPU. The CPU must spend time thinking about the received broadcast frame. (For example, IP Address Resolution Protocol [ARP] messages are LAN broadcasts, as mentioned in Chapter 4, "Fundamentals of IPv4 Addressing and Routing.") So, broadcasts happen, which is good, but broadcasts do require all the hosts to spend time processing each broadcast frame. The more devices in the same broadcast domain, the more unnecessary interruptions of each device's CPU.

This section of the book does not try to give a sweeping review of all VLAN design tradeoffs. Instead, you can see that the size of a VLAN should be considered, but many other factors come in to play as well. How big are the VLANs? How are the devices grouped? Do VLANs span across all switches or just a few? Is there any apparent consistency to the VLAN design, or is it somewhat haphazard? Answering these questions helps reveal what the designer was thinking, as well as what the realities of operating a network may have required.

NOTE If you would like more detail about Cisco recommendations about what to put in what VLAN, which impacts the size of VLANs, read the most recent Cisco document, "Campus LAN validated design" by searching on that phrase at Cisco.com.

Summarizing the main points about broadcast domains:

Key Topic

- Broadcasts exists, so be ready to analyze a design to define each broadcast domain, that is, each set of devices whose broadcasts reach the other devices in that domain.
- VLANs by definition are broadcast domains created though configuration.
- Routers, because they do not forward LAN broadcasts, create separate broadcast domains off their separate Ethernet interfaces.

Analyzing Campus LAN Topologies

The term *campus LAN* refers to the LAN created to support the devices in a building or in multiple buildings in somewhat close proximity to one another. For example, a company might lease office space in several buildings in the same office park. The network engineers can then build a campus LAN that includes switches in each building, plus Ethernet links between the switches in the buildings, to create a larger campus LAN.

When planning and designing a campus LAN, the engineers must consider the types of Ethernet available and the cabling lengths supported by each type. The engineers also need to choose the speeds required for each Ethernet segment. In addition, some thought needs to be given to the idea that some switches should be used to connect directly to end-user devices, whereas other switches might need to simply connect to a large number of these end-user switches. Finally, most projects require that the engineer consider the type of equipment that is already installed and whether an increase in speed on some segments is worth the cost of buying new equipment.

This second of three major sections of the chapter discusses the topology of a campus LAN design. Network designers do not just plug in devices to any port and connect switches to each other in an arbitrary way, like you might do with a few devices on the same table in a lab. Instead, there are known better ways to design the topology of a campus LAN, and this section introduces some of the key points and terms. The last major section of the chapter then looks at how to choose which Ethernet standard to use for each link in that campus LAN design, and why you might choose one versus another.

Two-Tier Campus Design (Collapsed Core)

To sift through all the requirements for a campus LAN, and then have a reasonable conversation about it with peers, most Cisco-oriented LAN designs use some common terminology to refer to the design. For this book's purposes, you should be aware of some of the key campus LAN design terminology.

The Two-Tier Campus Design

Figure K-10 shows a typical design of a large campus LAN, with the terminology included in the figure. This LAN has around 1000 PCs connected to switches that support around 25 ports each. Explanations of the terminology follow the figure.

Key Topic

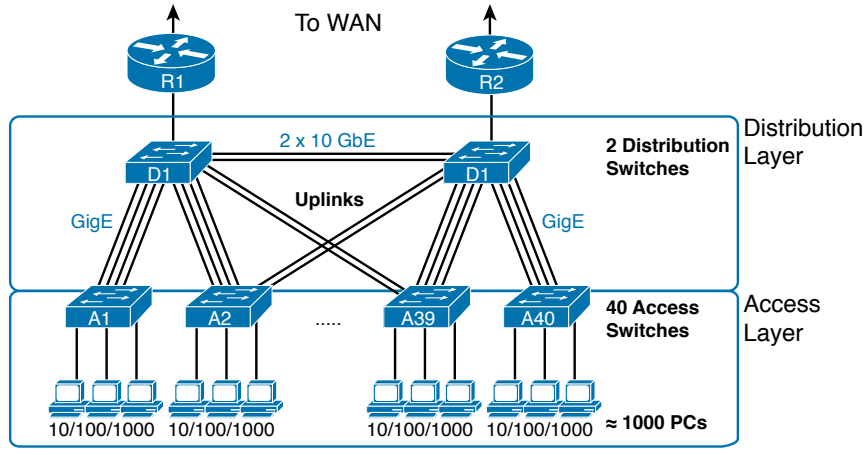


Figure K-10 *Campus LAN with Design Terminology Listed*

Cisco uses three terms to describe the role of each switch in a campus design: *access*, *distribution*, and *core*. The roles differ based on whether the switch forwards traffic from user devices and the rest of the LAN (access), or whether the switch forwards traffic between other LAN switches (distribution and core).

Access switches connect directly to end users, providing user device access to the LAN. Access switches normally send traffic to and from the end-user devices to which they are connected and sit at the edge of the LAN.

Distribution switches provide a path through which the access switches can forward traffic to each other. By design, each of the access switches connects to at least one distribution switch, typically to two distribution switches for redundancy. The distribution switches provide the service of forwarding traffic to other parts of the LAN. Note that most designs use at least two uplinks to two different distribution switches (as shown in Figure K-10) for redundancy.

The figure shows a two-tier design, with the tiers being the access tier (or layer) and the distribution tier (or layer). A two-tier design solves two major design needs:

- Provides a place to connect end-user devices (the access layer, with access switches)
- Connects the switches with a reasonable number of cables and switch ports by connecting all 40 access switches to two distribution switches

Topology Terminology Seen Within a Two-Tier Design

The exam topics happen to list a couple of terms about LAN and WAN topology and design, so this is a good place to pause to discuss those terms for a moment.

First, consider these more formal definitions of four topology terms:

Key Topic

Star: A design in which one central device connects to several others, so that if you drew the links out in all directions, the design would look like a star with light shining in all directions.

Full mesh: For any set of network nodes, a design that connects a link between each pair of nodes.

Partial mesh: For any set of network nodes, a design that connects a link between some pairs of nodes, but not all. In other words, a mesh that is not a full mesh.

Hybrid: A design that combines topology design concepts into a larger (typically more complex) design.

Armed with those formal definitions, note that the two-tier design is indeed a hybrid design that uses both a star topology at the access layer and a partial mesh at the distribution layer. To see why, consider Figure K-11. It redraws a typical access layer switch, but instead of putting the PCs all below the switch, it spreads them around the switch. Then on the right, a similar version of the same drawing shows why the term star might be used—the topology looks a little like a child's drawing of a star.

Key Topic

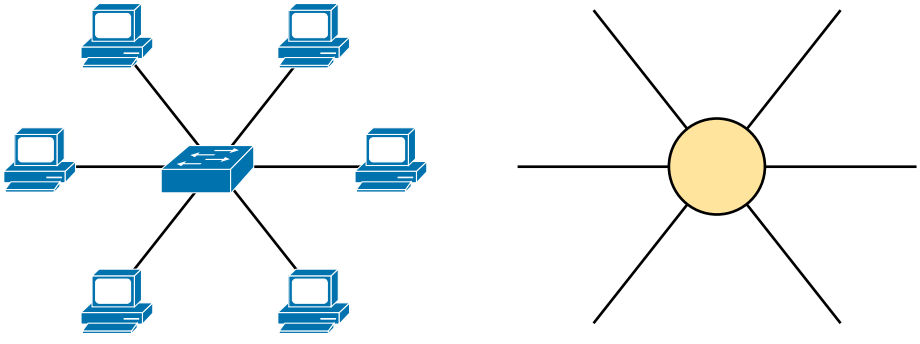


Figure K-11 *The Star Topology Design Concept in Networking*

The distribution layer creates a partial mesh. If you view the access and distribution switches as nodes in a design, some nodes have a link between them, and some do not. Just refer to Figure K-10 and note that, by design, none of the access layer switches connect to each other.

Finally, a design could use a full mesh. However, for a variety of reasons beyond the scope of the design discussion here, a campus design typically does not need to use the number of links and ports required by a full mesh design. However, just to make the point, first consider how many links and switch ports would be required for a single link between nodes in a full mesh, with six nodes, as shown in Figure K-12.

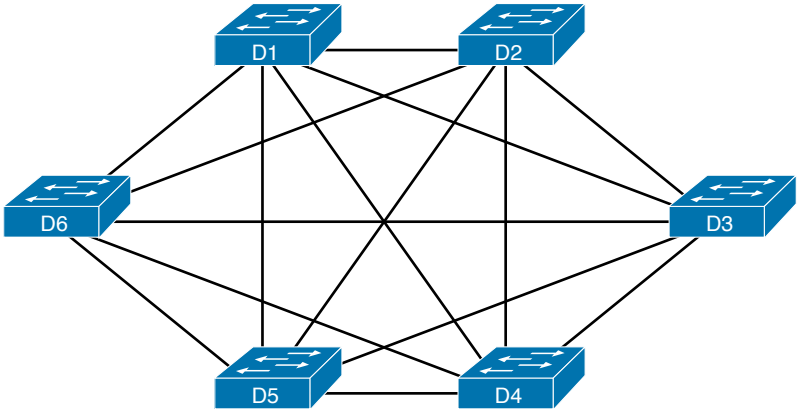


Figure K-12 *Using a Full Mesh at the Distribution Layer, 6 Switches, 15 Links*

Even with only six switches, a full mesh would consume 15 links (and 30 switch ports—two per link).

Now think about a full mesh at the distribution layer for a design like Figure K-10, with 40 access switches and two distribution switches. Rather than drawing it and counting it, the number of links is calculated with this old math formula from high school: $N(N - 1) / 2$, or in this case, 42 * 41 / 2 = 861 links, and 1722 switch ports consumed among all switches.

For comparison's sake, the partial mesh design of Figure K-10, with a pair of links from each access switch to each distribution switch, requires only 160 links and a total of 320 ports among all switches.

Three-Tier Campus Design (Core)

The two-tier design of Figure K-10, with a partial mesh of links at the distribution layer, happens to be the most common campus LAN design. It also goes by two common names: a two-tier design (for obvious reasons), and a collapsed core (for less obvious reasons). The term *collapsed core* refers to the fact that the two-tier design does not have a third tier, the core tier. This next topic examines a three-tier design that does have a core, for perspective.

Imagine your campus has just two or three buildings. Each building has a two-tier design inside the building, with a pair of distribution switches in each building and access switches spread around the building as needed. How would you connect the LANs in each building? Well, with just a few buildings, it makes sense to simply cable the distribution switches together, as shown in Figure K-13.

Key Topic

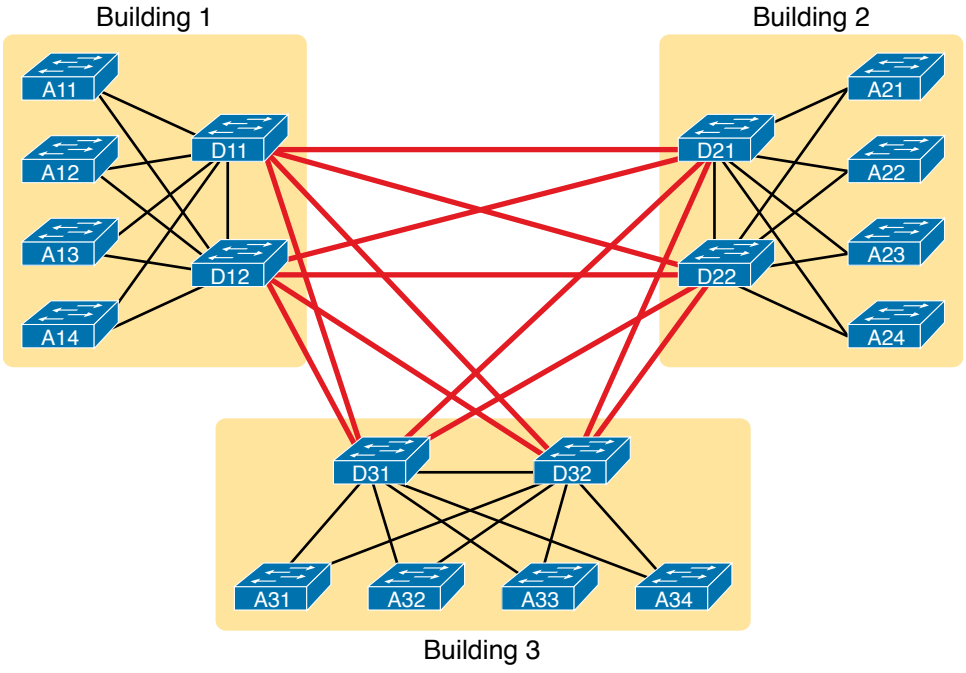


Figure K-13 *Two-Tier Building Design, No Core, Three Buildings*

The design in Figure K-13 works well, and many companies use this design. Sometimes the center of the network uses a full mesh, sometimes a partial mesh, depending on the availability of cables between the buildings.

However, a design with a third tier (a core tier) saves on switch ports and on cables in larger designs. And note that with the links between buildings, the cables run outside, are often more expensive to install, are almost always fiber cabling with more expensive switch ports, so conserving the number of cables used between buildings can help reduce costs.

A three-tier core design, unsurprisingly at this point, adds a few more switches (core switches), which provide one function: to connect the distribution switches. Figure K-14 shows the migration of the Figure K-13 collapsed core (that is, a design without a core) to a three-tier core design.

Key Topic

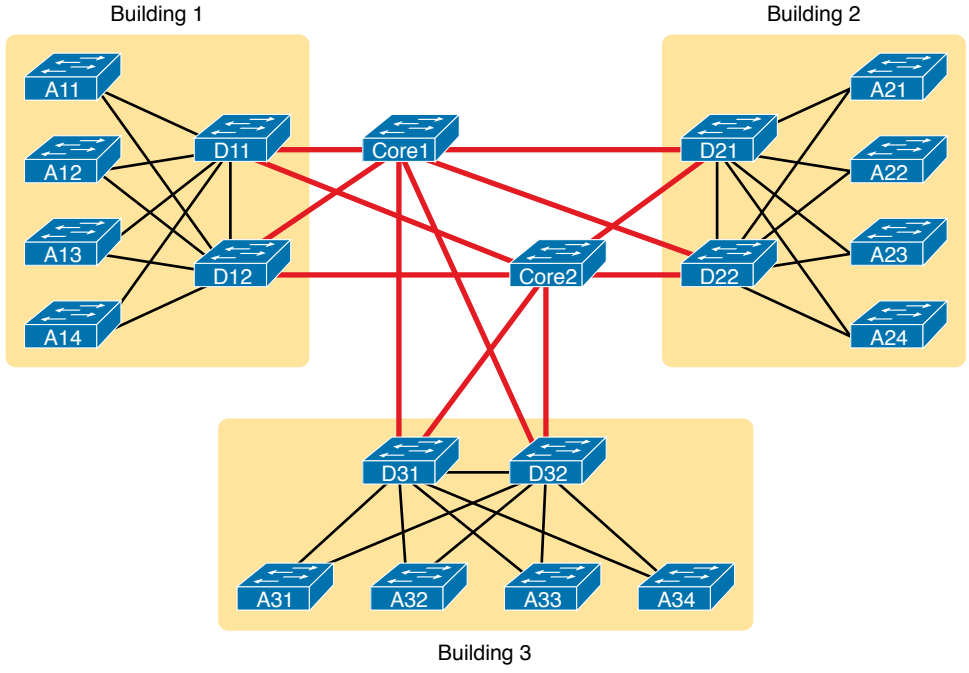


Figure K-14 *Three-Tier Building Design (Core Design), Three Buildings*

NOTE The core switches sit in the middle of the figure. In the physical world, they often sit in the same room as one of the distribution switches, rather than in some purpose-built room in the middle of the office park. The figure focuses more on the topology rather than the physical location.

By using a core design, with a partial mesh of links in the core, you still provide connectivity to all parts of the LAN, and to the routers that send packets over the WAN, just with fewer links between buildings.

The following list summarizes the terms that describe the roles of campus switches:

- **Access:** Provides a connection point (access) for end-user devices. Does not forward frames between two other access switches under normal circumstances.
- **Distribution:** Provides an aggregation point for access switches, providing connectivity to the rest of the devices in the LAN, forwarding frames between switches, but not connecting directly to end-user devices.
- **Core:** Aggregates distribution switches in very large campus LANs, providing very high forwarding rates for the larger volume of traffic due to the size of the network.

Topology Design Terminology

The ICND1 and CCNA exam topics specifically mention several network design terms related to topology. This next topic summarizes those key terms to connect the terms to the matching ideas.

First, consider Figure K-15, which shows a few of the terms. First, on the left, drawings often show access switches with a series of cables, parallel to each other. However, an access switch and its access links is often called a *star topology*. Why? Look at the redrawn access switch in the center of the figure, with the cables radiating out from the center. It does not look like a real star, but it looks a little like a child's drawing of a star, hence the term star topology.

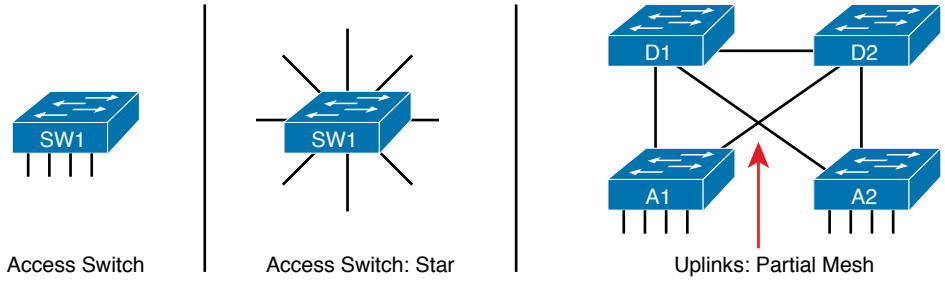


Figure K-15 *LAN Design Terminology*

The right side of the figure repeats a typical two-tier design, focusing on the mesh of links between the access and distribution switches. Any group of nodes that connect with more links than a star topology is typically called a *mesh*. In this case, the mesh is a *partial mesh*, because not all nodes have a direct link between each other. A design that connects all nodes with a link would be a *full mesh*.

Real networks make use of these topology ideas, but often a network combines the ideas together. For instance, the right side of Figure K-14 combines the star topology of the access layer with the partial mesh of the distribution layer. So you might hear these designs that combine concepts called a *hybrid design*.

Analyzing LAN Physical Standard Choices

When you look at the design of a network designed by someone else, you can look at all the different types of cabling used, the different types of switch ports, and the Ethernet standards used in each case. Then ask yourself: Why did they choose a particular type of Ethernet link for each link in the network? Asking that question, and investigating the answer, starts to reveal much about building the physical campus LAN.

The IEEE has done an amazing job developing Ethernet standards that give network designers many options. Two themes in particular have helped Ethernet grow over the long term:

Key Topic

- The IEEE has developed many additional 802.3 standards for different types of cabling, different cable lengths, and for faster speeds.
- All the physical standards rely on the same consistent data-link details, with the same standard frame formats. That means that one Ethernet LAN can use many types of physical links to meet distance, budget, and cabling needs.

For example, think about the access layer of the generic design drawings, but now think about cabling and Ethernet standards. In practice, access layer switches sit in a locked wiring closet somewhere on the same floor as the end user devices. Electricians have installed unshielded twisted-pair (UTP) cabling used at the access layer, running from that wiring closet to each wall plate at each office, cubicle, or any place where an Ethernet device might need to connect to the LAN. The type and quality of the cabling installed

between the wiring closet and each Ethernet outlet dictate what Ethernet standards can be supported. Certainly, whoever designed the LAN at the time the cabling was installed thought about what type of cabling was needed to support the types of Ethernet physical standards that were going to be used in that LAN.

Ethernet Standards

Over time, the IEEE has continued to develop and release new Ethernet standards, for new faster speeds and to support new and different cabling types and cable lengths. Figure K-16 shows some insight into Ethernet speed improvements over the years. The early standards up through the early 1990s ran at 10 Mbps, with steadily improving cabling and topologies. Then, with the introduction of Fast Ethernet (100 Mbps) in 1995, the IEEE began ramping up the speeds steadily over the next few decades, continuing even until today.

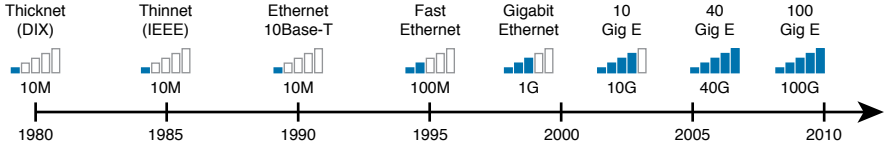


Figure K-16 *Ethernet Standards Timeline*

NOTE Often, the IEEE first introduces support for the next higher speed using some forms of fiber optic cabling, and later, sometimes many years later, the IEEE completes the work to develop standards to support the same speed on UTP cabling. Figure K-16 shows the earliest standards for each speed, no matter what cabling.

When the IEEE introduces support for a new type of cabling, or a faster speed, they create a new standard as part of 802.3. These new standards have a few letters behind the name. So, when speaking of the standards, sometimes you might refer to the standard name (with letters). For instance, the IEEE standardized Gigabit Ethernet support using inexpensive UTP cabling in standard 802.3ab. However, more often, engineers refer to that same standard as 1000BASE-T or simply Gigabit Ethernet. Table K-1 lists some of the IEEE 802.3 physical layer standards and related names for perspective.

Table K-1 IEEE Physical Layer Standards

Original IEEE Standard	Shorthand Name	Informal Names	Speed	Typical Cabling
802.3i	10BASE-T	Ethernet	10 Mbps	UTP
802.3u	100BASE-T	Fast Ethernet	100 Mbps	UTP
802.3z	1000BASE-X	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	Fiber
802.3ab	1000BASE-T	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	UTP
802.3ae	10GBASE-X	10 GigE	10 Gbps	Fiber
802.3an	10GBASE-T	10 GigE	10 Gbps	UTP
802.3ba	40GBASE-X	40 GigE	40 Gbps	Fiber
802.3ba	100GBASE-X	100 GigE	100 Gbps	Fiber

Choosing the Right Ethernet Standard for Each Link

When designing an Ethernet LAN, you can and should think about the topology, with an access layer, a distribution layer, and possibly a core layer. But thinking about the topology does not tell you which specific standards to follow for each link. Ultimately, you need to pick which Ethernet standard to use for each link, based on the following kinds of facts about each physical standard:

- The speed
- The maximum distance allowed between devices when using that standard/cabling
- The cost of the cabling and switch hardware
- The availability of that type of cabling already installed at your facilities

Consider the three most common types of Ethernet today (10BASE-T, 100BASE-T, and 1000BASE-T). They all have the same 100-meter UTP cable length restriction. They all use UTP cabling. However, not all UTP cabling meets the same quality standard, and as it turns out, the faster the Ethernet standard, the higher the required cable quality category needed to support that standard. As a result, some buildings might have better cabling that supports speeds up through Gigabit Ethernet, whereas some buildings may support only Fast Ethernet.

The Telecommunications Industry Association (TIA; tiaonline.org) defines Ethernet cabling quality standards. Each Ethernet UTP standard lists a TIA cabling quality (called a *category*) as the minimum category that the standard supports. For example, 10BASE-T allows for Category 3 (CAT3) cabling or better. 100BASE-T requires higher-quality CAT5 cabling, and 1000BASE-T requires even higher-quality CAT5e cabling. (The TIA standards follow a general "higher number is better cabling" in their numbering.) For instance, if an older facility had only CAT5 cabling installed between the wiring closets and each cubicle, the engineers would have to consider upgrading the cabling to fully support Gigabit Ethernet. Table K-2 lists the more common types of Ethernet and their cable types and length limitations.

Table K-2 Ethernet Types, Media, and Segment Lengths (Per IEEE)

Ethernet Type	Media	Maximum Segment Length
10BASE-T	TIA CAT3 or better, 2 pairs	100 m (328 feet)
100BASE-T	TIA CAT5 UTP or better, 2 pairs	100 m (328 feet)
1000BASE-T	TIA CAT5e UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T	TIA CAT6a UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T[1]	TIA CAT6 UTP or better, 4 pairs	38–55 m (127–180 feet)
1000BASE-SX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	9-micron single-mode fiber	5 km (3.1 miles)

[1] The option for 10GBASE-T with slightly less quality CAT6 cabling, but at shorter distances, is an attempt to support 10Gig Ethernet for some installations with CAT6 installed cabling.

Ethernet defines standards for using fiber optic cables as well. Fiber optic cables include ultrathin strands of glass through which light can pass. To send bits, the switches can alternate between sending brighter and dimmer light to encode 0s and 1s on the cable.

Generally comparing optical cabling versus UTP cabling Ethernet standards, two obvious points stand out. Optical standards allow much longer cabling, while generally costing more for the cable and the switch hardware components. Optical cables experience much less interference from outside sources compared to copper cables, which allows for longer distances.

When considering optical Ethernet links, many standards exist, but with two general categories. Comparing the two, the cheaper options generally support distances into the hundreds of meters, using less expensive light-emitting diodes (LED) to transmit data. Other optical standards support much longer distances into multiple kilometers, using more expensive cabling and using lasers to transmit the data. The trade-off is basic: For a given link, how long does the cable need to run, what standards support that distance, and which is the least expensive to meet that need?

In reality, most engineers remember only the general facts from tables like Table K-2: 100 meters for UTP, about 500 meters for multimode fiber, and about 5000 meters for some single mode fiber Ethernet standards. When it is time to get serious about designing the details of each link, the engineer must get into the details, calculating the length of each cable based on its path through the building, and so on.

Wireless LANs Combined with Wired Ethernet

Modern campus LANs include a large variety of wireless devices that connect to the access layer of the LAN. As it turns out, Cisco organizes wireless LANs into a separate certification track—CCNA, CCNP, and CCIE Wireless—so the CCNA R&S track has traditionally had only a little wireless LAN coverage. The current version of the exams are no different, with this one exam CCNA R&S topic mentioning wireless LANs:

> Describe the impact of infrastructure components in an enterprise network: Access points and wireless controllers

Do not let that small mention of wireless technology make you think that wireless is less important than Ethernet. In fact, there may be more wireless devices than wired at the access layer of today's enterprise networks. Both are important; Cisco just happens to keep the educational material for wireless in a separate certification track.

This last topic in the chapter examines that one exam topic that mentions two wireless terms.

Home Office Wireless LANs

First, the IEEE defines both Ethernet LANs and Wireless LANs. In case it was not obvious yet, all Ethernet standards use cables—that is, Ethernet defines wired LANs. The IEEE 802.11 working group defines Wireless LANs, also called Wi-Fi per a trademarked term from the Wi-Fi Alliance (wi-fi.org), a consortium that helps to encourage wireless LAN development in the marketplace.

Most of you have used Wi-Fi, and may use it daily. Some of you may have set it up at home, with a basic setup as shown in Figure K-17. In a home, you probably used a single consumer device called a *wireless router*. One side of the device connects to the Internet, while the other side connects to the devices in the home. In the home, the devices can connect either with Wi-Fi or with a wired Ethernet cable.

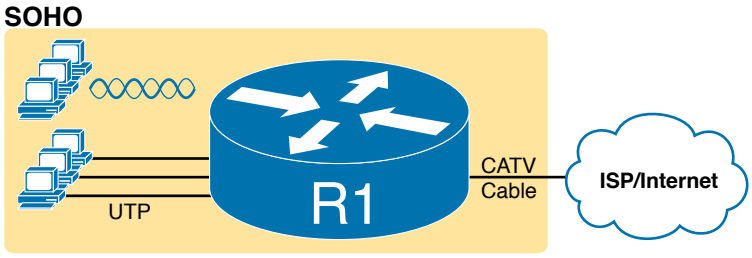


Figure K-17 *A Typical Home Wired and Wireless LAN*

While the figure shows the hardware as a single router icon, internally, that one wireless router acts like three separate devices you would find in an enterprise campus:

- An Ethernet switch, for the wired Ethernet connections
- A wireless access point (AP), to communicate with the wireless devices and forward the frames to/from the wired network
- A router, to route IP packets to/from the LAN and WAN (Internet) interfaces

Figure K-18 repeats the previous figure, breaking out the internal components as if they were separate physical devices, just to make the point that a single consumer wireless router acts like several different devices.

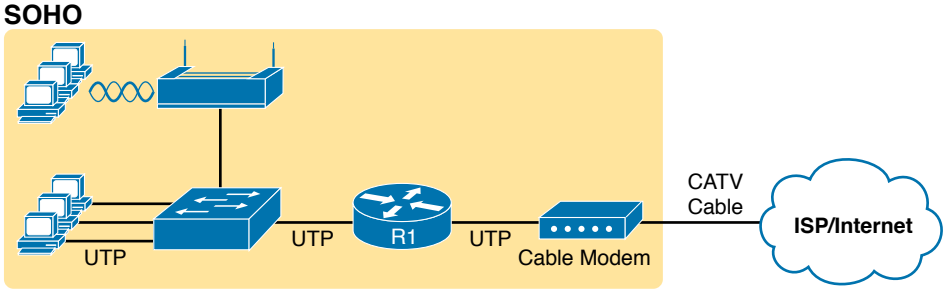


Figure K-18 *A Representation of the Functions Inside a Consumer Wireless Routing Product*

In a small office/home office (SOHO) wireless LAN, the wireless AP acts autonomously, doing all the work required to create and control the wireless LAN (WLAN). (In most enterprise WLANs, the AP does not act autonomously.) In other words, the autonomous AP communicates with the various wireless devices using 802.11 protocols and radio waves. It uses Ethernet protocols on the wired side. It converts between the differences in header formats between 802.11 and 802.3 frames before forwarding to/from 802.3 Ethernet and 802.11 wireless frames.

Beyond those basic forwarding actions, the autonomous AP must perform a variety of control and management functions. The AP authenticates new devices, defines the name of the WLAN (called a service set ID, or SSID), and other details.

Enterprise Wireless LANs and Wireless LAN Controllers

If you connect to your WLAN at home from your tablet, phone, or laptop, and then walk down the street with that same device, you expect to lose your Wi-Fi connection at some point. You do not expect to somehow automatically connect to a neighbor's Wi-Fi network, particularly if they did the right thing and set up security functions on their AP to prevent others from accessing their home Wi-Fi network. The neighborhood does not create one WLAN supported by the devices in all the houses and apartments; instead, it has lots of little autonomous WLANs.

However, in an enterprise, the opposite needs to happen. We want people to be able to roam around the building and office campus and keep connected to the Wi-Fi network. This requires many APs, which work together rather than autonomously to create one wireless LAN.

First, think about the number of APs an enterprise might need. Each AP can cover only a certain amount of space, depending on a large number of conditions and the wireless standard. (The size varies, but the distances sit in the 100 to 200 feet range.) At the same time, you might have the opposite problem; you may just need lots of APs in a small space, just to add capacity to the WLAN. Much of the time spent designing WLANs revolves around deciding how many APs to place in each space, and of what types, to handle the traffic.

NOTE If you have not paid attention before, start looking around the ceilings of any new buildings you enter, even retail stores, and look for their wireless APs.

Each AP must then connect to the wired LAN, because most of the destinations that wireless users need to communicate with sit in the wired part of the network. In fact, the APs typically sit close to where users sit, for obvious reasons, so the APs connect to the same access switches as the end users, as shown in Figure K-19.

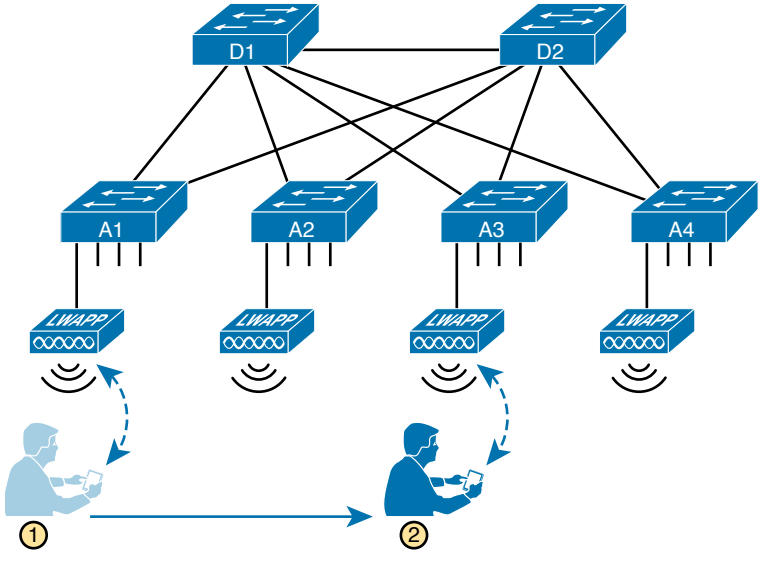


Figure K-19 *Campus LAN, Multiple Lightweight APs, with Roaming*

Now imagine that is you at the bottom of the figure. Your smartphone has Wi-Fi enabled, so that when you walk into work, your phone automatically connects to the company WLAN. You roam around all day, going to meetings, lunch, and so on. All day long you stay connected to the company WLAN, but your phone connects to and uses many different APs.

Supporting roaming and other enterprise WLAN features by using autonomous APs can be difficult at best. You could imagine that if you had a dozen APs per floor, you might have hundreds of APs in a campus—all of which need to know about that one WLAN.

The solution: remove all the control and management features from the APs, and put them in one centralized place, called a Wireless Controller, or Wireless LAN Controller (WLC). The APs no longer act autonomously, but instead act as lightweight APs (LWAPs), just forwarding data between the wireless LAN and the WLC. All the logic to deal with roaming, defining WLANs (SSIDs), authentication, and so on happens in the centralized WLC rather than on each AP. Summarizing:

Wireless LAN controller: Controls and manages all AP functions (for example, roaming, defining WLANs, authentication)

Lightweight AP (LWAP): Forwards data between the wired and wireless LAN, and specifically forwarding data through the WLC using a protocol like Control And Provisioning of Wireless Access Points (CAPWAP)

With the WLC and LWAP design, the combined LWAPs and WLC can create one big wireless network, rather than creating a multitude of disjointed wireless networks. The key to making it all work is that all wireless traffic flows through the WLC, as shown in Figure K-20. (The LWAPs commonly use a protocol called CAPWAP, by the way.)

Key Topic

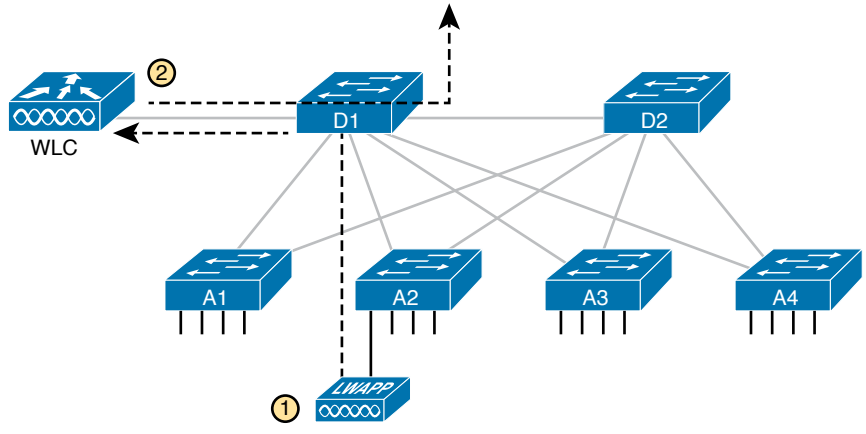


Figure K-20 *Campus LAN, Multiple Lightweight APs, with Roaming*

By forwarding all the traffic through the WLC, the WLC can make the right decisions across the enterprise. For example, you might create a marketing WLAN, an engineering WLAN, and so on, and all the APs know about and support those multiple different WLANs. Users that connect to the engineering WLAN should use the same authentication rules regardless of which AP they use—and the WLC makes that possible. Or consider

roaming for a moment. If at one instant a packet arrives for your phone, and you are associated with AP1, and when the next packet arrives over the wired network you are now connected to AP4, how could that packet be delivered through the network? Well, it always goes to the WLC, and because the WLC keeps in contact with the APs and knows that your phone just roamed to another AP, the WLC knows where to forward the packet.

APPENDIX L

Subnet Design

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 21 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

So far in this book, most of the discussion about IPv4 used examples with the addresses and masks already given. This book has shown many examples already, but the examples so far do not ask you to pick the IP address or pick the mask. Instead, as discussed back in Chapter 11, "Perspectives on IPv4 Subnetting," this book so far has assumed that someone else designed the IP addressing and subnetting plan, and this book shows how to implement it.

This chapter turns that model around. It goes back to the progression of building and implementing IPv4, as discussed in Chapter 11, as shown in Figure L-1. This chapter picks up the story right after some network engineer has chosen a Class A, B, or C network to use for the enterprise's IPv4 network. And then this chapter discusses the design choices related to picking one subnet mask to use for all subnets (the first major section) and what subnet IDs that choice creates (the second major section).

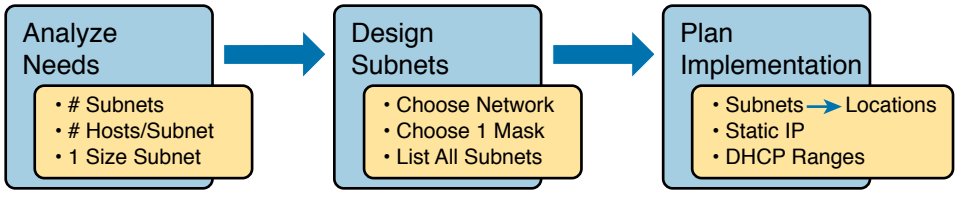


Figure L-1 *Subnet Design and Implementation Process from Chapter 11*

Foundation Topics

Choosing the Mask(s) to Meet Requirements

This first major section examines how to find all the masks that meet the stated requirements for the number of subnets and the number of hosts per subnet. To that end, the text assumes that the designer has already determined these requirements and has chosen the network number to be subnetted. The designer has also made the choice to use a single subnet mask value throughout the classful network.

Armed with the information in this chapter, you can answer questions such as the following, a question that matters both for real engineering jobs and the Cisco exams:

> You are using Class B network 172.16.0.0. You need 200 subnets and 200 hosts/subnet. Which of the following subnet mask(s) meet the requirements? (This question is then followed by several answers that list different subnet masks.)

To begin, this section reviews the concepts in Chapter 13's section "Choose the Mask." That section introduced the main concepts about how an engineer, when designing subnet conventions, must choose the mask based on the requirements.

After reviewing the related concepts from Chapter 13, this section examines this topic in more depth. In particular, this chapter looks at three general cases:

- No masks meet the requirements.
- One and only one mask meets the requirements.
- Multiple masks meet the requirements.

For this last case, the text discusses how to determine all masks that meet the requirements and the trade-offs related to choosing which one mask to use.

Review: Choosing the Minimum Number of Subnet and Host Bits

The network designer must examine the requirements for the number of subnets and number of hosts/subnet, and then choose a mask. As discussed in detail in Chapter 15, "Analyzing Subnet Masks," a classful view of IP addresses defines the three-part structure of an IP address: network, subnet, and host. The network designer must choose the mask so that the number of subnet and host bits (S and H, respectively, in Figure L-2) meet the requirements.

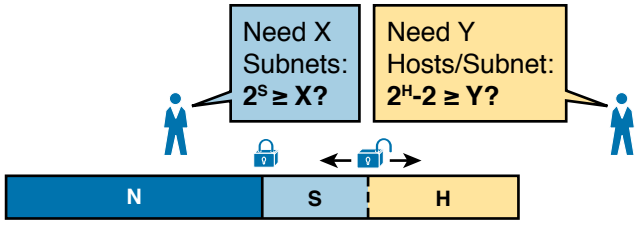


Figure L-2 *Choosing the Number of Subnet and Host Bits*

Basically, the designer must choose S subnet bits so that the number of subnets that can be uniquely numbered with S bits (2^S) is at least as large as the required number of subnets. The

designer applies similar logic to the number of host bits H, while noting that the formula is $2^H - 2$, because of the two reserved numbers in each subnet. So, keeping the powers of 2 handy, as shown in Table L-1, will be useful when working through these problems.

Table L-1 Powers of 2 Reference for Designing Masks

Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X
1	2	5	32	9	512	13	8192
2	4	6	64	10	1024	14	16,384
3	8	7	128	11	2048	15	32,768
4	16	8	256	12	4096	16	65,536

More formally, the process must determine the minimum values for both S and H that meet the requirements. The following list summarizes the initial steps to choose the mask:

Step 1. Determine the number of network bits (N) based on the class.

Step 2. Determine the smallest value of S, so that 2^S => X, where X represents the required number of subnets.

Step 3. Determine the smallest value of H, so that $2^H - 2$ => Y, where Y represents the required number of hosts/subnet.

The next three sections examine how to use these initial steps to choose a subnet mask.

No Masks Meet Requirements

After you determine the required number of subnet and host bits, those bits might not fit into a 32-bit IPv4 subnet mask. Remember, the mask always has a total of 32 bits, with binary 1s in the network and subnet parts and binary 0s in the host part. For the exam, a question might provide a set of requirements that simply cannot be met with 32 total bits.

For example, consider the following sample exam question:

> A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 300 subnets and 280 hosts per subnet. Which of the following masks could the engineer choose?

The three-step process shown in the previous section shows that these requirements mean that a total of 34 bits will be needed, so no mask meets the requirements. First, as a Class B network, 16 network bits exist, with 16 host bits from which to create the subnet part and to leave enough host bits to number the hosts in each subnet. For the number of subnet bits, S=8 does not work, because $2^8 = 256 < 300$. However, S=9 works, because $2^9 = 512$ => 300. Similarly, because $2^8 - 2 = 254$, which is less than 300, 8 host bits are not enough but 9 host bits ($2^9 - 2 = 510$) are just enough.

These requirements do not leave enough space to number all the hosts and subnet, because the network, subnet, and host parts add up to more than 32:

> N=16, because as a Class B network, 16 network bits exist.
>
> The minimum S=9, because S=8 provides too few subnets ($2^8 = 256 < 300$) but S=9 provides $2^9 = 512$ subnets.

The minimum H=9, because H=8 provides too few hosts ($2^8 - 2 = 254 < 280$) but H=9 provides $2^9 - 2 = 510$ hosts/subnet.

Figure L-3 shows the resulting format for the IP addresses in this subnet, after the engineer has allocated 9 subnet bits on paper. Only 7 host bits remain, but the engineer needs 9 host bits.

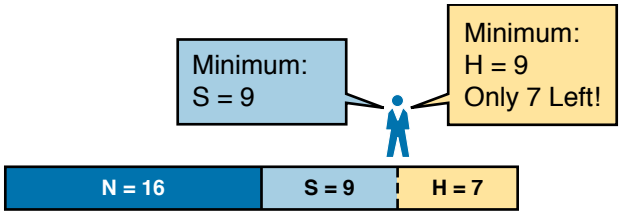


Figure L-3 *Too Few Bits for the Host Part, Given the Requirements*

One Mask Meets Requirements

The process discussed in this chapter in part focuses on finding the smallest number of subnet bits and the smallest number of host bits to meet the requirements. If the engineer tries to use these minimum values, and the combined network, subnet, and host parts add up to exactly 32 bits, exactly one mask meets the requirements.

For example, consider a revised version of the example in the previous section, with smaller numbers of subnet and hosts, as follows:

A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 200 subnets and 180 hosts per subnet. Which of the following masks could the engineer choose?

The three-step process to determine the numbers of network, minimum subnet, and minimum host bits results in a need for 16, 8, and 8 bits, respectively. As before, with a Class B network, 16 network bits exist. With a need for only 200 subnets, S=8 does work, because $2^8 = 256 => 200$; 7 subnet bits would not supply enough subnets ($2^7 = 128$). Similarly, because $2^8 - 2 = 254 => 180$, 8 host bits meet the requirements; 7 host bits (for 126 total hosts/subnet) would not be enough.

Figure L-4 shows the resulting format for the IP addresses in this subnet.

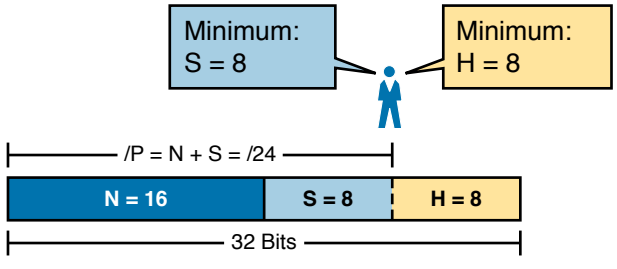


Figure L-4 *One Mask That Meets Requirements*

Figure L-4 shows the mask conceptually. To find the actual mask value, simply record the mask in prefix format (/P), where P = N + S or, in this case, /24.

Multiple Masks Meet Requirements

Depending on the requirements and choice of network, several masks might meet the requirements for the numbers of subnets and hosts/subnet. In these cases, you need to find all the masks that could be used. Then, you have a choice, but what should you consider when choosing one mask among all those that meet your requirements? This section shows how to find all the masks, as well as the facts to consider when choosing one mask from the list.

Finding All the Masks: Concepts

To help you better understand how to find all the subnet masks in binary, this section uses two major steps. In the first major step, you build the 32-bit binary subnet mask on paper. You write down binary 1s for the network bits, binary 1s for the subnet bits, and binary 0s for the host bits, just as always. However, you will use the minimum values for S and H. And when you write down these bits, you will not have 32 bits yet!

For example, consider the following problem, similar to the earlier examples in this chapter but with some changes in the requirements:

> A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 50 subnets and 180 hosts per subnet. Which of the following masks could the engineer choose?

This example is similar to an earlier example, except that only 50 subnets are needed in this case. Again, the engineer is using private IP network 172.16.0.0, meaning 16 network bits. The design requires only 6 subnet bits in this case, because $2^6 = 64 => 50$, and with only 5 subnet bits, $2^5 = 32 < 50$. The design then requires a minimum of 8 host bits.

One way to discuss the concepts and find all the masks that meet these requirements is to write down the bits in the subnet mask: binary 1s for the network and subnet parts and binary 0s for the host part. However, think of the 32-bit mask as 32-bit positions, and when writing the binary 0s, *write them on the far right*. Figure L-5 shows the general idea.

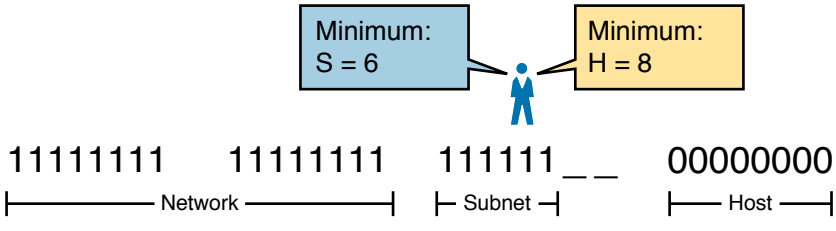


Figure L-5 *Incomplete Mask with N=16, S=6, and H=8*

Figure L-5 shows 30 bits of the mask, but the mask must have 32 bits. The 2 remaining bits might become subnet bits, being set to binary 1. Alternatively, these 2 bits could be made host bits, being set to binary 0. The engineer simply needs to choose based on whether he would like more subnet bits, to number more subnets, or more host bits, to number more hosts/subnet.

Regardless of the requirements, when choosing any IPv4 subnet mask, you must always follow this rule:

Key Topic

A subnet mask begins with all binary 1s, followed by all binary 0s, with no interleaving of 1s and 0s.

With the example shown in Figure L-5, with 2 open bits, one value (binary 01) breaks this rule. However, the other three combinations of 2 bits (00, 10, and 11) do not break the rule. As a result, three masks meet the requirements in this example, as shown in Figure L-6.

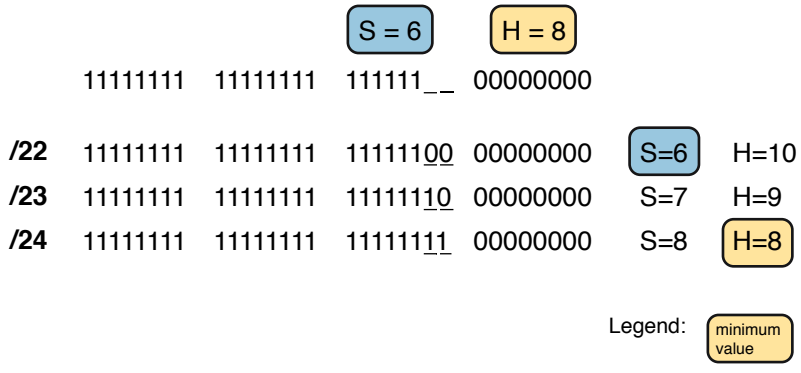


Figure L-6 *Three Masks That Meet the Requirements*

In the three masks, the first has the least number of subnet bits among the three masks, but therefore has the most number of host bits. So, the first mask maximizes the number of hosts/subnet. The last mask uses the minimum value for the number of host bits, therefore using the most number of subnet bits allowed while still meeting the requirements. As a result, the last mask maximizes the number of subnets allowed.

Finding All the Masks: Math

Although the concepts related to the example shown in Figures L-5 and L-6 are important, you can find the range of masks that meets the requirements more easily just using some simple math. The process to find the masks just requires a few steps, after you know N and the minimum values of S and H. The process finds the value of /P when using the least number of subnet bits, and when using the least number of host bits, as follows:

Key Topic

Step 1. Calculate the shortest prefix mask (/P) based on the *minimum value of S*, where P = N + S.

Step 2. Calculate the longest prefix mask (/P) based on the *minimum value of H*, where P = 32 – H.

Step 3. The range of valid masks includes all /P values between the two values calculated in the previous steps.

For example, in the example shown in Figure L-6, N = 16, the minimum S = 6, and the minimum H = 8. The first step identifies the shortest prefix mask (the /P with the smallest value of P) of /22 by adding N and S (16 + 6). The second step identifies the longest prefix mask that meets the requirements by subtracting the smallest possible value for H (8, in this

case) from 32, for a mask of /24. The third step reminds us that the range is from /22 to /24, meaning that /23 is also an option.

Choosing the Best Mask

When multiple possible masks meet the stated requirements, the engineer has a choice of masks. That, of course, begs some questions: Which mask should you choose? Why would one mask be better than the other? The reasons can be summarized into three main options:

Key Topic

To maximize the number of hosts/subnet: To make this choice, use the shortest prefix mask (that is, the mask with the smallest /P value), because this mask has the largest host part.

To maximize the number of subnets: To make this choice, use the longest prefix mask (that is, the mask with the largest /P value), because this mask has the largest subnet part.

To increase both the numbers of supported subnets and hosts: To make this choice, choose a mask in the middle of the range, which gives you both more subnet bits and more host bits.

For example, in Figure L-6, the range of masks that meet the requirements is /22 – /24. The shortest mask, /22, has the least subnet bits but the largest number of host bits (10) of the three answers, maximizing the number of hosts/subnet. The longest mask, /24, maximizes the number of subnet bits (8), maximizing the number of subnets, at least among the options that meet the original requirements. The mask in the middle, /23, provides some growth in both subnets and hosts/subnet.

The Formal Process

Although this chapter has explained various steps in finding a subnet mask to meet the design requirements, it has not yet collected these concepts into a list for the entire process. The following list collects all these steps into one place for reference. Note that this list does not introduce any new concepts compared to the rest of this chapter; it just puts all the ideas in one place.

Key Topic

Step 1. Find the number of network bits (N) per class rules.

Step 2. Calculate the minimum number of subnet bits (S) so that 2^S => the number of required subnets.

Step 3. Calculate the minimum number of host bits (H) so that $2^H - 2$ => the number of required hosts/subnet.

Step 4. If N + S + H > 32, no mask meets the need.

Step 5. If N + S + H = 32, one mask meets the need. Calculate the mask as /P, where P = N + S.

Step 6. If N + S + H < 32, multiple masks meet the need:

- **A.** Calculate mask /P based on the minimum value of S, where P = N + S. This mask maximizes the number of hosts/subnet.
- **B.** Calculate mask /P based on the minimum value of H, where P = 32 – H. This mask maximizes the number of possible subnets.
- **C.** Note that the complete range of masks includes all prefix lengths between the two values calculated in Steps 6A and 6B.

Practice Choosing Subnet Masks

Take the usual two-phase approach to learning new subnetting math and processes. Take the time now to practice to make sure you understand the fundamentals, using the book and notes as needed. Then, sometime before taking the exam, practice until you can reach the goals in the right column of Table L-2.

Table L-2 Keep-Reading and Take-Exam Goals for Choosing a Subnet Mask

Time Frame	Before Moving to the Next Chapter	Before Taking the Exam
Focus On	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	15 seconds

Practice Problems for Choosing a Subnet Mask

The following list shows three separate problems, each with a classful network number and a required number of subnets and hosts/subnet. For each problem, determine the minimum number of subnet and host bits that meet the requirements. If more than one mask exists, note which mask maximizes the number of hosts/subnet and which maximizes the number of subnets. If only one mask meets the requirements, simply list that mask. List the masks in prefix format:

1. Network 10.0.0.0, need 1500 subnets, need 300 hosts/subnet
2. Network 172.25.0.0, need 130 subnets, need 127 hosts/subnet
3. Network 192.168.83.0, need 8 subnets, need 8 hosts/subnet

Table L-7, found in the later section "Answers to Earlier Practice Problems," lists the answers.

Finding All Subnet IDs

After the person designing the IP subnetting plan has chosen the one mask to use throughout the Class A, B, or C network, he will soon need to start assigning specific subnet IDs for use in specific VLANs, serial links, and other places in the internetwork that need a subnet. But what are those subnet IDs? As it turns out, after the network ID and one subnet mask for all subnets have been chosen, finding all the subnet IDs just requires doing a little math. This second major section of this chapter focuses on that math, which focuses on a single question:

> Given a single Class A, B, or C network, and the single subnet mask to use for all subnets, what are all the subnet IDs?

When learning how to answer this question, you can think about the problem in either binary or decimal. This chapter approaches the problem using decimal. Although the process itself requires only simple math, the process requires practice before most people can confidently answer this question.

The decimal process begins by identifying the first, or numerically lowest, subnet ID. After that, the process identifies a pattern in all subnet IDs for a given subnet mask so that you can find each successive subnet ID through simple addition. This section examines the key ideas behind this process first; then you are given a formal definition of the process.

NOTE Some videos included on the companion website describe the same fundamental processes to find all subnet IDs. You can view those videos before or after reading this section, or even instead of reading this section, as long as you learn how to independently find all subnet IDs. The process step numbering in the videos might not match the steps shown in this edition of the book.

First Subnet ID: The Zero Subnet

The first step in finding all subnet IDs of one network is incredibly simple: Copy the network ID. That is, take the Class A, B, or C network ID—in other words, the classful network ID—and write it down as the first subnet ID. No matter what Class A, B, or C network you use, and no matter what subnet mask you use, the first (numerically lowest) subnet ID is equal to the network ID.

For example, if you begin with classful network 172.20.0.0, no matter what the mask is, the first subnet ID is 172.20.0.0.

This first subnet ID in each network goes by two special names: either s*ubnet zero* or the *zero subnet*. The origin of these names is related to the fact that a network's zero subnet, when viewed in binary, has a subnet part of all binary 0s. In decimal, the zero subnet can be easily identified, because the zero subnet always has the exact same numeric value as the network ID itself.

In the past, engineers avoided using zero subnets because of the ambiguity with one number that could represent the entire classful network or it could represent one subnet inside the classful network. To help control that, IOS has a global command that can be set one of two ways:

ip subnet-zero, which allows the configuration of addresses in the zero subnet.

no ip subnet-zero, which prevents the configuration of addresses in the zero subnet.

Although most sites use the default setting to allow zero subnets, you can use the **no ip subnet-zero** command to prevent configuring addresses that are part of a zero subnet. Example L-1 shows how a router rejects an **ip address** command after changing to use **no ip subnet-zero**. Note that the error message does not mention the zero subnet, instead simply stating "bad mask."

Example L-1 *Effects of* **[no] ip subnet-zero** *on a Local Router*

```
R1# configure terminal
Enter configuration commands, one per line.   End with CNTL/Z.
R1(config)# no ip subnet-zero
R1(config)# interface g0/1
R1(config-if)# ip address 10.0.0.1 255.255.255.0
Bad mask /24 for address 10.0.0.1
```

Note that the **no ip subnet-zero** command affects the local router's **ip address** commands, as well as the local router's **ip route** commands (which define static routes). However, it does not affect the local router's routes as learned with a routing protocol.

Finding the Pattern Using the Magic Number

Subnet IDs follow a predictable pattern, at least when using our assumption of a single subnet mask for all subnets of a network. The pattern uses the *magic number*, as discussed in Chapter 14, "Analyzing Existing Subnets." To review, the magic number is 256, minus the mask's decimal value, in a particular octet that this book refers to as the *interesting octet*.

Figure L-7 shows four examples of these patterns with four different masks. For example, just look at the top of the figure to start. It lists mask 255.255.128.0 on the left. The third octet is the interesting octet, with a mask value other than 0 or 255 in that octet. The left side shows a magic number calculated as 256 – 128 = 128. So, the pattern of subnet IDs is shown in the highlighted number line; that is, the subnet IDs when using this mask will have either a 0 or 128 in the third octet. For example, if using network 172.16.0.0, the subnet IDs would be 172.16.0.0 and 172.16.128.0.

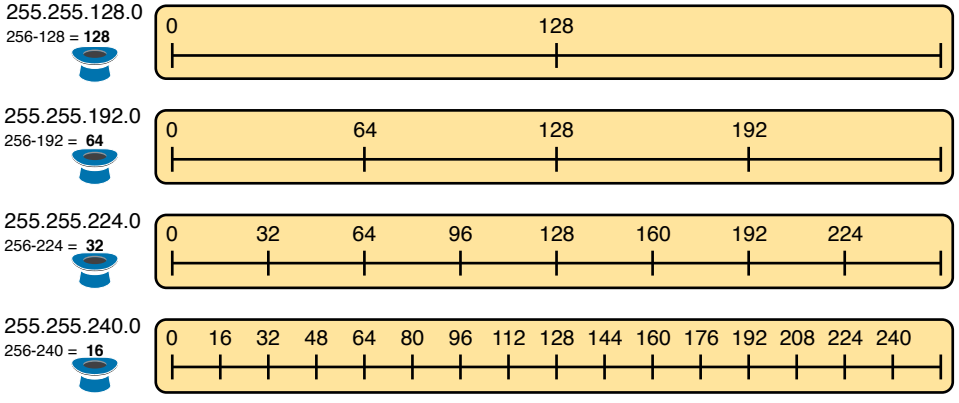


Figure L-7 *Patterns with Magic Numbers for Masks /17 – /20*

Now focus on the second row, with another example, with mask 255.255.192.0. This row shows a magic number of 64 (256 – 192 = 64), so the subnet IDs will use a value of 0, 64, 128, or 192 (multiples of 64) in the third octet. For example, if used with network 172.16.0.0, the subnet IDs would be 172.16.0.0, 172.16.64.0, 172.16.128.0, and 172.16.192.0.

Looking at the third row/example, the mask is 255.255.224.0, with a magic number of 256 – 224 = 32. So, as shown in the center of the figure, the subnet ID values will be multiples of 32. For example, if used with network 172.16.0.0 again, this mask would tell us that the subnet IDs are 172.16.0.0, 172.16.32.0, 172.16.64.0, 172.16.96.0, and so on.

Finally, for the bottom example, mask 255.255.240.0 makes the magic number, in the third octet, be 16. So, all the subnet IDs will be a multiple of 16 in the third octet, with those values shown in the middle of the figure.

A Formal Process with Less Than 8 Subnet Bits

Although it can be easy to see the patterns in Figure L-7, it might not be as obvious exactly how to apply those concepts to find all the subnet IDs in every case. This section outlines a specific process to find all the subnet IDs.

To simplify the explanations, this section assumes that less than 8 subnet bits exist. Later, the section "Finding All Subnets with More Than 8 Subnet Bits," describes the full process that can be used in all cases.

First, to organize your thoughts, you might want to organize the data into a table like Table L-3. This book refers to this chart as the list-all-subnets chart.

Table L-3 Generic List-All-Subnets Chart

Octet	1	2	3	4
Mask				
Magic Number				
Network Number/Zero Subnet				
Next Subnet				
Next Subnet				
Next Subnet				
Broadcast Subnet				
Out of Range—Used by Process				

A formal process to find all subnet IDs, given a network and a single subnet mask, is as follows:

Key Topic

Step 1. Write down the subnet mask, in decimal, in the first empty row of the table.

Step 2. Identify the interesting octet, which is the one octet of the mask with a value other than 255 or 0. Draw a rectangle around the column of the interesting octet.

Step 3. Calculate and write down the magic number by subtracting the *subnet mask's interesting octet* from 256.

Step 4. Write down the classful network number, which is the same number as the zero subnet, in the next empty row of the list-all-subnets chart.

Step 5. To find each successive subnet number:

A. For the three uninteresting octets, copy the previous subnet number's values.

B. For the interesting octet, add the magic number to the previous subnet number's interesting octet.

Step 6. When the sum calculated in Step 5B reaches 256, stop the process. The number with the 256 in it is out of range, and the previous subnet number is the broadcast subnet.

Although the written process is long, with practice, most people can find the answers much more quickly with this decimal-based process than by using binary math. As usual, most people learn this process best by seeing it in action, exercising it, and then practicing it. To that end, review the two following examples and watch any videos that came with this book that show additional examples.

Example 1: Network 172.16.0.0, Mask 255.255.240.0

To begin this example, focus on the first four of the six steps, when subnetting network 172.16.0.0 using mask 255.255.240.0. Figure L-8 shows the results of these first four steps:

Step 1. Record mask 255.255.240.0, which was given as part of the problem statement. (Figure L-8 also shows the network ID, 172.16.0.0, for easy reference.)

Step 2. The mask's third octet is neither 0 nor 255, which makes the third octet interesting.

Step 3. Because the mask's value in the third octet is 240, the magic number = 256 – 240 = 16.

Step 4. Because the network ID is 172.16.0.0, the first subnet ID, the zero subnet, is also 172.16.0.0.

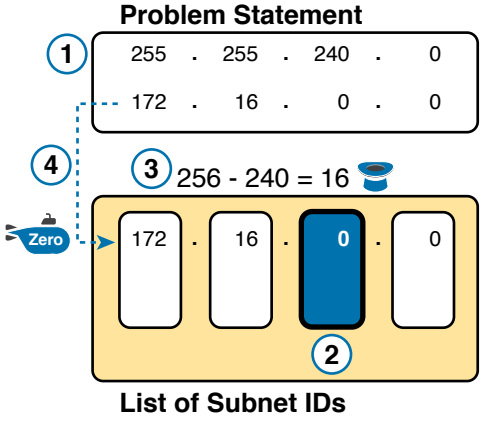


Figure L-8 *Results of First Four Steps: 172.16.0.0, 255.255.240.0.*

These first four steps discover the first subnet (the zero subnet) and get you ready to do the remaining steps by identifying the interesting octet and the magic number. Step 5 in the process tells you to copy the three boring octets and add the magic number (16, in this case) in the interesting octet (octet 3, in this case). Keep repeating this step until the interesting octet value equals 256 (per Step 6). When the total is 256, you have listed all the subnet IDs, and the line with 256 on it is not a correct subnet ID. Figure L-9 shows the results of the Step 5 actions.

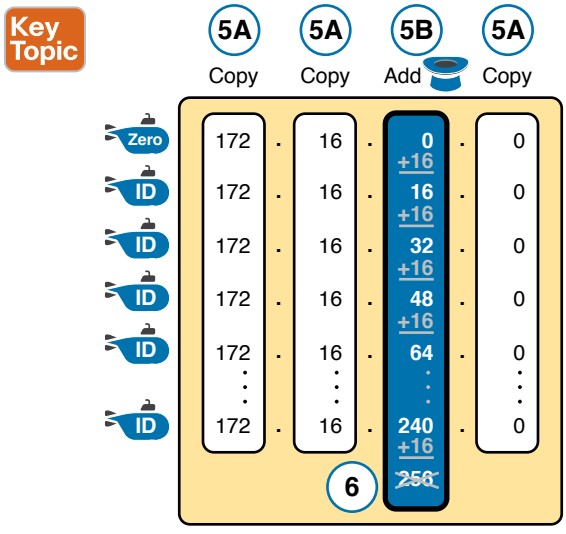


Figure L-9 *List of Subnet IDs: 172.16.0.0, 255.255.240.0*

NOTE In any list of all the subnet IDs of a network, the numerically highest subnet ID is called the *broadcast subnet*. Decades ago, engineers avoided using the broadcast subnet. However, using the broadcast subnet causes no problems. The term *broadcast subnet* has its origins in the fact that if you determine the subnet broadcast address inside the broadcast subnet, it has the same numeric value as the network-wide broadcast address.

NOTE People sometimes confuse the terms *broadcast subnet* and *subnet broadcast address*. The *broadcast subnet* is one subnet, namely the numerically highest subnet; only one such subnet exists per network. The term *subnet broadcast address* refers to the one number in each and every subnet that is the numerically highest number in that subnet.

Example 2: Network 192.168.1.0, Mask 255.255.255.224

With a Class C network and a mask of 255.255.255.224, this example makes the fourth octet the interesting octet. However, the process works the same, with the same logic, just with the interesting logic applied in a different octet. As with the previous example, the following list outlines the first four steps, with Figure L-10 showing the results of the first four steps:

Step 1. Record mask 255.255.255.224, which was given as part of the problem statement, and optionally record the network number (192.168.1.0).

Step 2. The mask's fourth octet is neither 0 nor 255, which makes the fourth octet interesting.

Step 3. Because the mask's value in the fourth octet is 224, the magic number = 256 – 224 = 32.

Step 4. Because the network ID is 192.168.1.0, the first subnet ID, the zero subnet, is also 192.168.1.0.

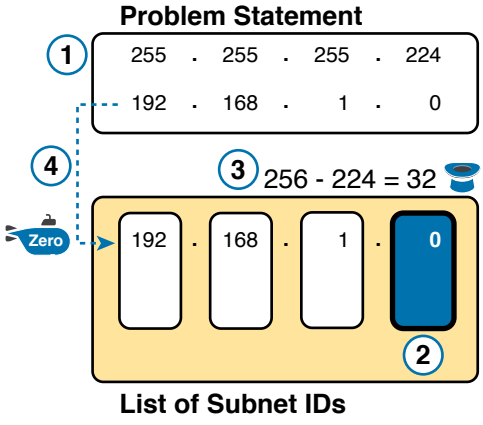


Figure L-10 *Results of First Four Steps: 192.168.1.0, 255.255.255.224*

From this point, Step 5 in the process tells you to copy the values in the first three octets and then add the magic number (32, in this case) in the interesting octet (octet 4, in this case). Keep doing so until the interesting octet value equals 256 (per Step 6). When the total is 256, you have listed all the subnet IDs, and the line with 256 on it is not a correct subnet ID. Figure L-11 shows the results of these steps.

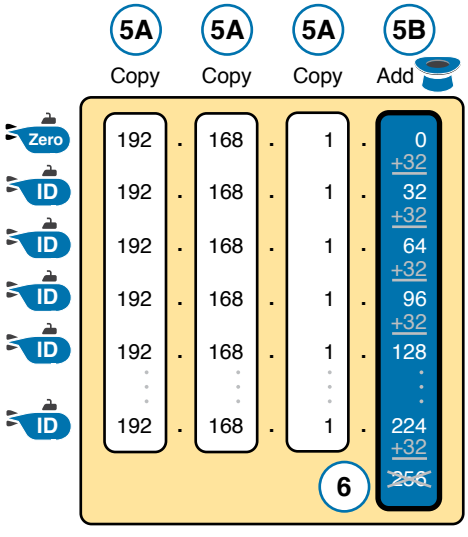


Figure L-11 *List of Subnet IDs: 192.168.1.0, 255.255.255.224*

Finding All Subnets with Exactly 8 Subnet Bits

The formal process in the earlier section "A Formal Process with Less Than 8 Subnet Bits" identified the interesting octet as the octet whose mask value is neither a 255 nor a 0. If the mask defines exactly 8 subnet bits, you must use a different logic to identify the interesting octet; otherwise, the same process can be used. In fact, the actual subnet IDs can be a little more intuitive.

Only two cases exist with exactly 8 subnet bits:

A Class A network with mask 255.255.0.0; the entire second octet contains subnet bits.

A Class B network with mask 255.255.255.0; the entire third octet contains subnet bits.

In each case, use the same process as with less than 8 subnet bits, but identify the interesting octet as the one octet that contains subnet bits. Also, because the mask's value is 255, the magic number will be 256 – 255 = 1, so the subnet IDs are each 1 larger than the previous subnet ID.

For example, for 172.16.0.0, mask 255.255.255.0, the third octet is the interesting octet and the magic number is 256 – 255 = 1. You start with the zero subnet, equal in value to network number 172.16.0.0, and then add 1 in the third octet. For example, the first four subnets are as follows:

172.16.0.0 (zero subnet)

172.16.1.0

172.16.2.0

172.16.3.0

Finding All Subnets with More Than 8 Subnet Bits

Earlier, the section "A Formal Process with Less Than 8 Subnet Bits" assumed less than 8 subnet bits for the purpose of simplifying the discussions while you learn. In real life, you need to be able to find all subnet IDs with any valid mask, so you cannot assume less than 8 subnet bits.

The examples that have at least 9 subnet bits have a minimum of 512 subnet IDs, so writing down such a list would take a lot of time. To conserve space, the examples will use shorthand rather than list hundreds or thousands of subnet IDs.

The process with less than 8 subnet bits told you to count in increments of the magic number in one octet. With more than 8 subnet bits, the new expanded process must tell you how to count in multiple octets. So, this section breaks down two general cases: (a) when 9–16 subnet bits exist, which means that the subnet field exists in only two octets, and (b) cases with 17 or more subnet bits, which means that the subnet field exists in three octets.

Process with 9–16 Subnet Bits

To understand the process, you need to know a few terms that the process will use. Figure L-12 shows the details, with an example that uses Class B network 130.4.0.0 and mask 255.255.255.192. The lower part of the figure details the structure of the addresses per the mask: a network part of two octets because it is a Class B address, a 10-bit subnet part per the mask (/26), and 6 host bits.

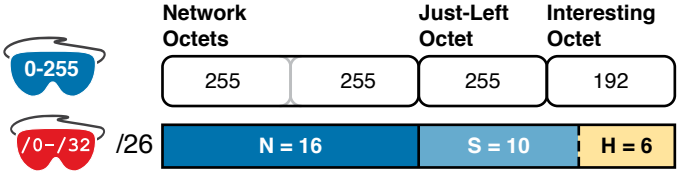


Figure L-12 *Fundamental Concepts and Terms for the >8 Subnet Bit Process*

In this case, subnet bits exist in two octets: octets 3 and 4. For the purposes of the process, the rightmost of these octets is the interesting octet, and the octet just to the left is the cleverly named *just-left* octet.

The updated process, which makes adjustments for cases in which the subnet field is longer than 1 octet, tells you to count in increments of the magic number in the interesting octet, but count by 1s in the just-left octet. Formally:

Key Topic

Step 1. Calculate subnet IDs using the 8-subnet-bits-or-less process. However, when the total adds up to 256, move to the next step; consider the subnet IDs listed so far as a *subnet block*.

Step 2. Copy the previous subnet block, but add 1 to the just-left octet in all subnet IDs in the new block.

Step 3. Repeat Step 2 until you create the block with a just-left octet of 255, but go no further.

To be honest, the formal concept can cause you problems until you work through some examples, so even if the process remains a bit unclear in your mind, you should work through the following examples instead of rereading the formal process.

First, consider an example based on Figure L-12, with network 130.4.0.0 and mask 255.255.255.192. Figure L-12 already showed the structure, and Figure L-13 shows the subnet ID block created at Step 1.

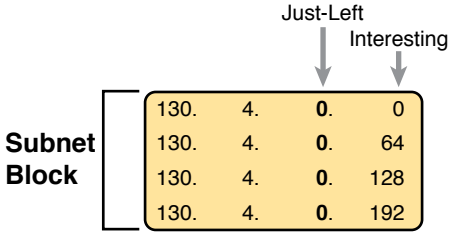


Figure L-13 *Step 1: Listing the First Subnet ID Block*

The logic at Step 1, to create this subnet ID block of four subnet IDs, follows the same magic number process seen before. The first subnet ID, 130.4.0.0, is the zero subnet. The next three subnet IDs are each 64 bigger, because the magic number, in this case, is 256 – 192 = 64.

Steps 2 and 3 from the formal process tell you how to create 256 subnet blocks, and by doing so, you will list all 1024 subnet IDs. To do so, create 256 total subnet blocks: one with a 0 in the just-left octet, one with a 1 in the just-left octet, and another with a 2 in the just-left octet, up through 255. The process continues through the step at which you create the subnet block with 255 in the just-left octet (third octet, in this case). Figure L-14 shows the idea, with the addition of the first few subnet blocks.

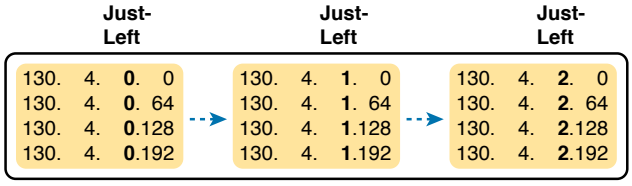


Figure L-14 *Step 2: Replicating the Subnet Block with +1 in the Just-Left Octet*

This example, with 10 total subnet bits, creates 256 blocks of four subnets each, for a total of 1024 subnets. This math matches the usual method of counting subnets, because $2^{10} = 1024$.

L

Process with 17 or More Subnet Bits

To create a subnet design that allows 17 or more subnet bits to exist, the design must use a Class A network. In addition, the subnet part will consist of the entire second and third octets, plus part of the fourth octet. That means a lot of subnet IDs: at least 2^{17} (or 131,072) subnets. Figure L-15 shows an example of just such a structure, with a Class A network and a /26 mask.

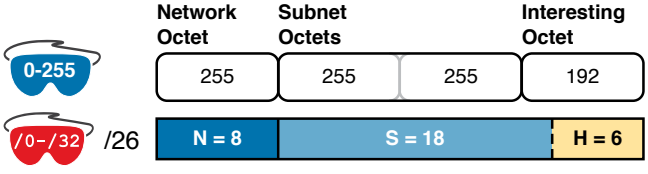


Figure L-15 *Address Structure with 18 Subnet Bits*

To find all the subnet IDs in this example, you use the same general process as with 9–16 subnet bits, but with many more subnet blocks to create. In effect, you have to create a subnet block for all combinations of values (0–255, inclusive) in both the second and third octet. Figure L-16 shows the general idea. Note that with only 2 subnet bits in the fourth octet in this example, the subnet blocks will have four subnets each.

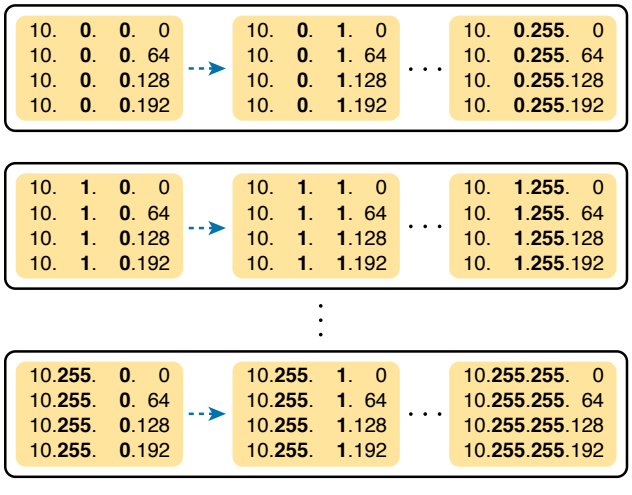


Figure L-16 *256 Times 256 Subnet Blocks of Four Subnets*

Practice Finding All Subnet IDs

Before moving to the next chapter, practice until you get the right answer most of the time—but use any tools you want and take all the time you need. Then, you can move on with your reading. *Before taking the exam*, practice until you reach the goals in the right column of Table L-4, which summarizes the key concepts and suggestions for this two-phase approach.

Table L-4 Keep-Reading and Take-Exam Goals for This Chapter's Topics

Time Frame	Before Moving to the Next Chapter	Before Taking the Exam
Focus On	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	45 seconds

Practice Problems for Finding All Subnet IDs

The following list shows three separate problems, each with a classful network number and prefix-style mask. Find all subnet IDs for each problem:

1. 192.168.9.0/27
2. 172.30.0.0/20
3. 10.0.0.0/17

The section "Answers to Earlier Practice Problems," later in this chapter, lists the answers.

Answers to Earlier Practice Problems

Answers to Practice Choosing Subnet Masks

The earlier section "Practice Choosing Subnet Masks" listed three practice problems. The answers are listed here so that the answers are nearby but not visible from the list of problems. Table L-5 lists the answers, with notes related to each problem following the table.

Table L-5 Practice Problems: Find the Masks That Meet Requirements

Problem	Class	Minimum Subnet Bits	Minimum Host Bits	Prefix Range	Prefix to Maximize Subnets	Prefix to Maximize Hosts
1	A	11	9	/19 – /23	/23	/19
2	B	8	8	/24	—	—
3	C	3	4	/27 – /28	/28	/27

1. N=8, because the problem lists Class A network 10.0.0.0. With a need for 1500 subnets, 10 subnet bits supply only 1024 subnets (per Table L-1), but 11 subnet bits (S) would provide 2048 subnets—more than the required 1500. Similarly, the smallest number of host bits would be 9, because $2^8 - 2 = 254$, and the design requires 300

hosts/subnet. The shortest prefix mask would then be /19, found by adding N (8) and the smallest usable number of subnet bits S (11). Similarly, with a minimum H value of 9, the longest prefix mask, maximizing the number of subnets, is 32 – H = /23.

2. N=16, because the problem lists Class B network 172.25.0.0. With a need for 130 subnets, 7 subnet bits supply only 128 subnets (per Table L-1), but 8 subnet bits (S) would provide 256 subnets—more than the required 130. Similarly, the smallest number of host bits would be 8, because $2^7 - 2 = 126$—close to the required 127, but not quite enough, making H = 8 the smallest number of host bits that meets requirements. Note that the network, minimum subnet bits, and minimum host bits add up to 32, so only one mask meets the requirements, namely /24, found by adding the number of network bits (16) to the minimum number of subnet bits (8).
3. N=24, because the problem lists Class C network 192.168.83.0. With a need for eight subnets, 3 subnet bits supply enough, but just barely. The smallest number of host bits would be 4, because $2^3 - 2 = 6$, and the design requires 8 hosts/subnet. The shortest prefix mask would then be /27, found by adding N (24) and the smallest usable number of subnet bits S (3). Similarly, with a minimum H value of 4, the longest prefix mask, maximizing the number of subnets, is 32 – H = /28.

Answers to Practice Finding All Subnet IDs

The earlier section "Practice Finding All Subnet IDs" listed three practice problems. The answers are listed here so that they are not visible from the same page as the list of problems.

Answer, Practice Problem 1

Problem 1 lists network 192.168.9.0, mask /27. The mask converts to DDN mask 255.255.255.224. When used with a Class C network, which has 24 network bits, only 3 subnet bits exist, and they all sit in the fourth octet. So, this problem is a case of less than 8 subnet bits, with the fourth octet as the interesting octet.

To get started listing subnets, first write down the zero subnet and then start adding the magic number in the interesting octet. The zero subnet equals the network ID (192.168.9.0, in this case). The magic number, calculated as 256 – 224 = 32, should be added to the previous subnet ID's interesting octet. Table L-6 lists the results.

Table L-6 List-All-Subnets Chart: 192.168.9.0/27

Octet	1	2	3	4
Mask	255	255	255	224
Magic Number	—	—	—	32
Classful Network/Subnet Zero	192	168	9	0
First Nonzero Subnet	192	168	9	32
Next Subnet	192	168	9	64
Next Subnet	192	168	9	96
Next Subnet	192	168	9	128
Next Subnet	192	168	9	160

Octet	1	2	3	4
Next Subnet	192	168	9	192
Broadcast Subnet	192	168	9	224
Invalid—Used by Process	192	168	9	256

Answer, Practice Problem 2

Problem 2 lists network 172.30.0.0, mask /20. The mask converts to DDN mask 255.255.240.0. When used with a Class B network, which has 16 network bits, only 4 subnet bits exist, and they all sit in the third octet. So, this problem is a case of less than 8 subnet bits, with the third octet as the interesting octet.

To get started listing subnets, first write down the zero subnet and then start adding the magic number in the interesting octet. The zero subnet equals the network ID (or 172.30.0.0, in this case). The magic number, calculated as 256 – 240 = 16, should be added to the previous subnet ID's interesting octet. Table L-7 lists the results.

Table L-7 List-All-Subnets Chart: 172.30.0.0/20

Octet	1	2	3	4
Mask	255	255	240	0
Magic Number	—	—	16	—
Classful Network/Subnet Zero	172	30	0	0
First Nonzero Subnet	172	30	16	0
Next Subnet	172	30	32	0
Next Subnet	172	30	Skipping…	0
Next Subnet	172	30	224	0
Broadcast Subnet	172	30	240	0
Invalid—Used by Process	172	30	256	0

Answer, Practice Problem 3

Problem 3 lists network 10.0.0.0, mask /17. The mask converts to DDN mask 255.255.128.0. When used with a Class A network, which has 8 network bits, 9 subnet bits exist. Using the terms unique to this chapter, octet 3 is the interesting octet, with only 1 subnet bit in that octet, and octet 2 is the just-left octet, with 8 subnet bits.

In this case, begin by finding the first subnet block. The magic number is 256 – 128 = 128. The first subnet (zero subnet) equals the network ID. So, the first subnet ID block includes the following:

10.0.0.0

10.0.128.0

Then, you create a subnet block for all 256 possible values in the just-left octet, or octet 2 in this case. The following list shows the first three subnet ID blocks, plus the last subnet ID block, rather than listing page upon page of subnet IDs:

10.0.0.0 (zero subnet)

10.0.128.0

10.1.0.0

10.1.128.0

10.2.0.0

10.2.128.0

…

10.255.0.0

10.255.128.0 (broadcast subnet)

This page intentionally left blank

APPENDIX M

Practice for Appendix L: Subnet Design

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Appendix G of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

This appendix exists as two halves to match the two major sections of the chapter. The first half lists mask design problems, and then the answers to those problems. The second half lists problems where you need to find the subnet ID, but with less than 8 subnet bits and with more than 8 subnet bits.

To solve these problems, use the processes explained in Appendix L of *CCNA 200-301 Official Cert Guide, Volume 1*, the current edition of this book you are reading.

Mask Design Practice Problems

This section lists problems with a short set of requirements regarding how a particular classful network should be subnetted. The requirements include the classful network, the number of subnets the design must support, and the number of hosts in each subnet. For each problem, supply the following information:

- The minimum number of subnet and host bits needed in the mask to support the design requirements
- The dotted-decimal format mask(s) that meet the requirements
- The mask you would choose if the problem said to maximize the number of subnets
- The mask you would choose if the problem said to maximize the number of hosts per subnet

Also note that you should assume that the two special subnets in each network—the zero subnet and broadcast subnet—are allowed to be used for these questions.

When doing the problems, the information in Table M-1 can be helpful. Note that Appendix A, "Numeric Reference Tables," in the printed book, also includes this table.

Table M-1 Powers of 2

Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X
1	2	5	32	9	512	13	8192
2	4	6	64	10	1024	14	16,384
3	8	7	128	11	2048	15	32,768
4	16	8	256	12	4096	16	65,536

Find the key facts for these sets of requirements:

1. Network 10.0.0.0, need 50 subnets, need 200 hosts/subnet
2. Network 172.32.0.0, need 125 subnets, need 125 hosts/subnet
3. Network 192.168.44.0, need 15 subnets, need 6 hosts/subnet
4. Network 10.0.0.0, need 300 subnets, need 500 hosts/subnet
5. Network 172.32.0.0, need 500 subnets, need 15 hosts/subnet
6. Network 172.16.0.0, need 2000 subnets, need 2 hosts/subnet

Mask Design Answers

This section includes the answers to the six problems listed in this appendix. The answer section for each problem explains how to use the process outlined in Appendix L, "Subnet Design," to find the answers.

Answer to Mask Design Problem 1

Problem 1 shows a Class A network, with 8 network bits, with a minimum of 6 subnet bits and 8 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.252.0.0 (maximizes the number of hosts per subnet)
- 255.254.0.0
- 255.255.0.0
- 255.255.128.0
- 255.255.192.0
- 255.255.224.0
- 255.255.240.0
- 255.255.248.0
- 255.255.252.0
- 255.255.254.0
- 255.255.255.0 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

NOTE The following explanation uses step numbers that match the process listed in Appendix L, but only the steps from that process that apply to this problem. As a result, the step numbers in the explanation are not sequential.

Step 1. The question lists Class A network 10.0.0.0, so there are 8 network bits.

Step 2. The question states that 50 subnets are needed. A mask with 5 subnet bits supplies only 2^5 (32) subnets, but a mask with 6 subnet bits supplies 2^6 (64) subnets. So, the mask needs at least 6 subnet bits.

Step 3. The question states that 200 hosts are needed per subnet. A mask with 7 host bits supplies only $2^7 - 2$ (126) hosts per subnet, but a mask with 8 host bits supplies $2^8 - 2$ (254) hosts per subnet. So, the mask needs at least 8 host bits.

Step 6A. With N=8, a minimum S=6, and a minimum H=8, multiple masks exist. The first mask, with the minimum number of subnet bits, is /14, found by adding N (8) to the minimum value of S (6). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 8. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 8 = /24.

Step 6C. All masks between /14 and /24 also meet the requirements.

M

Answer to Mask Design Problem 2

Problem 2 shows a Class B network, with 16 network bits, with a minimum of 7 subnet bits and 7 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.254.0 (maximizes the number of hosts/subnet)
- 255.255.255.0
- 255.255.255.128 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.32.0.0, so there are 16 network bits.

Step 2. The question states that 125 subnets are needed. A mask with 6 subnet bits supplies only 2^6 (64) subnets, but a mask with 7 subnet bits supplies 2^7 (128) subnets. So, the mask needs at least 7 subnet bits.

Step 3. The question states that 125 hosts are needed per subnet. A mask with 6 host bits supplies only $2^6 - 2$ (62) hosts per subnet, but a mask with 7 host bits supplies $2^7 - 2$ (126) hosts per subnet. So, the mask needs at least 7 host bits.

Step 6A. With N=16, a minimum S=7, and a minimum H=7, multiple masks exist. The first mask, with the minimum number of subnet bits, is /23, found by adding N (16) to the minimum value of S (7). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 7. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 7 = /25.

Step 6C. All masks between /23 and /25 also meet the requirements (/23, /24, and /25).

Answer to Mask Design Problem 3

Problem 3 shows a Class C network, with 24 network bits, with a minimum of 4 subnet bits and 3 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.240 (maximizes the number of hosts/subnet)
- 255.255.255.248 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class C network 192.168.44.0, so there are 24 network bits.

Step 2. The question states that 15 subnets are needed. A mask with 3 subnet bits supplies only 2^3 (8) subnets, but a mask with 4 subnet bits supplies 2^4 (16) subnets. So, the mask needs at least 4 subnet bits.

Step 3. The question states that 6 hosts are needed per subnet. A mask with 2 host bits supplies only $2^2 – 2$ (2) hosts per subnet, but a mask with 3 host bits supplies $2^3 – 2$ (6) hosts per subnet. So, the mask needs at least 3 host bits.

Step 6A. With N=24, a minimum S=4, and a minimum H=3, multiple masks exist. The first mask, with the minimum number of subnet bits, is /28, found by adding N (24) to the minimum value of S (4). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 3. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 3 = /29.

Step 6C. Only masks /28 and /29 meet the requirements.

Answer to Mask Design Problem 4

Problem 4 shows a Class A network, with 8 network bits, with a minimum of 9 subnet bits and 9 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.128.0 (maximizes the number of hosts/subnet)
- 255.255.192.0
- 255.255.224.0
- 255.255.240.0
- 255.255.248.0
- 255.255.252.0
- 255.255.254.0 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class A network 10.0.0.0, so there are 8 network bits.

Step 2. The question states that 300 subnets are needed. A mask with 8 subnet bits supplies only 2^8 (256) subnets, but a mask with 9 subnet bits supplies 2^9 (512) subnets. So, the mask needs at least 9 subnet bits.

Step 3. The question states that 500 hosts are needed per subnet. A mask with 8 host bits supplies only $2^8 - 2$ (254) hosts per subnet, but a mask with 9 host bits supplies $2^9 - 2$ (510) hosts per subnet. So, the mask needs at least 9 host bits.

Step 6A. With N=8, a minimum S=9, and a minimum H=9, multiple masks exist. The first mask, with the minimum number of subnet bits, is /17, found by adding N (8) to the minimum value of S (9). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 9. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 9 = /23.

Step 6C. All masks between /17 and /23 also meet the requirements (/17, /18, /19, /20, /21, /22, /23).

Answer to Mask Design Problem 5

Problem 5 shows a Class B network, with 16 network bits, with a minimum of 9 subnet bits and 5 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.128 (maximizes the number of hosts/subnet)
- 255.255.255.192
- 255.255.255.224 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.32.0.0, so there are 16 network bits.

Step 2. The question states that 500 subnets are needed. A mask with 8 subnet bits supplies only 2^8 (256) subnets, but a mask with 9 subnet bits supplies 2^9 (512) subnets. So, the mask needs at least 9 subnet bits.

Step 3. The question states that 15 hosts are needed per subnet. A mask with 4 host bits supplies only $2^4 - 2$ (14) hosts per subnet, but a mask with 5 host bits supplies $2^5 - 2$ (30) hosts per subnet. So, the mask needs at least 5 host bits.

Step 6A. With N=16, a minimum S=9, and a minimum H=5, multiple masks exist. The first mask, with the minimum number of subnet bits, is /25, found by adding N (16) to the minimum value of S (9). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 5. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 5 = /27.

Step 6C. All masks between /25 and /27 also meet the requirements (/25, /26, /27).

Answer to Mask Design Problem 6

Problem 6 shows a Class B network, with 16 network bits, with a minimum of 11 subnet bits and 2 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.224 (maximizes the number of hosts/subnet)
- 255.255.255.240
- 255.255.255.248
- 255.255.255.252 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.16.0.0, so there are 16 network bits.

Step 2. The question states that 2000 subnets are needed. A mask with 10 subnet bits supplies only 2^{10} (1024) subnets, but a mask with 11 subnet bits supplies 2^{11} (2048) subnets. So, the mask needs at least 11 subnet bits.

Step 3. The question states that 2 hosts are needed per subnet. A mask with 2 host bits supplies $2^2 - 2$ (2) hosts per subnet. So, the mask needs at least 2 host bits.

Step 6A. With N=16, a minimum S=11, and a minimum H=2, multiple masks exist. The first mask, with the minimum number of subnet bits, is /27, found by adding N (16) to the minimum value of S (11). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 2. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 2 = /30.

Step 6C. All masks between /27 and /30 also meet the requirements (/27, /28, /29, /30).

Practice Finding All Subnet IDs

The remainder of this appendix lists two sets of problems. Both problem sets list an IP network and mask; your job is to list all the subnet IDs for each network/mask combination. The first problem set includes problems that happen to have 8 or fewer subnet bits, and the second problem set includes problems that happen to have more than 8 subnet bits. In particular, for each problem, find the following:

- All subnet numbers
- The subnet that is the zero subnet
- The subnet that is the broadcast subnet

To find this information, you can use the processes explained in Appendix L.

Find Subnet IDs, Problem Set 1: 8 or Fewer Subnet Bits

The problems, which consist of a classful network and static-length mask, are as follows:

1. 172.32.0.0/22
2. 200.1.2.0/28

3. 10.0.0.0/15
4. 172.20.0.0/24

Find Subnet IDs, Problem Set 2: More Than 8 Subnet Bits

The problems, which consist of a classful network and static-length mask, are as follows:

1. 172.32.0.0/25
2. 10.0.0.0/21

Answers to Find Subnet IDs, Problem Set 1

This section includes the answers to the four problems listed in Problem Set 1.

Problem Set 1, Answer 1: 172.32.0.0/22

The answer is as follows:

- 172.32.0.0 (zero subnet)
- 172.32.4.0
- 172.32.8.0
- 172.32.12.0
- 172.32.16.0
- 172.32.20.0
- 172.32.24.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 4 to the third octet.)
- 172.32.248.0
- 172.32.252.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has fewer than 8 subnet bits (6 bits), because the network is a Class B network (16 network bits), and the mask has 22 binary 1s in it—implying 10 host bits and leaving 6 subnet bits.
- The mask in dotted-decimal format is 255.255.252.0. The interesting octet is the third octet because the subnet bits are all in the third octet.
- Each successive subnet number is 4 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 252 = 4.

As a result, in this case, all the subnets begin with 172.32, have a multiple of 4 in the third octet, and end in 0.

Table M-2 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-2 8 or Fewer Subnet Bits, Question 1: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	252	0
Magic Number (Step 3)			256 – 252 = 4	
Zero Subnet Number (Step 4)	172	32	0	0
Next Subnet (Step 5)	172	32	4	0
Next Subnet (Step 5)	172	32	8	0
Next Subnet (Step 5)	172	32	12	0
Next Subnet (Step 5)	172	32	16	0
(You might need many more such rows.)	172	32	X	0
Next Subnet	172	32	244	0
Next Subnet (Step 5)	172	32	248	0
Broadcast Subnet (Step 6)	172	32	252	0
Out of Range—Stop Process (Step 6)			256	

Problem Set 1, Answer 2: 200.1.2.0/28

The answer is as follows:

- 200.1.2.0 (zero subnet)
- 200.1.2.16
- 200.1.2.32
- 200.1.2.48
- 200.1.2.64
- 200.1.2.80

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 16 to the fourth octet.)
- 200.1.2.224
- 200.1.2.240 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has fewer than 8 subnet bits (4 bits), because the network is a Class C network (24 network bits), and the mask has 28 binary 1s in it, which implies 4 host bits and leaves 4 subnet bits.
- The mask in dotted-decimal format is 255.255.255.240. The interesting octet is the fourth octet, because all the subnet bits are in the fourth octet.
- Each successive subnet number is 16 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 240 = 16.

As a result, in this case, all the subnets begin with 200.1.2 and have a multiple of 16 in the fourth octet.

Table M-3 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-3 Problem Set 1, Question 2: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	255	240
Magic Number (Step 3)				256 – 240 = 16
Zero Subnet Number (Step 4)	200	1	2	0
Next Subnet (Step 5)	200	1	2	16
Next Subnet (Step 5)	200	1	2	32
Next Subnet (Step 5)	200	1	2	48
(You might need many more such rows.) (Step 5)	200	1	2	X
Next Subnet (Step 5)	200	1	2	224
Broadcast Subnet (Step 6)	200	1	2	240
Out of Range—Stop Process (Step 6)				256

Problem Set 1, Answer 3: 10.0.0.0/15

The answer is as follows:

- 10.0.0.0 (zero subnet)
- 10.2.0.0
- 10.4.0.0
- 10.6.0.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 2 to the second octet.)
- 10.252.0.0
- 10.254.0.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has fewer than 8 subnet bits (7 subnet bits), because the network is a Class A network (8 network bits), and the mask has 15 binary 1s in it, which implies 17 host bits and leaves 7 subnet bits.
- The mask in dotted-decimal format is 255.254.0.0. The interesting octet is the second octet, because all the subnet bits exist in the second octet.
- Each successive subnet number is 2 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 254 = 2.

As a result, in this case, all the subnets begin with 10, have a multiple of 2 in the second octet, and end in 0.0.

Table M-4 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-4 Problem Set 1, Question 3: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	254	0	0
Magic Number (Step 3)		256 – 254 = 2		
Zero Subnet Number (Step 4)	10	0	0	0
Next Subnet (Step 5)	10	2	0	0
Next Subnet (Step 5)	10	4	0	0
Next Subnet (Step 5)	10	6	0	0
(You might need many more such rows.) (Step 5)	10	X	0	0
Next Subnet (Step 5)	10	252	0	0
Broadcast Subnet (Step 6)	10	254	0	0
Out of Range—Stop Process (Step 6)		256		

Problem Set 1, Answer 4: 172.20.0.0/24

This problem has an 8-bit subnet field, meaning that 2^8, or 256, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 172.20.0.0 (zero subnet)
- 172.20.1.0
- 172.20.2.0
- 172.20.3.0
- 172.20.4.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 1 to the third octet.)
- 172.20.252.0
- 172.20.253.0
- 172.20.254.0
- 172.20.255.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has exactly 8 subnet bits, specifically all bits in the third octet, making the third octet the interesting octet.
- The magic number is 256 – 255 = 1, because the mask's value in the interesting (third) octet is 255.
- Beginning with the network number of 172.20.0.0, which is the same value as the zero subnet, just add the magic number (1) in the interesting octet.

Essentially, you just count by 1 in the third octet until you reach the highest legal number (255). The first subnet, 172.20.0.0, is the zero subnet, and the last subnet, 172.20.255.0, is the broadcast subnet.

Answers to Find Subnet IDs, Problem Set 2

Problem Set 2, Answer 1: 172.32.0.0/25

This problem has a 9-bit subnet field, meaning that 2^9, or 512, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 172.32.0.0 (zero subnet)
- 172.32.0.128
- 172.32.1.0
- 172.32.1.128
- 172.32.2.0
- 172.32.2.128
- 172.32.3.0
- 172.32.3.128

  (Skipping many subnets; the subnets occur in blocks of two, with either 0 or 128 in the fourth octet, with each successive block being one greater in the third octet.)
- 172.32.254.0
- 172.32.254.128
- 172.32.255.0
- 172.32.255.128 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has more than 8 subnet bits (9 bits), because the network is a Class B network (16 network bits), and the mask has 25 binary 1s in it, which implies 7 host bits and leaves 9 subnet bits.
- Using the terminology in Appendix L, octet 4 is the *interesting* octet, where the counting occurs based on the magic number. Octet 3 is the "just left" octet, in which the process counts by 1, from 0 to 255.
- The magic number, which will be used to calculate each successive subnet number, is 256 – 128 = 128.

To calculate the first subnet block, use the same six-step process as used in the simpler problems that have 8 or fewer subnet bits. In this case, with only 1 subnet bit in octet 4, only two subnets exist in each subnet block. Table M-5 shows the steps as compared to the six-step process to find the subnets in a subnet block.

Table M-5 Creating the First Subnet Block

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	255	128
Magic Number (Step 3)				256 – 128 = 128
Zero Subnet Number (Step 4)	172	32	0	0
Next Subnet (Step 5)	172	32	0	128
Step 6 Needs to Be Used Here (Sum of 256 in the 4th Octet)	172	32	0	256

The table represents the logic, but to make sure that the answer is clear, the first subnet block includes the following:

172.32.0.0

172.32.0.128

The next major task—to create subnet blocks for all possible values in the "just left" octet—completes the process. Essentially, create 256 blocks like the previous list. The first has a value of 0, in the "just left" octet; the next has a value of 1; the next, a value of 2; and so on, through a block that begins with 172.30.255. Figure M-1 shows the concept.

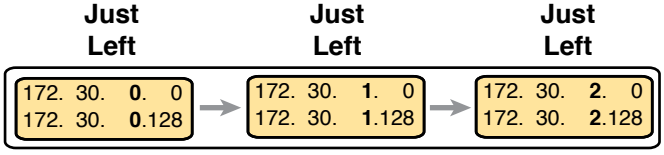


Figure M-1 *Creating Subnet Blocks by Adding 1 in the "Just Left" Octet*

Problem Set 2, Answer 2: 10.0.0.0/21

This problem has a 13-bit subnet field, meaning that 2^{13}, or 8192, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 10.0.0.0 (zero subnet)
- 10.0.8.0
- 10.0.16.0
- 10.0.24.0

  (Skipping several subnets)
- 10.0.248.0
- 10.1.0.0
- 10.1.8.0
- 10.1.16.0

  (Skipping several subnets)
- 10.1.248.0

- 10.2.0.0
- 10.2.8.0
- 10.2.16.0

  (Skipping several subnets)
- 10.255.232.0
- 10.255.240.0
- 10.255.248.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has more than 8 subnet bits (13 bits), because the network is a Class A network (8 network bits), and the mask has 21 binary 1s in it, which implies 11 host bits and leaves 13 subnet bits.
- Using the terminology in Appendix L, octet 3 is the interesting octet, where the counting occurs based on the magic number. Octet 2 is the "just left" octet, in which the process counts by 1, from 0 to 255.
- The magic number, which will be used to calculate each successive subnet number, is 256 – 248 = 8.

To calculate the first subnet block, use the same six-step process as used in the simpler problems that have 8 or fewer subnet bits. In this case, with 5 subnet bits in octet 3, 32 subnets exist in each subnet block. Table M-6 shows the steps as compared to the six-step process to find the subnets in a subnet block.

Table M-6 Creating the First Subnet Block

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	248	0
Magic Number (Step 3)			256 – 248 = 8	
Zero Subnet Number (Step 4)	10	0	0	0
Next Subnet (Step 5)	10	0	8	0
(Skipping several subnets)	10	0	X	0
Next Subnet (Step 5)	10	0	248	0
Step 6 Needs to Be Used Here (Sum of 256 in the 3rd Octet)	10	0	256	0

The table represents the logic, but to make sure that the answer is clear, the first subnet block includes the following:

10.0.0.0
10.0.8.0
10.0.16.0
10.0.24.0
10.0.32.0
10.0.40.0

10.0.48.0

10.0.56.0

10.0.64.0

And so on…

10.0.248.0

The next major task—to create subnet blocks for all possible values in the "just left" octet—completes the process. Essentially, create 256 blocks like the previous list. The first has a value of 0, in the "just left" octet; the next has a value of 1; the next, a value of 2; and so on, through a block that begins with 10.255. Figure M-2 shows the concept.

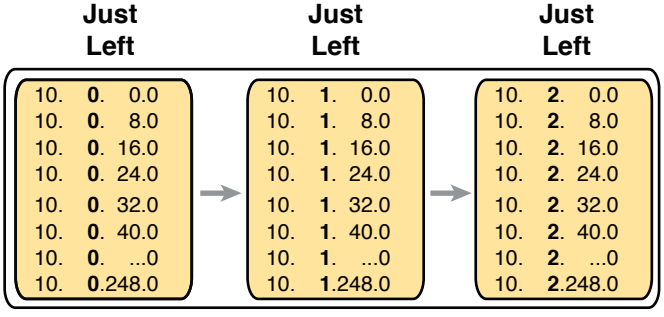


Figure M-2 *Creating Subnet Blocks by Adding 1 in the "Just Left" Octet*

APPENDIX N

Variable-Length Subnet Masks

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 22 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

IPv4 addressing and subnetting use a lot of terms, a lot of small math steps, and a lot of concepts that fit together. While learning those concepts, it helps to keep things as simple as possible. One way this book has kept the discussion simpler so far was to show examples that use one mask only inside a single Class A, B, or C network.

This chapter removes that restriction by introducing variable-length subnet masks (VLSM). VLSM simply means that the subnet design uses more than one mask in the same classful network. VLSM has some advantages and disadvantages, but when learning, the main challenge is that a subnetting design that uses VLSM requires more math, and it requires that you think about some other issues as well. This chapter walks you through the concepts, the issues, and the math.

Foundation Topics

VLSM Concepts and Configuration

VLSM occurs when an internetwork uses more than one mask for different subnets of a single Class A, B, or C network. Figure N-1 shows an example of VLSM used in Class A network 10.0.0.0.

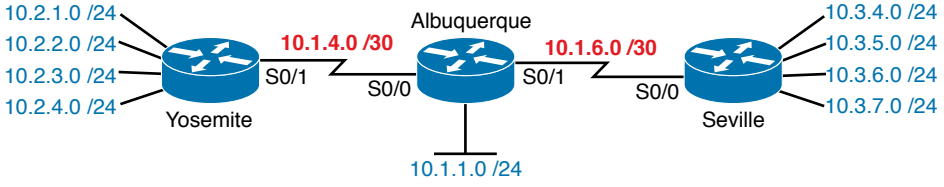


Figure N-1 *VLSM in Network 10.0.0.0: Masks /24 and /30*

Figure N-1 shows a typical choice of using a /30 prefix (mask 255.255.255.252) on point-to-point serial links, with mask /24 (255.255.255.0) on the LAN subnets. All subnets are of Class A network 10.0.0.0, with two masks being used, therefore meeting the definition of VLSM.

Oddly enough, a common mistake occurs when people think that VLSM means "using more than one mask in some internetwork" rather than "using more than one mask *in a single classful network*." For example, if in one internetwork diagram, all subnets of network 10.0.0.0 use a 255.255.240.0 mask, and all subnets of network 11.0.0.0 use a 255.255.255.0 mask, the design uses two different masks. However, Class A network 10.0.0.0 uses only one mask, and Class A network 11.0.0.0 uses only one mask. In that case, the design does not use VLSM.

VLSM provides many benefits for real networks, mainly related to how you allocate and use your IP address space. Because a mask defines the size of the subnet (the number of host addresses in the subnet), VLSM allows engineers to better match the need for addresses with the size of the subnet. For example, for subnets that need fewer addresses, the engineer uses a mask with fewer host bits, so the subnet has fewer host IP addresses. This flexibility reduces the number of wasted IP addresses in each subnet. By wasting fewer addresses, more space remains to allocate more subnets.

VLSM can be helpful for both public and private IP addresses, but the benefits are more dramatic with public networks. With public networks, the address savings help engineers avoid having to obtain another registered IP network number from regional IP address assignment authorities. With private networks, as defined in RFC 1918, running out of addresses is not as big a negative, because you can always grab another private network from RFC 1918 if you run out.

Classless and Classful Routing Protocols

Before you can deploy a VLSM design, you must first use a routing protocol that supports VLSM. To support VLSM, the routing protocol must advertise the mask along with each subnet. Without mask information, the router receiving the update would be confused.

For example, if a router learned a route for 10.1.8.0, but with no mask information, what does that mean? Is that subnet 10.1.8.0/24? 10.1.8.0/23? 10.1.8.0/30? The dotted-decimal number 10.1.8.0 happens to be a valid subnet number with a variety of masks, and because multiple masks can be used with VLSM, the router has no good way to make an educated guess. To effectively support VLSM, the routing protocol needs to advertise the correct mask along with each subnet so that the receiving router knows the exact subnet that is being advertised.

By definition, *classless routing protocols* advertise the mask with each advertised route, and *classful routing protocols* do not. The classless routing protocols, as noted in Table N-1, are the newer, more advanced routing protocols. Not only do these more advanced classless routing protocols support VLSM, but they also support manual route summarization, which allows a routing protocol to advertise one route for a larger subnet instead of multiple routes for smaller subnets.

Key Topic

Table N-1 Classless and Classful Interior IP Routing Protocols

Routing Protocol	Is It Classless?	Sends Mask in Updates?	Supports VLSM?	Supports Manual Route Summarization?
RIPv1	No	No	No	No
RIPv2	Yes	Yes	Yes	Yes
EIGRP	Yes	Yes	Yes	Yes
OSPF	Yes	Yes	Yes	Yes

Beyond VLSM itself, the routing protocols do not have to be configured to support VLSM or to be classless. There is no command to enable or disable the fact that classless routing protocols include the mask with each route. The only configuration choice you must make is to use a classless routing protocol.

VLSM Configuration and Verification

Cisco routers do not configure VLSM, enable or disable it, or need any configuration to use it. From a configuration perspective, VLSM is simply a side effect of using the **ip address** interface subcommand. Routers collectively configure VLSM by virtue of having IP addresses in the same classful network but with different masks.

For example, Example N-1 shows two of the interfaces from router Yosemite from Figure N-1. The example shows the IP address assignments on two interfaces, one with a /24 mask and one with a /30 mask, both with IP addresses in Class A network 10.0.0.0.

Example N-1 *Configuring Two Interfaces on Yosemite, Resulting in VLSM*

```
Yosemite# configure terminal
Yosemite(config)# interface Fa0/0
Yosemite(config-if)# ip address 10.2.1.1 255.255.255.0
Yosemite(config-if)# interface S0/1
Yosemite(config-if)# ip address 10.1.4.1 255.255.255.252
```

The use of VLSM can also be detected by a detailed look at the output of the **show ip route** command. This command lists routes in groups, by classful network, so that you see all the subnets of a single Class A, B, or C network all in a row. Just look down the list, and

look to see, if any, how many different masks are listed. For example, Example N-2 lists the routing table on Albuquerque from Figure N-1; Albuquerque uses masks /24 and /30 inside network 10.0.0.0, as noted in the highlighted line in the example.

Example N-2 *Albuquerque Routing Table with VLSM*

```
Albuquerque# show ip route
! Legend omitted for brevity

     10.0.0.0/8 is variably subnetted, 14 subnets, 3 masks
D       10.2.1.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.2.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.3.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.4.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.3.4.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.5.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.6.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.7.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
C       10.1.1.0/24 is directly connected, FastEthernet0/0
L       10.1.1.1/32 is directly connected, FastEthernet0/0
C       10.1.6.0/30 is directly connected, Serial0/1
L       10.1.6.1/32 is directly connected, Serial0/1
C       10.1.4.0/30 is directly connected, Serial0/0
L       10.1.4.1/32 is directly connected, Serial0/0
```

NOTE For the purposes of understanding whether a design uses VLSM, ignore the /32 "local" routes that a router automatically creates for its own interface IP addresses.

So ends the discussion of VLSM as an end to itself. This chapter is devoted to VLSM, but it took a mere three to four pages to fully describe it. Why the entire VLSM chapter? Well, to work with VLSM, to find problems with it, to add subnets to an existing design, and to design using VLSM from scratch—in other words, to apply VLSM to real networks—takes skill and practice. To do these same tasks on the exam requires skill and practice. The rest of this chapter examines the skills to apply VLSM and provides some practice for these two key areas:

- Finding VLSM overlaps
- Adding new VLSM subnets without overlaps

Finding VLSM Overlaps

Key Topic

Regardless of whether a design uses VLSM, the subnets used in any IP internetwork design should not overlap their address ranges. When subnets in different locations overlap their addresses, a router's routing table entries overlap. As a result, hosts in different locations can be assigned the same IP address. Routers clearly cannot route packets correctly in these cases. In short, a design that uses overlapping subnets is considered to be an incorrect design and should not be used.

This section begins with a short discussion about VLSM design, to drive home the ideas behind VLSM overlaps. It then gets into an operational and troubleshooting approach to the topic, by looking at existing designs and trying to find any existing overlaps.

Designing Subnetting Plans with VLSM

When creating a subnetting plan using VLSM, you have to be much more careful in choosing what subnets to use. First, whatever masks you use in a VLSM design, each subnet ID must be a valid subnet ID given the mask that you use for that subnet.

For example, consider a subnet plan for Class B network 172.16.0.0. To create a subnet with a /24 mask, the subnet ID must be a subnet ID that you could choose if you subnetted the whole Class B network with that same mask. Appendix L, "Subnet Design," discusses how to find those subnets in depth, but with a Class B network and a /24 mask, the possible subnet IDs should be easy to calculate by now: 172.16.0.0 (the zero subnet), then 172.16.1.0, 172.16.2.0, 172.16.3.0, 172.16.4.0, and so on, up through 172.16.255.0.

NOTE Subnet IDs must always follow this important binary rule as noted back in Chapter 14, "Analyzing Existing Subnets": In binary, each subnet ID has a host field of all binary 0s. If you use the math and processes to find all subnet IDs per Appendix L, all those subnet IDs happen to have binary 0s in the host fields.

Now expand your thinking about subnet IDs to a VLSM design. To begin, you would decide that you need some subnets with one mask, other subnets with another mask, and so on, to meet the requirements for different sizes of different subnets. For instance, imagine you start with a brand-new VLSM design, with Class B network 172.16.0.0. You plan to have some subnets with /22 masks, some with /23, and some with /24. You might develop then a planning diagram, or at least draw the ideas, with something like Figure N-2.

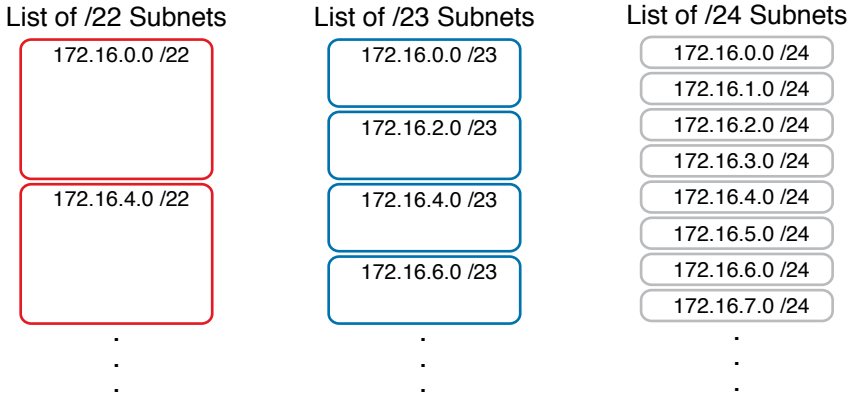


Figure N-2 *Possible Subnet IDs of Network 172.16.0.0, with /22, /23, and /24 Masks*

The drawing shows the first few subnet IDs available with each mask, but you cannot use all subnets from all three lists in a design. As soon as you choose to use one subnet from any column, you remove some subnets from the other lists because subnets cannot overlap. Overlapping subnets are subnets whose range of addresses include some of the same addresses.

As an example, Figure N-3 shows the same list of the first few possible /22, /23, and /24 subnets of Class B network 172.16.0.0. However, it shows a check mark beside two subnets that have been allocated for use; that is, on paper, the person making the subnetting plan has decided to use these two subnets somewhere in the network. The subnets with a dark gray shading and an *X* in them can no longer be used because they have some overlapping addresses with the subnets that have check marks (172.16.3.0/24 and 172.16.4.0/22).

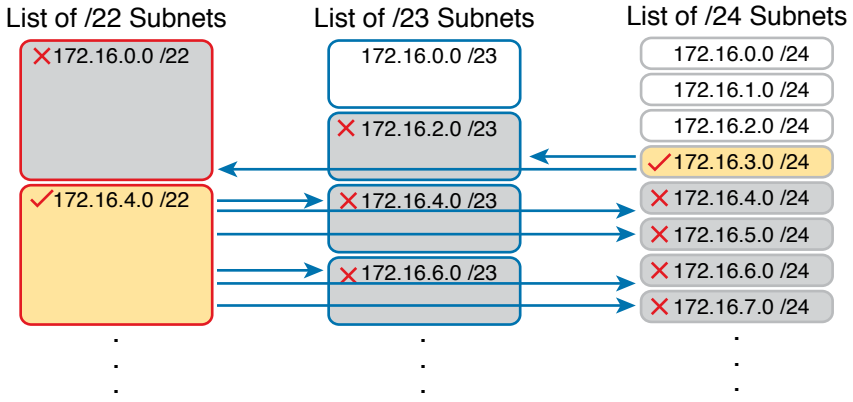


Figure N-3 *Selecting Two Subnets Disallows Other Subnets in Different Columns*

Just to complete the example, first look at subnet 172.16.4.0 on the lower left. That subnet includes addresses from the subnet ID of 172.16.4.0 through the subnet broadcast address of 172.16.7.255. As you can see just by looking at the subnet IDs to the right, all the subnets referenced with the arrowed lines are within that same range of addresses.

Now look to the upper right of the figure, to subnet 172.16.3.0/24. The subnet has a range of 172.16.3.0–172.16.3.255 including the subnet ID and subnet broadcast address. That subnet overlaps with the two subnets referenced to the left. For instance, subnet 172.16.0.0/22 includes the range from 172.16.0.0–172.16.3.255. But because there is some overlap, once the design has allocated the 172.16.3.0/24 subnet, the 172.16.2.0/23 and 172.16.0.0/22 subnets could not be used without causing problems, because:

A subnetting design, whether using VLSM or not, should not allow subnets whose address ranges overlap. If overlapping subnets are implemented, routing problems occur and some hosts simply cannot communicate outside their subnets.

These address overlaps are easier to see when not using VLSM. When not using VLSM, overlapped subnets have identical subnet IDs, so to find overlaps, you just have to look at the subnet IDs. With VLSM, overlapped subnets may not have the same subnet ID, as was the case in this most recent example with the subnets across the top of Figure N-3. To find these overlaps, you have to look at the entire range of addresses in each subnet, from subnet ID to subnet broadcast address, and compare the range to the other subnets in the design.

An Example of Finding a VLSM Overlap

For example, imagine that a practice question for the CCENT exam shows Figure N-4. It uses a single Class B network (172.16.0.0), with VLSM, because it uses three different masks: /23, /24, and /30.

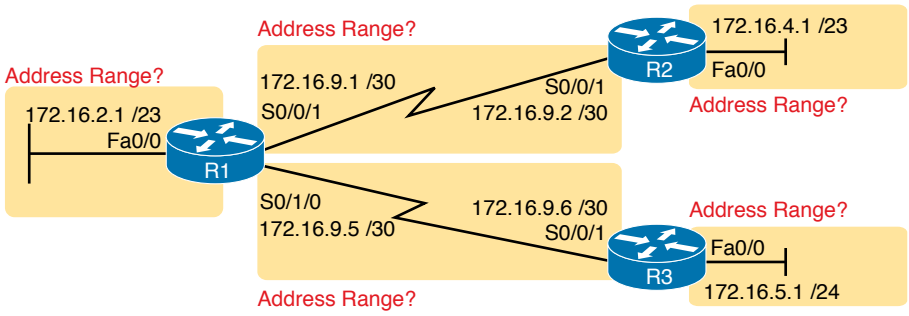


Figure N-4 *VLSM Design with Possible Overlap*

Now imagine that the exam question shows you the figure, and either directly or indirectly asks whether overlapping subnets exist. This type of question might simply tell you that some hosts cannot ping each other, or it might not even mention that the root cause could be that some of the subnets overlap. To answer such a question, you could follow this simple but possibly laborious process:

N

Key Topic

Step 1. Calculate the subnet ID and subnet broadcast address of each subnet, which gives you the range of addresses in that subnet.

Step 2. List the subnet IDs in numerical order (along with their subnet broadcast addresses).

Step 3. Scan the list from top to bottom, comparing each pair of adjacent entries, to see whether their range of addresses overlaps.

For example, Table N-2 completes the first two steps based on Figure N-4, listing the subnet IDs and subnet broadcast addresses, in numerical order based on the subnet IDs.

Table N-2 Subnet IDs and Broadcast Addresses, in Numerical Order, from Figure N-4

Subnet	Subnet Number	Broadcast Address
R1 LAN	172.16.2.0	172.16.3.255
R2 LAN	172.16.4.0	172.16.5.255
R3 LAN	172.16.5.0	172.16.5.255
R1-R2 serial	172.16.9.0	172.16.9.3
R1-R3 serial	172.16.9.4	172.16.9.7

The VLSM design is invalid in this case because of the overlap between R2's LAN subnet and R3's LAN subnet. As for the process, Step 3 states the somewhat obvious step of comparing the address ranges to see whether any overlaps occur. Note that, in this case, none of the subnet numbers are identical, but two entries (highlighted) do overlap. The design is invalid because of the overlap, and one of these two subnets would need to be changed.

As far as the three-step process works, note that if two adjacent entries in the list overlap, compare three entries at the next step. The two subnets already marked as overlapped can overlap with the next subnet in the list. For example, the three subnets in the following list overlap in that the first subnet overlaps with the second and third subnets in the list. If you

followed the process shown here, you would have first noticed the overlap between the first two subnets in the list, so you would then also need to check the next subnet in the list to find out if it overlapped.

10.1.0.0/16 (subnet ID 10.1.0.0, broadcast 10.1.255.255)

10.1.200.0/24 (subnet ID 10.1.200.0, broadcast 10.1.200.255)

10.1.250.0/24 (subnet ID 10.1.250.0, broadcast 10.1.250.255)

Practice Finding VLSM Overlaps

As typical of anything to with applying IP addressing and subnetting, practice helps. To that end, Table N-3 lists three practice problems. Just start with the five IP addresses listed in a single column, and then follow the three-step process outlined in the previous section to find any VLSM overlaps. The answers can be found near the end of this chapter, in the section "Answers to Earlier Practice Problems."

Table N-3 VLSM Overlap Practice Problems

Problem 1	Problem 2	Problem 3
10.1.34.9/22	172.16.126.151/22	192.168.1.253/30
10.1.29.101/23	172.16.122.57/27	192.168.1.113/28
10.1.23.254/22	172.16.122.33/30	192.168.1.245/29
10.1.17.1/21	172.16.122.1/30	192.168.1.125/30
10.1.1.1/20	172.16.128.151/20	192.168.1.122/30

Adding a New Subnet to an Existing VLSM Design

The task described in this section happens frequently in real networks: choosing new subnets to add to an existing design. In real life, you can use IP Address Management (IPAM) tools that help you choose a new subnet so that you do not cause an overlap. However, for the CCNA exam, you need to be ready to do the mental process and math of choosing a subnet that does not create an overlapped VLSM subnet condition. In other words, you need to pick a new subnet and not make a mistake!

For example, consider the internetwork shown earlier in Figure N-2, with classful network 172.16.0.0. An exam question might suggest that a new subnet, with a /23 prefix length, needs to be added to the design. The question might also say, "Pick the numerically lowest subnet number that can be used for the new subnet." In other words, if both 172.16.4.0 and 172.16.6.0 would work, use 172.16.4.0.

So, you really have a couple of tasks: To find all the subnet IDs that could be used, rule out the ones that would cause an overlap, and then check to see whether the question guides you to pick either the numerically lowest (or highest) subnet ID. This list outlines the specific steps:

Key Topic

Step 1. Pick the subnet mask (prefix length) for the new subnet, based on the design requirements (if not already listed as part of the question).

Step 2. Calculate all possible subnet numbers of the classful network using the mask from Step 1, along with the subnet broadcast addresses.

Step 3. Make a list of existing subnet IDs and matching subnet broadcast addresses.

Step 4. Compare the existing subnets to the candidate new subnets to rule out overlapping new subnets.

Step 5. Choose the new subnet ID from the remaining subnets identified at Step 4, paying attention to whether the question asks for the numerically lowest or numerically highest subnet ID.

An Example of Adding a New VLSM Subnet

For example, Figure N-5 shows an existing internetwork that uses VLSM. (The figure uses the same IP addresses as shown in Figure N-4, but with R3's LAN IP address changed to fix the VLSM overlap shown in Figure N-4.) In this case, you need to add a new subnet to support 300 hosts. Imagine that the question tells you to use the smallest subnet (least number of hosts) to meet that requirement. You use some math and logic you learned earlier in your study to choose mask /23, which gives you 9 host bits, for $2^9 - 2 = 510$ hosts in the subnet.

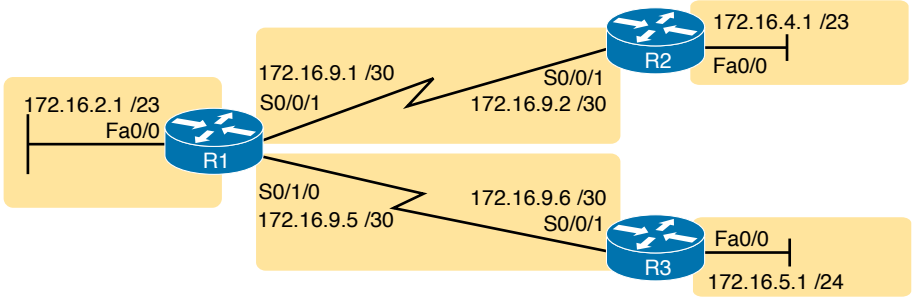


Figure N-5 *Internetwork to Which You Need to Add a /23 Subnet, Network 172.16.0.0*

At this point, just follow the steps listed before Figure N-5. For Step 1, you have already been given the mask (/23). For Step 2, you need to list all the subnet numbers and broadcast addresses of 172.16.0.0, assuming the /23 mask. You will not use all these subnets, but you need the list for comparison to the existing subnets. Table N-4 shows the results, at least for the first five possible /23 subnets.

Table N-4 First Five Possible /23 Subnets

Subnet	Subnet Number	Subnet Broadcast Address
First (zero)	172.16.0.0	172.16.1.255
Second	172.16.2.0	172.16.3.255
Third	172.16.4.0	172.16.5.255
Fourth	172.16.6.0	172.16.7.255
Fifth	172.16.8.0	172.16.9.255

Next, at Step 3, list the existing subnet numbers and broadcast addresses, as shown earlier in Figure N-5. To do so, do the usual math to take an IP address/mask to then find the subnet ID and subnet broadcast address. Table N-5 summarizes that information, including the locations, subnet numbers, and subnet broadcast addresses.

Table N-5 Existing Subnet IDs and Broadcast Addresses from Figure N-5

Subnet	Subnet Number	Subnet Broadcast Address
R1 LAN	172.16.2.0	172.16.3.255
R2 LAN	172.16.4.0	172.16.5.255
R3 LAN	172.16.6.0	172.16.6.255
R1-R2 serial	172.16.9.0	172.16.9.3
R1-R3 serial	172.16.9.4	172.16.9.7

At this point, you have all the information you need to look for the overlap at Step 4. Simply compare the range of numbers for the subnets in the previous two tables. Which of the possible new /23 subnets (Table N-4) overlap with the existing subnets (Table N-5)? In this case, the second through fifth subnets in Table N-4 overlap, so rule those out as candidates to be used. (Table N-4 denotes those subnets with gray highlights.)

Step 5 has more to do with the exam than with real network design, but it is still worth listing as a separate step. Multiple-choice questions sometimes need to force you into a single answer, and asking for the numerically lowest or highest subnet does that. This particular example asks for the numerically lowest subnet number, which in this case is 172.16.0.0/23.

NOTE The answer, 172.16.0.0/23, happens to be a zero subnet. For the exam, the zero subnet should be avoided if (a) the question implies the use of classful routing protocols or (b) the routers are configured with the **no ip subnet-zero** global configuration command. Otherwise, assume that the zero subnet can be used.

Answers to Earlier Practice Problems

Answers to Practice Finding VLSM Overlaps

This section lists the answers to the three practice problems in the section "Practice Finding VLSM Overlaps," as listed earlier in Table N-3. Note that the tables that list details of the answer reordered the subnets as part of the process.

In Problem 1, the second and third subnet IDs listed in Table N-6 happen to overlap. The second subnet's range completely includes the range of addresses in the third subnet.

Table N-6 VLSM Overlap Problem 1 Answers (Overlaps Highlighted)

Reference	Original Address and Mask	Subnet ID	Broadcast Address
1	10.1.1.1/20	10.1.0.0	10.1.15.255
2	10.1.17.1/21	10.1.16.0	10.1.23.255
3	10.1.23.254/22	10.1.20.0	10.1.23.255
4	10.1.29.101/23	10.1.28.0	10.1.29.255
5	10.1.34.9/22	10.1.32.0	10.1.35.255

In Problem 2, again the second and third subnet IDs (listed in Table N-7) happen to overlap, and again, the second subnet's range completely includes the range of addresses in the third subnet. Also, the second and third subnet IDs are the same value, so the overlap is more obvious.

Table N-7 VLSM Overlap Problem 2 Answers (Overlaps Highlighted)

Reference	Original Address and Mask	Subnet ID	Broadcast Address
1	172.16.122.1/30	172.16.122.0	172.16.122.3
2	172.16.122.57/27	172.16.122.32	172.16.122.63
3	172.16.122.33/30	172.16.122.32	172.16.122.35
4	172.16.126.151/22	172.16.124.0	172.16.127.255
5	172.16.128.151/20	172.16.128.0	172.16.143.255

In Problem 3, three subnets overlap. Subnet 1's range completely includes the range of addresses in the second and third subnets, as shown in Table N-8. Note that the second and third subnets do not overlap with each other, so for the process in this book to find all the overlaps, after you find that the first two subnets overlap, you should compare the next entry in the table (3) with both of the two known-to-overlap entries (1 and 2).

Table N-8 VLSM Overlap Problem 3 Answers (Overlaps Highlighted)

Reference	Original Address and Mask	Subnet ID	Broadcast Address
1	192.168.1.113/28	192.168.1.112	192.168.1.127
2	192.168.1.122/30	192.168.1.120	192.168.1.123
3	192.168.1.125/30	192.168.1.124	192.168.1.127
4	192.168.1.245/29	192.168.1.240	192.168.1.247
5	192.168.1.253/30	192.168.1.252	192.168.1.255

This page intentionally left blank

APPENDIX O

Spanning Tree Protocol Implementation

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 3 of the book *CCNA ICND2 200-105 Official Cert Guide*, published in 2016.

Cisco IOS–based LAN switches enable Spanning Tree Protocol (STP) by default on all interfaces in every VLAN. However, network engineers who work with medium-size to large-size Ethernet LANs usually want to configure at least some STP settings. First and foremost, Cisco IOS switches traditionally default to use STP rather than Rapid STP (RSTP), and the simple upgrade to RSTP improves convergence. For most LANs with more than a few switches, the network engineer will likely want to influence the choices made by STP, whether using traditional STP or RSTP—choices such as which switch becomes root, with predictability about which switch ports will block/discard when all ports are physically working. The configuration can also be set so that when links or switches fail, the engineer can predict the STP topology in those cases, as well.

This chapter discusses configuration and verification of STP. The first major section weaves a story of how to change different settings, per VLAN, with the **show** commands that reveal the current STP status affected by each configuration command. Those settings impact both STP and RSTP, but the examples use switches that use traditional 802.1D STP rather than RSTP. The second major section shows how to configure the optional STP features PortFast, BPDU Guard, and EtherChannel (specifically Layer 2 EtherChannel). The final major section of this chapter looks at the simple (one command) configuration to enable RSTP, and the differences and similarities in **show** command output that occur when using RSTP versus STP.

Foundation Topics

Implementing STP

Cisco IOS switches usually use STP (IEEE 802.1D) by default rather than RSTP, and with effective default settings. You can buy some Cisco switches and connect them with Ethernet cables in a redundant topology, and STP will ensure that frames do not loop. And you never even have to think about changing any settings!

Although STP works without any configuration, most medium-size to large-size campus LANs benefit from some STP configuration. With all defaults, the switches choose the root based on the lowest burned-in MAC address on the switches because they all default to use the same STP priority. As a better option, configure the switches so that the root is predictable.

For instance, Figure O-1 shows a typical LAN design model, with two distribution layer switches (D1 and D2). The design may have dozens of access layer switches that connect to end users; the figure shows just three access switches (A1, A2, and A3). For a variety of reasons, most network engineers make the distribution layer switches be the root. For instance, the configuration could make D1 be the root by having a lower priority, with D2 configured with the next lower priority, so it becomes root if D1 fails.

Key Topic

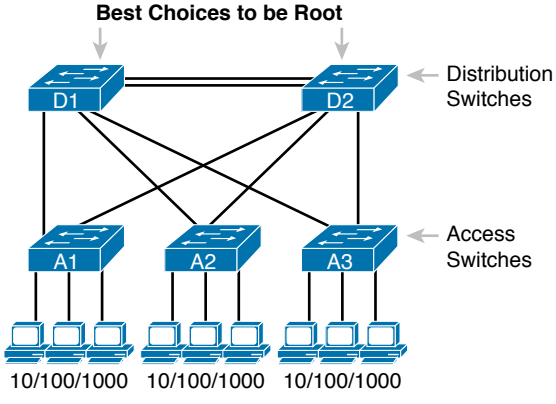


Figure O-1 *Typical Configuration Choice: Making Distribution Switch Be Root*

This first section of the chapter examines a variety of topics that somehow relate to STP configuration. It begins with a look at STP configuration options, as a way to link the concepts of Chapter 2 to the configuration choices in this chapter. Following that, this section introduces some **show** commands for the purpose of verifying the default STP settings before changing any configuration.

Setting the STP Mode

The IEEE first standardized STP as the IEEE 802.1D standard, first published back in 1990. To put some perspective on that date, Cisco sold no LAN switches at the time, and virtual LANs did not exist yet. Instead of multiple VLANs in a LAN, there was just one broadcast domain, and one instance of STP. However, the addition of VLANs and the introduction of LAN switches into the market have created a need to add to and extend STP.

Today, Cisco IOS–based LAN switches allow you to use one of three STP configuration modes that reflect that history. The first two sections of this chapter use the mode called Per-VLAN Spanning Tree Plus (PVST+, or sometimes PVSTP), a Cisco-proprietary improvement of 802.1D STP. The *per-VLAN* part of the name gives away the main feature: PVST+ creates a different STP topology per VLAN, whereas 802.1D actually did not. PVST+ also introduced PortFast. Cisco switches often use PVST+ as the default STP mode per a default global command of **spanning-tree mode pvst**.

Over time, Cisco added RSTP support as well, with two STP modes that happen to use RSTP. One mode basically takes PVST+ and upgrades it to use RSTP logic as well, with a mode called *Rapid PVST+*, enabled with the global command **spanning-tree mode rapid-pvst**. Cisco IOS–based switches support a third mode, called Multiple Spanning Tree (MST) (or Multiple Instance of Spanning Tree), enabled with the **spanning-tree mode mst** command.

Connecting STP Concepts to STP Configuration Options

STP uses two types of numbers for most of its decisions: the BID and STP port costs. Focusing on those two types of numbers, consider this summary of what STP does behind the scenes:

- Uses the BID to elect the root switch, electing the switch with the numerically lowest BID
- Uses the total STP cost in each path to the root, when each nonroot switch chooses its own root port (RP)
- Uses each switch's root cost, which is in turn based on STP port costs, when switches decide which switch port becomes the designated port (DP) on each LAN segment

Unsurprisingly, Cisco switches let you configure part of a switch's BID and the STP port cost, which in turn influences the choices each switch makes with STP.

Per-VLAN Configuration Settings

Beyond supporting the configuration of the BID and STP port costs, Cisco switches support configuring both settings per VLAN. By default, Cisco switches use IEEE 802.1D, not RSTP (802.1w), with a Cisco-proprietary feature called Per-VLAN Spanning Tree Plus (PVST+). PVST+ (often abbreviated as simply PVST today) creates a different instance of STP for each VLAN. So, before looking at the tunable STP parameters, you need to have a basic understanding of PVST+, because the configuration settings can differ for each instance of STP.

PVST+ gives engineers a load-balancing tool with STP. By changing some STP configuration parameters differently for different VLANs, the engineer could cause switches to pick different RPs and DPs in different VLANs. As a result, some traffic in some VLANs can be forwarded over one trunk, and traffic for other VLANs can be forwarded over a different trunk.

Figure O-2 shows the basic idea, with SW3 forwarding odd-numbered VLAN traffic over the left trunk (Gi0/1) and even-numbered VLANs over the right trunk (Gi0/2).

Key Topic

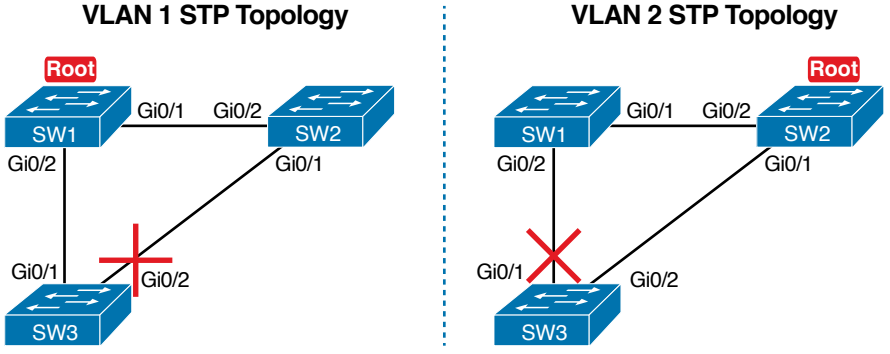


Figure O-2 *Load Balancing with PVST+*

The next few pages look specifically at how to change the BID and STP port cost settings, per VLAN, when using the default PVST+ mode.

The Bridge ID and System ID Extension

Originally, a switch's BID was formed by combining the switch's 2-byte priority and its 6-byte MAC address. Later, the IEEE changed the rules, splitting the original priority field into two separate fields, as shown in Figure O-3: a 4-bit priority field and a 12-bit subfield called the *system ID extension* (which represents the VLAN ID).

Key Topic

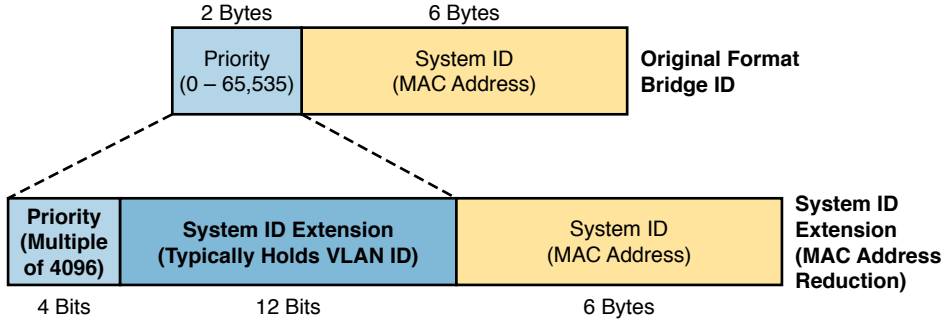


Figure O-3 *STP System ID Extension*

Cisco switches let you configure the BID, but only the priority part. The switch fills in its universal (burned-in) MAC address as the system ID. It also plugs in the VLAN ID of a VLAN in the 12-bit system ID extension field. The only part configurable by the network engineer is the 4-bit priority field.

Configuring the number to put in the priority field, however, is one of the strangest things to configure on a Cisco router or switch. As shown at the top of Figure O-3, the priority field was originally a 16-bit number, which represented a decimal number from 0 to 65,535. Because of that history, the current configuration command (**spanning-tree vlan** *vlan-id* **priority** *x*) requires a decimal number between 0 and 65,535. But not just any number in that range will suffice—it must be a multiple of 4096: 0, 4096, 8192, 12288, and so on, up through 61,440.

The switch still sets the first 4 bits of the BID based on the configured value. As it turns out, of the 16 allowed multiples of 4096, from 0 through 61,440, each has a different binary value in their first 4 bits: 0000, 0001, 0010, and so on, up through 1111. The switch sets the true 4-bit priority based on the first 4 bits of the configured value.

Although the history and configuration might make the BID priority idea seem a bit convoluted, having an extra 12-bit field in the BID works well in practice because it can be used to identify the VLAN ID. VLAN IDs range from 1 to 4094, requiring 12 bits. Cisco switches place the VLAN ID into the system ID extension field, so each switch has a unique BID per VLAN.

For example, a switch configured with VLANs 1 through 4, with a default base priority of 32,768, has a default STP priority of 32,769 in VLAN 1, 32,770 in VLAN 2, 32,771 in VLAN 3, and so on. So, you can view the 16-bit priority as a base priority (as configured in the **spanning-tree vlan** *vlan-id* **priority** *x* command) plus the VLAN ID.

NOTE Cisco switches must use the system ID extension version of the bridge ID; it cannot be disabled.

Per-VLAN Port Costs

Each switch interface defaults its per-VLAN STP cost based on IEEE recommendations. On interfaces that support multiple speeds, Cisco switches base the cost on the current actual speed. So, if an interface negotiates to use a lower speed, the default STP cost reflects that lower speed. If the interface negotiates to use a different speed, the switch dynamically changes the STP port cost as well.

Alternatively, you can configure a switch's STP port cost with the **spanning-tree** [**vlan** *vlan-id*] **cost** *cost* interface subcommand. You see this command most often on trunks because setting the cost on trunks has an impact on the switch's root cost, whereas setting STP costs on access ports does not.

For the command itself, it can include the VLAN ID, or not. The command only needs a **vlan** parameter on trunk ports to set the cost per VLAN. On a trunk, if the command omits the VLAN parameter, it sets the STP cost for all VLANs whose cost is not set by a **spanning-tree vlan** *x* **cost** command for that VLAN.

STP Configuration Option Summary

Table O-1 summarizes the default settings for both the BID and the port costs and lists the optional configuration commands covered in this chapter.

Key Topic

Table O-1 STP Defaults and Configuration Options

Setting	Default	Command(s) to Change Default
BID priority	Base: 32,768	**spanning-tree vlan** *vlan-id* **root** {**primary** | **secondary**} **spanning-tree vlan** *vlan-id* **priority** *priority*
Interface cost	100 for 10 Mbps 19 for 100 Mbps 4 for 1 Gbps 2 for 10 Gbps	**spanning-tree vlan** *vlan-id* **cost** *cost*
PortFast	Not enabled	**spanning-tree portfast**
BPDU Guard	Not enabled	**spanning-tree bpduguard enable**

Next, the configuration section shows how to examine the operation of STP in a simple network, along with how to change these optional settings.

Verifying STP Operation

Before taking a look at how to change the configuration, first consider a few STP verification commands. Looking at these commands first will help reinforce the default STP settings. In particular, the examples in this section use the network shown in Figure O-4.

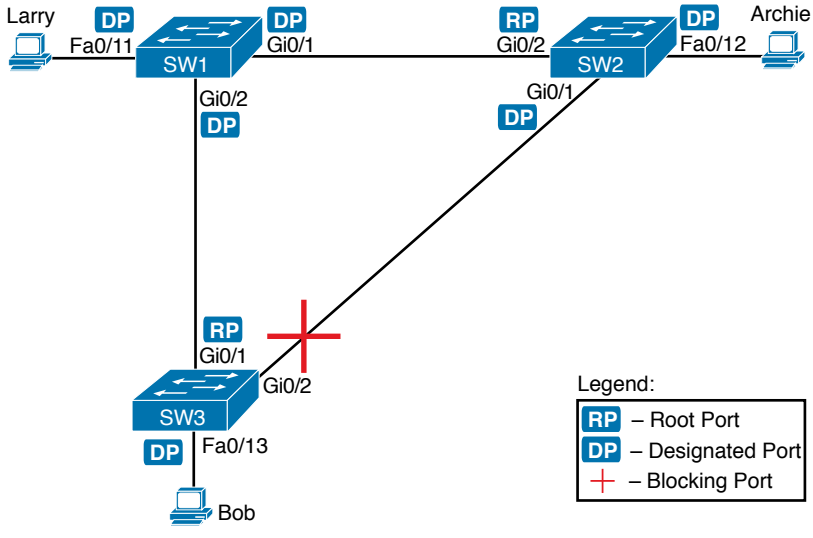


Figure O-4 *Sample LAN for STP Configuration and Verification Examples*

Example O-1 begins the discussion with a useful command for STP: the **show spanning-tree vlan 10** command. This command identifies the root switch and lists settings on the local switch. Example O-1 lists the output of this command on both SW1 and SW2, as explained following the example.

Example O-1 *STP Status with Default STP Parameters on SW1 and SW2*

```
SW1# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     1833.9d7b.0e80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/11              Desg FWD 19        128.11   P2p Edge
Gi0/1               Desg FWD 4         128.25   P2p
Gi0/2               Desg FWD 4         128.26   P2p
```

```
SW2# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
              Cost        4
              Port        26 (GigabitEthernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     1833.9d7b.1380
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/12              Desg FWD 19        128.12   P2p
Gi0/1               Desg FWD 4         128.25   P2p
Gi0/2               Root FWD 4         128.26   P2p
```

Example O-1 begins with the output of the **show spanning-tree vlan 10** command on SW1. This command first lists three major groups of messages: one group of messages about the root switch, followed by another group about the local switch, and ending with interface role and status information. In this case, SW1 lists its own BID as the root, with

even a specific statement that "This bridge is the root," confirming that SW1 is now the root of the VLAN 10 STP topology.

Next, compare the highlighted lines of the same command on SW2 in the lower half of the example. SW2 lists SW1's BID details as the root; in other words, SW2 agrees that SW1 has won the root election. SW2 does not list the phrase "This bridge is the root." SW2 then lists its own (different) BID details in the lines after the details about the root's BID.

The output also confirms a few default values. First, each switch lists the priority part of the BID as a separate number: 32778. This value comes from the default priority of 32768, plus VLAN 10, for a total of 32778. The output also shows the interface cost for some Fast Ethernet and Gigabit Ethernet interfaces, defaulting to 19 and 4, respectively.

Finally, the bottom of the output from the **show spanning-tree** command lists each interface in the VLAN, including trunks, with the STP port role and port state listed. For instance, on switch SW1, the output lists three interfaces, with a role of Desg for designated port (DP) and a state of FWD for forwarding. SW2 lists three interfaces, two DPs, and one root port, so all three are in an FWD or forwarding state.

Example O-1 shows a lot of good STP information, but two other commands, shown in Example O-2, work better for listing BID information in a shorter form. The first, **show spanning-tree root**, lists the root's BID for each VLAN. This command also lists other details, like the local switch's root cost and root port. The other command, **show spanning-tree vlan 10 bridge**, breaks out the BID into its component parts. In this example, it shows SW2's priority as the default of 32768, the VLAN ID of 10, and the MAC address.

Example O-2 *Listing Root Switch and Local Switch BIDs on Switch SW2*

```
SW2# show spanning-tree root

                                        Root    Hello Max Fwd
Vlan                Root ID             Cost    Time  Age Dly  Root Port
---------------- ------------------- --------- ----- --- ---  ------------
VLAN0001         32769 1833.9d5d.c900        23    2   20  15  Gi0/1
VLAN0010         32778 1833.9d7b.0e80         4    2   20  15  Gi0/2
VLAN0020         32788 1833.9d7b.0e80         4    2   20  15  Gi0/2
VLAN0030         32798 1833.9d7b.0e80         4    2   20  15  Gi0/2
VLAN0040         32808 1833.9d7b.0e80         4    2   20  15  Gi0/2

SW2# show spanning-tree vlan 10 bridge

                                             Hello  Max  Fwd
Vlan                     Bridge ID            Time  Age  Dly  Protocol
---------------- --------------------------------- -----  ---  ---  --------
VLAN0010         32778 (32768,  10) 1833.9d7b.1380    2    20   15  ieee
```

Note that both the commands in Example O-2 have a VLAN option: **show spanning-tree** [**vlan** *x*] **root** and **show spanning-tree** [**vlan** *x*] **bridge**. Without the VLAN listed, each command lists one line per VLAN; with the VLAN, the output lists the same information, but just for that one VLAN.

Configuring STP Port Costs

Changing the STP port costs requires a simple interface subcommand: **spanning-tree [vlan** *x*] **cost** *x*. To show how it works, consider the following example, which changes what happens in the network shown in Figure O-4.

Back in Figure O-4, with default settings, SW1 became root, and SW3 blocked on its G0/2 interface. A brief scan of the figure, based on the default STP cost of 4 for Gigabit interfaces, shows that SW3 should have found a cost 4 path and a cost 8 path to reach the root, as shown in Figure O-5.

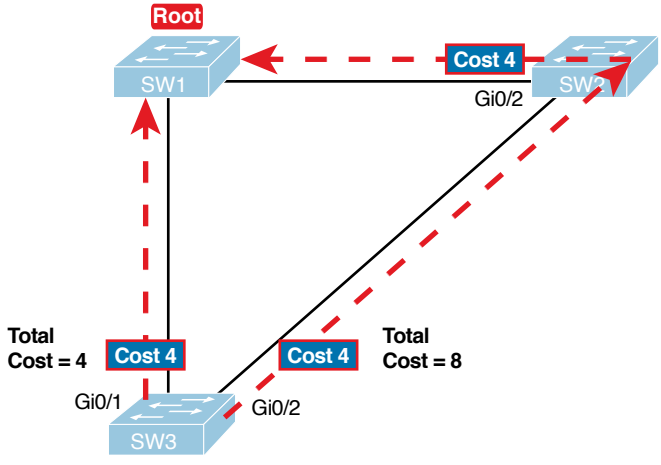


Figure O-5 *Analysis of SW3's Current Root Cost of 4 with Defaults*

To show the effects of changing the port cost, the next example shows a change to SW3's configuration, setting its G0/1 port cost higher so that the better path to the root goes out SW3's G0/2 port instead. Example O-3 also shows several other interesting effects.

Example O-3 *Manipulating STP Port Cost and Watching the Transition to Forwarding State*

```
SW3# debug spanning-tree events
Spanning Tree event debugging is on
SW3# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW3(config)# interface gigabitethernet0/1
SW3(config-if)# spanning-tree vlan 10 cost 30
SW3(config-if)# ^Z
SW3#
*Mar 11 06:28:00.860: STP: VLAN0010 new root port Gi0/2, cost 8
*Mar 11 06:28:00.860: STP: VLAN0010 Gi0/2 -> listening
*Mar 11 06:28:00.860: STP: VLAN0010 sent Topology Change Notice on Gi0/2
*Mar 11 06:28:00.860: STP[10]: Generating TC trap for port GigabitEthernet0/1
*Mar 11 06:28:00.860: STP: VLAN0010 Gi0/1 -> blocking
*Mar 11 06:28:15.867: STP: VLAN0010 Gi0/2 -> learning
*Mar 11 06:28:30.874: STP[10]: Generating TC trap for port GigabitEthernet0/2
*Mar 11 06:28:30.874: STP: VLAN0010 sent Topology Change Notice on Gi0/2
*Mar 11 06:28:30.874: STP: VLAN0010 Gi0/2 -> forwarding
```

This example starts with the **debug spanning-tree events** command on SW3. This command tells the switch to issue debug log messages whenever STP performs changes to an interface's role or state. These messages show up in the example as a result of the configuration.

Next, the example shows the configuration to change SW3's port cost, in VLAN 10, to 30, with the **spanning-tree vlan 10 cost 30** interface subcommand. Based on the figure, the root cost through SW3's G0/1 will now be 30 instead of 4. As a result, SW3's best cost to reach the root is cost 8, with SW3's G0/2 as its root port.

The debug messages tell us what STP on SW3 is thinking behind the scenes, with timestamps. Note that the first five debug messages, displayed immediately after the user exited configuration mode in this case, all happen at the same time (down to the same millisecond). Notably, G0/1, which had been forwarding, immediately moves to a blocking state. Interface G0/2, which had been blocking, does not go to a forwarding state, instead moving to a listening state (at least, according to this message).

Now look for the debug message that lists G0/2 transitioning to learning state, and then the next one that shows it finally reaching forwarding state. How long between the messages? In each case, the message's timestamps show that 15 seconds passed. In this experiment, the switches used a default setting of forward delay (15 seconds). So, these debug messages confirm the steps that STP takes to transition an interface from blocking to forwarding state.

If you did not happen to enable a debug when configuring the cost, using **show** commands later can confirm the same choice by SW3, to now use its G0/2 port as its RP. Example O-4 shows the new STP port cost setting on SW3, along with the new root port and root cost, using the **show spanning-tree vlan 10** command. Note that G0/2 is now listed as the root port. The top of the output lists SW3's root cost as 8, matching the analysis shown in Figure O-5.

Example O-4 *New STP Status and Settings on SW3*

```
SW3# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
              Cost        8
              Port        26 (GigabitEthernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     f47f.35cb.d780
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/23              Desg FWD 19        128.23   P2p
Gi0/1               Altn BLK 30        128.25   P2p
Gi0/2               Root FWD 4         128.26   P2p
```

Configuring Priority to Influence the Root Election

The other big STP configuration option is to influence the root election by changing the priority of a switch. The priority can be set explicitly with the **spanning-tree vlan** *vlan-id* **priority** *value* global configuration command, which sets the base priority of the switch. (This is the command that requires a parameter of a multiple of 4096.)

However, Cisco gives us a better configuration option than configuring a specific priority value. In most designs, the network engineers pick two switches to be root: one to be root if all switches are up, and another to take over if the first switch fails. Switch IOS supports this idea with the **spanning-tree vlan** *vlan-id* **root primary** and **spanning-tree vlan** *vlan-id* **root secondary** commands.

The **spanning-tree vlan** *vlan-id* **root primary** command tells the switch to set its priority low enough to become root right now. The switch looks at the current root in that VLAN, and at the root's priority. Then the local switch chooses a priority value that causes the local switch to take over as root.

Remembering that Cisco switches use a default base priority of 32,768, this command chooses the base priority as follows:

Key Topic

- If the current root has a base priority higher than 24,576, the local switch uses a base priority of 24,576.
- If the current root's base priority is 24,576 or lower, the local switch sets its base priority to the highest multiple of 4096 that still results in the local switch becoming root.

For the switch intended to take over as the root if the first switch fails, use the **spanning-tree vlan** *vlan-id* **root secondary** command. This command is much like the **spanning-tree vlan** *vlan-id* **root primary** command, but with a priority value worse than the primary switch but better than all the other switches. This command sets the switch's base priority to 28,672 regardless of the current root's current priority value.

For example, in Figures O-4 and O-5, SW1 was the root switch, and as shown in various commands, all three switches defaulted to use a base priority of 32,768. Example O-5 shows a configuration that makes SW2 the primary root, and SW1 the secondary, just to show the role move from one to the other. These commands result in SW2 having a base priority of 24,576, and SW1 having a base priority of 28,672.

Example O-5 *Making SW2 Become Root Primary, and SW1 Root Secondary*

```
! First, on SW2:
SW2# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW2(config)# spanning-tree vlan 10 root primary
SW2(config)# ^Z
```

```
! Next, SW1 is configured to back-up SW1
SW1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW1(config)# spanning-tree vlan 10 root secondary
SW1(config)# ^Z
SW1#
```

```
! The next command shows the local switch's BID (SW1)
SW1# show spanning-tree vlan 10 bridge

                                                Hello  Max  Fwd
Vlan                        Bridge ID            Time  Age  Dly  Protocol
---------------- --------------------------------- -----  ---  ---  --------
VLAN0010         28682 (28672,  10) 1833.9d7b.0e80    2    20   15  ieee

! The next command shows the root's BID (SW2)
SW1# show spanning-tree vlan 10 root

                                       Root    Hello Max Fwd
Vlan                 Root ID           Cost    Time  Age Dly  Root Port
---------------- ------------------- --------- ----- --- ---  ------------
VLAN0010         24586 1833.9d7b.1380         4    2   20  15  Gi0/1
```

The output of the two **show** commands clearly points out the resulting priority values on each switch. First, the **show spanning-tree bridge** command lists the local switch's BID information, while the **show spanning-tree root** command lists the root's BID, plus the local switch's root cost and root port (assuming it is not the root switch). So, SW1 lists its own BID, with priority 28,682 (base 28,672, with VLAN 10) with the **show spanning-tree bridge** command. Still on SW1, the output lists the root's priority as 24,586 in VLAN 10, implied as base 24,576 plus 10 for VLAN 10, with the **show spanning-tree root** command.

Note that alternatively you could have configured the priority settings specifically. SW1 could have used the **spanning-tree vlan 10 priority 28672** command, with SW2 using the **spanning-tree vlan 10 priority 24576** command. In this particular case, both options would result in the same STP operation.

Implementing Optional STP Features

This just-completed first major section of the chapter showed examples that used PVST+ only, assuming a default global command of **spanning-tree mode pvst**. At the same time, all the configuration commands shown in that first section, commands that influence STP operation, would influence both traditional STP and RSTP operation.

This section, the second of three major sections in this chapter, now moves on to discuss some useful but optional features that make both STP and RSTP work even better.

Configuring PortFast and BPDU Guard

You can easily configure the PortFast and BPDU Guard features on any interface, but with two different configuration options. One option works best when you want to enable these features only on a few ports, and the other works best when you want to enable these features on most every access port.

First, to enable the features on just one port at a time, use the **spanning-tree portfast** and the **spanning-tree bpduguard enable** interface subcommands. Example O-6 shows an

example of the process, with SW3's F0/4 interface enabling both features. (Also, note the long warning message IOS lists when enabling PortFast; using PortFast on a port connected to other switches can indeed cause serious problems.)

Example O-6 *Enabling PortFast and BPDU Guard on One Interface*

```
SW3# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW3(config)# interface fastEthernet 0/4
SW3(config-if)# spanning-tree portfast
%Warning: portfast should only be enabled on ports connected to a single
 host. Connecting hubs, concentrators, switches, bridges, etc... to this
 interface  when portfast is enabled, can cause temporary bridging loops.
 Use with CAUTION

%Portfast has been configured on FastEthernet0/4 but will only
 have effect when the interface is in a non-trunking mode.

SW3(config-if)# spanning-tree bpduguard ?
  disable  Disable BPDU guard for this interface
  enable   Enable BPDU guard for this interface

SW3(config-if)# spanning-tree bpduguard enable
SW3(config-if)# ^Z
SW3#
```

Example O-7 shows some brief information about the interface configuration of both PortFast and BPDU Guard. Of course, the **show running-config** command (not shown) would confirm the configuration commands from Example O-6. The **show spanning-tree interface fastethernet0/4 portfast** command in Example O-7 lists the PortFast status of the interface; note that the status value of *enabled* is displayed only if PortFast is configured and the interface is up. The **show spanning-tree interface detail** command then shows a line near the end of the output that states that PortFast and BPDU Guard are enabled. Note that this command would not list those two highlighted lines of output if these two features were not enabled.

Example O-7 *Verifying PortFast and BPDU Guard Configuration*

```
SW3# show spanning-tree interface fastethernet0/4 portfast
VLAN0104            enabled

SW11# show spanning-tree interface F0/4 detail
 Port 4 (FastEthernet0/4) of VLAN0001 is designated forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.4.
   Designated root has priority 32769, address bcc4.938b.a180
   Designated bridge has priority 32769, address bcc4.938b.e500
   Designated port id is 128.4, designated path cost 19
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
```

```
  The port is in the portfast mode
  Link type is point-to-point by default
  Bpdu guard is enabled
  BPDU: sent 1721, received 0
```

PortFast and BPDU Guard are disabled by default on all interfaces, and to use them, each interface requires interface subcommands like those in Example O-6. Alternately, for both features, you can enable the feature globally. Then, for interfaces for which the feature should be disabled, you can use another interface subcommand to disable the feature.

The ability to change the global default for these features reduces the number of interface subcommands required. For instance, on an access layer switch with 48 access ports and two uplinks, you probably want to enable both PortFast and BPDU Guard on all 48 access ports. Rather than requiring the interface subcommands on all 48 of those ports, enable the features globally, and then disable them on the uplink ports.

Table O-2 summarizes the commands to enable and disable both PortFast and BPDU Guard, both globally and per interface. For instance, the global command **spanning-tree portfast default** changes the default so that all interfaces use PortFast, unless a port also has the **spanning-tree portfast disable** interface subcommand configured.

Table O-2 Enabling and Disabling PortFast and BPDU Guard, Globally and Per Interface

Action	Globally	One Interface
Disable PortFast	**no spanning-tree portfast default**	**spanning-tree portfast disable**
Enable PortFast	**spanning-tree portfast default**	**spanning-tree portfast**
Disable BPDU Guard	**no spanning-tree portfast bpduguard default**	**spanning-tree bpduguard disable**
Enable BPDU Guard	**spanning-tree portfast bpduguard default**	**spanning-tree bpduguard enable**

Example O-8 shows another new command, **show spanning-tree summary**. This command shows the current global settings for several STP parameters, including the PortFast and BPDU Guard features. This output was gathered on a switch that had enabled both PortFast and BPDU Guard globally.

Example O-8 *Displaying Status of Global Settings for PortFast and BPDU Guard*

```
SW1# show spanning-tree summary
Switch is in pvst mode
Root bridge for: none
EtherChannel misconfig guard is enabled
Extended system ID           is enabled
Portfast Default             is enabled
PortFast BPDU Guard Default  is enabled
Portfast BPDU Filter Default is disabled
Loopguard Default            is disabled
UplinkFast                   is disabled
BackboneFast                 is disabled
```

```
Configured Pathcost method used is short

Name                 Blocking Listening Learning Forwarding STP Active
-------------------- -------- --------- -------- ---------- ----------
VLAN0001                    3         0        0          2          5
-------------------- -------- --------- -------- ---------- ----------
1 vlan                      3         0        0          2          5
```

Configuring EtherChannel

Two neighboring switches can treat multiple parallel links between each other as a single logical link called an *EtherChannel*. STP operates on the EtherChannel, instead of the individual physical links, so that STP either forwards or blocks on the entire logical EtherChannel for a given VLAN. As a result, a switch in a forwarding state can then load balance traffic over all the physical links in the EtherChannel. Without EtherChannel, only one of the parallel links between two switches would be allowed to forward traffic, with the rest of the links blocked by STP.

NOTE All references to EtherChannel in this Chapter refer to Layer 2 EtherChannels, and not to Layer 3 EtherChannels.

EtherChannel may be one of the most challenging switch features to make work. First, the configuration has several options, so you have to remember the details of which options work together. Second, the switches also require a variety of other interface settings to match among all the links in the channel, so you have to know those settings as well.

This section focuses on the correct EtherChannel configuration.

Configuring a Manual EtherChannel

The simplest way to configure an EtherChannel is to add the correct **channel-group** configuration command to each physical interface, on each switch, all with the **on** keyword. The **on** keyword tells the switches to place a physical interface into an EtherChannel.

Before getting into the configuration and verification, however, you need to start using three terms as synonyms: *EtherChannel*, *PortChannel*, and *Channel-group*. Oddly, IOS uses the **channel-group** configuration command, but then to display its status, IOS uses the **show etherchannel** command. Then, the output of this **show** command refers to neither an "EtherChannel" nor a "Channel-group," instead using the term "PortChannel." So, pay close attention to these three terms in the example.

To configure an EtherChannel manually, follow these steps:

Key Topic

Step 1. Add the **channel-group** *number* **mode on** command in interface configuration mode under each physical interface that should be in the channel to add it to the channel.

Step 2. Use the same number for all commands on the same switch, but the channel-group number on the neighboring switch can differ.

Example O-9 shows a simple example, with two links between switches SW1 and SW2, as shown in Figure O-6. The configuration shows SW1's two interfaces placed into channel-group 1, with two **show** commands to follow.

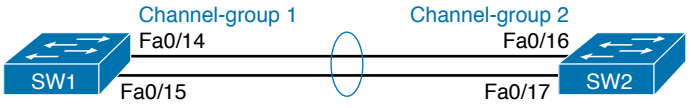


Figure O-6 *Sample LAN Used in EtherChannel Example*

Example O-9 *Configuring and Monitoring EtherChannel*

```
SW1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW1(config)# interface fa 0/14
SW1(config-if)# channel-group 1 mode on
SW1(config)# interface fa 0/15
SW1(config-if)# channel-group 1 mode on
SW1(config-if)# ^Z

SW1# show spanning-tree vlan 3

VLAN0003
  Spanning tree enabled protocol ieee
  Root ID     Priority    28675
              Address     0019.e859.5380
              Cost        12
              Port        72 (Port-channel1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    28675  (priority 28672 sys-id-ext 3)
              Address     0019.e86a.6f80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Po1              Root FWD 12        128.64   P2p Peer(STP)

SW1# show etherchannel 1 summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
```

```
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)           -        Fa0/14(P)   Fa0/15(P)
```

Take a few moments to look at the output in the two **show** commands in the example, as well. First, the **show spanning-tree** command lists Po1, short for PortChannel1, as an interface. This interface exists because of the **channel-group** commands using the **1** parameter. STP no longer operates on physical interfaces F0/14 and F0/15, instead operating on the PortChannel1 interface, so only that interface is listed in the output.

Next, note the output of the **show etherchannel 1 summary** command. It lists as a heading "Port-channel," with Po1 below it. It also lists both F0/14 and F0/15 in the list of ports, with a (P) beside each. Per the legend, the *P* means that the ports are bundled in the port channel, which is a code that means these ports have passed all the configuration checks and are valid to be included in the channel.

NOTE Cisco uses the term *EtherChannel* to refer to the concepts discussed in this section. To refer to the item configured in the switch, Cisco instead uses the term *port channel*, with the command keyword **port-channel**. For the purposes of understanding the technology, you may treat these terms as synonyms. However, it helps to pay close attention to the use of the terms *port channel* and *EtherChannel* as you work through the examples in this section, because IOS uses both.

Configuring Dynamic EtherChannels

Cisco switches support two different protocols that allow the switches to negotiate whether a particular link becomes part of an EtherChannel or not. Basically, the configuration enables the protocol for a particular channel-group number. At that point, the switch can use the protocol to send messages to/from the neighboring switch and discover whether their configuration settings pass all checks. If a given physical link passes, the link is added to the EtherChannel and used; if not, it is placed in a down state, and not used, until the configuration inconsistency can be resolved.

Cisco switches support the Cisco-proprietary Port Aggregation Protocol (PAgP) and the IEEE standard Link Aggregation Control Protocol (LACP), based on IEEE standard 802.3ad. Although differences exist between the two, to the depth discussed here, they both accomplish the same task: negotiate so that only links that pass the configuration checks are actually used in an EtherChannel.

To configure either protocol, a switch uses the **channel-group** configuration commands on each switch, but with a keyword that either means "use this protocol and begin negotiations" or "use this protocol and wait for the other switch to begin negotiations." As shown in Figure O-7, the **desirable** and **auto** keywords enable PAgP, and the **active** and **passive** keywords enable LACP. With these options, at least one side has to begin the negotiations. In other words, with PAgP, at least one of the two sides must use **desirable**, and with LACP, at least one of the two sides must use **active**.

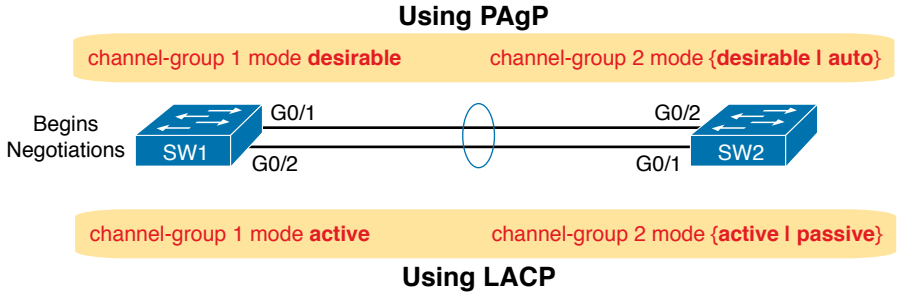


Figure O-7 *Correct EtherChannel Configuration Combinations*

NOTE Do not use the **on** parameter on one end, and either **auto** or **desirable** (or for LACP, **active** or **passive**) on the neighboring switch. The **on** option uses neither PAgP nor LACP, so a configuration that uses **on**, with PAgP or LACP options on the other end, would prevent the EtherChannel from working.

For example, in the design shown in Figure O-7, imagine both physical interfaces on both switches were configured with the **channel-group 2 mode desirable** interface subcommand. As a result, the two switches would negotiate and create an EtherChannel. Example O-10 shows the verification of that configuration, with the command **show etherchannel 2 port-channel**. This command confirms the protocol in use (PAgP, because the **desirable** keyword was configured), and the list of interfaces in the channel.

Example O-10 *EtherChannel Verification: PAgP Desirable Mode*

```
SW1# show etherchannel 2 port-channel
                Port-channels in the group:
                ---------------------------


Port-channel: Po2
------------

Age of the Port-channel   = 0d:00h:04m:04s
Logical slot/port   = 16/1          Number of ports = 2
GC                  = 0x00020001      HotStandBy port = null
Port state          = Port-channel Ag-Inuse
Protocol            =   PAgP
Port security       = Disabled
```

```
Ports in the Port-channel:

Index   Load   Port     EC state        No of bits
------+------+------+------------------+-----------
  0     00     Gi0/1    Desirable-Sl       0
  0     00     Gi0/2    Desirable-Sl       0

Time since last port bundled:    0d:00h:03m:57s    Gi0/2
```

Implementing RSTP

All you have to do to migrate from STP to RSTP is to configure the **spanning-tree mode rapid-pvst** global command on all the switches. However, for exam preparation, it helps to work through the various **show** commands, particularly to prepare for Simlet questions. Those questions can ask you to interpret **show** command output without allowing you to look at the configuration, and the output of **show** commands when using STP versus RSTP is very similar.

This third and final major section of this chapter focuses on pointing out the similarities and differences between STP and RSTP as seen in Catalyst switch configuration and verification commands. This section explains the configuration and verification of RSTP, with emphasis on how to identify RSTP features.

Identifying the STP Mode on a Catalyst Switch

Cisco Catalyst switches operate in some STP mode as defined by the **spanning-tree mode** global configuration command. Based on this command's setting, the switch is using either 802.1D STP or 802.1w RSTP, as noted in Table O-3.

Key Topic

Table O-3 Cisco Catalyst STP Configuration Modes

Parameter on spanning-tree mode Command	Uses STP or RSTP?	Protocol Listed in Command Output	Description
pvst	STP	ieee	Default; Per-VLAN Spanning Tree instance
rapid-pvst	RSTP	rstp	Like PVST, but uses RSTP rules instead of STP for each STP instance
mst	RSTP	mst	Creates multiple RSTP instances but does not require one instance per each VLAN

To determine whether a Cisco Catalyst switch uses RSTP, you can look for two types of information. First, you can look at the configuration, as noted in the left column of Table O-3. Also, some **show** commands list the STP protocol as a reference to the configuration of the **spanning-tree mode** global configuration command. A protocol of **rstp** or **mst** refers to one of the modes that uses RSTP, and a protocol of **ieee** refers to the mode that happens to use STP.

Before looking at an example of the output, review the topology in Figure O-8. The remaining RSTP examples in this chapter use this topology. In the RSTP examples in this chapter, SW1 will become root, and SW3 will block on one port (G0/2), as shown.

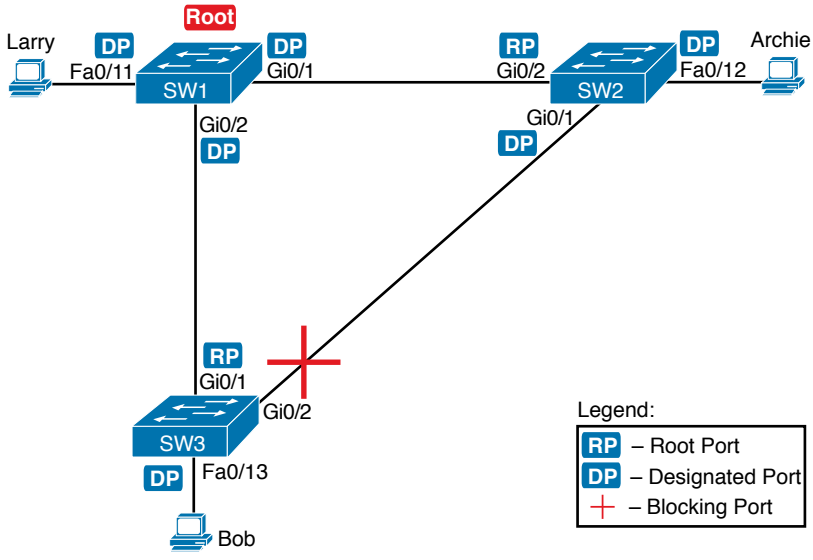


Figure O-8 *Network Topology for STP and RSTP Examples*

The first example focuses on VLAN 10, with all switches using 802.1D STP and the default setting of **spanning-tree mode pvst**. This setting creates an instance of STP per VLAN (which is the per-VLAN part of the name) and uses 802.1D STP. Each switch places the port connected to the PC into VLAN 10 and enables both PortFast and BPDU Guard. Example O-11 shows a sample configuration from switch SW3, with identical interface subcommands configured on SW1's F0/11 and SW2's F0/12 ports, respectively.

Example O-11 *Sample Configuration from Switch SW3*

```
SW3# show running-config interface Fastethernet 0/13

Building configuration...

Current configuration : 117 bytes
!
interface FastEthernet0/13
 switchport access vlan 10
 spanning-tree portfast
 spanning-tree bpduguard enable
end
```

At this point, the three switches use 802.1D STP because all use the default PVST mode. Example O-12 shows the evidence of STP's work, with only subtle and indirect clues that STP happens to be in use.

Example O-12 *Output That Confirms the Use of 802.1D STP on Switch SW3*

```
SW3# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
              Cost        4
              Port        25 (GigabitEthernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     f47f.35cb.d780
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/13              Desg FWD 19        128.13   P2p Edge
Gi0/1               Root FWD 4         128.25   P2p
Gi0/2               Altn BLK 4         128.26   P2p


SW3# show spanning-tree vlan 10 bridge

                                                  Hello  Max  Fwd
Vlan                         Bridge ID             Time  Age  Dly  Protocol
---------------- --------------------------------- -----  ---  ---  --------
VLAN0010         32778 (32768,  10) f47f.35cb.d780    2    20   15  ieee
```

The highlighted parts of the example note the references to the STP protocol as ieee, which implies that STP is in use. The term *ieee* is a reference to the original IEEE 802.1D STP standard.

To migrate this small network to use RSTP, configure the **spanning-tree mode rapid-pvst** command. This continues the use of per-VLAN spanning-tree instances, but it applies RSTP logic to each STP instance. Example O-13 shows the output of the same two commands from Example O-12 after configuring the **spanning-tree mode rapid-pvst** command on all three switches.

Example O-13 *Output That Confirms the Use of 802.1w RSTP on Switch SW3*

```
SW3# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol rstp
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
```

```
              Cost        4
              Port        25 (GigabitEthernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32778  (priority 32768 sys-id-ext 10)
              Address     f47f.35cb.d780
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/13              Desg FWD 19        128.13   P2p Edge
Gi0/1               Root FWD 4         128.25   P2p
Gi0/2               Altn BLK 4         128.26   P2p


SW3# show spanning-tree vlan 10 bridge

                                                  Hello  Max  Fwd
Vlan                         Bridge ID             Time  Age  Dly  Protocol
---------------- --------------------------------- -----  ---  ---  --------
VLAN0010         32778 (32768,  10) f47f.35cb.d780    2    20   15  rstp
```

Pay close attention to the differences between the 802.1D STP output in Example O-12 and the 802.1w RSTP output in Example O-13. Literally, the only difference is rstp instead of ieee in one place in the output of each of the two commands listed. In this case, rstp refers to the configuration of the **spanning-tree mode rapid-pvst** global config command, which implied the use of RSTP.

RSTP Port Roles

RSTP adds two port roles to STP: the alternate port and the backup port. Example O-14 repeats an excerpt from the **show spanning-tree vlan 10** command on switch SW3 to show an example of the alternate port role. SW3 (as shown earlier in Figure O-8) is not the root switch, with G0/1 as its root port and G0/2 as an alternate port.

Example O-14 *Output Confirming SW3's Root Port and Alternate Port Roles*

```
SW3# show spanning-tree vlan 10
! Lines omitted for brevity
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/13              Desg FWD 19        128.13   P2p Edge
Gi0/1               Root FWD 4         128.25   P2p
Gi0/2               Altn BLK 4         128.26   P2p
```

The good news is that the output clearly lists which port is the root port (Gi0/1) and which port is the alternate root port (Gi0/2). The only trick is to know that Altn is a shortened version of the word *alternate*.

Pay close attention to this short description of an oddity about the STP and RSTP output on Catalyst switches! Cisco Catalyst switches often show the alternate and backup ports in output even when using STP and not RSTP. The alternate and backup port concepts are RSTP concepts. The switches only converge faster using these concepts when using RSTP. But **show** command output, when using STP and not RSTP, happens to identify what would be the alternate and backup ports if RSTP were used.

Key Topic

Why might you care about such trivia? Seeing output that lists an RSTP alternate port does not confirm that the switch is using RSTP. So, do not make that assumption on the exam. To confirm that a switch uses RSTP, you must look at the configuration of the **spanning-tree mode** command, or look for the protocol as summarized back in Table O-3.

For instance, just compare the output of Example O-12 and Example O-14. Example O-12 shows output for this same SW3, with the same parameters, except that all switches used PVST mode, meaning all the switches used STP. Example O-12's output (based on STP) lists SW3's G0/2 as Altn, meaning alternate, even though the alternate port concept is not an STP concept, but an RSTP concept.

RSTP Port States

RSTP added one new port state compared to STP, discarding, using it as a replacement for the STP port states of disabled and blocking. You might think that after you configure a switch to use RSTP rather than STP, instead of seeing ports in a blocking state, you would now see the discarding state. However, the Cisco Catalyst switch output basically ignores the new term *discarding*, continuing to use the old term *blocking* instead.

For example, scan back to the most recent RSTP example (Example O-14), to the line for SW3's port G0/2. Then look for the column with heading STS, which refers to the status or state. The output shows G0/2 is listed as BLK, or blocking. In theory, because SW3 uses RSTP, the port state ought to be discarding, but the switch IOS continues to use the older notation of BLK for blocking.

Just as one more bit of evidence, the command **show spanning-tree vlan 10 interface gigabitethernet0/2 state** lists the STP or RSTP port state with the state fully spelled out. Example O-15 shows this command, taken from SW3, for interface G0/2. Note the fully spelled-out *blocking* term instead of the RSTP term *discarding*.

Example O-15 *SW3, an RSTP Switch, Continues to Use the Old Blocking Term*

```
SW3# show spanning-tree vlan 10 interface gigabitEthernet 0/2 state
VLAN0010            blocking
```

RSTP Port Types

Cisco Catalyst switches determine the RSTP port type based on two port settings: the current duplex (full or half) and whether the PortFast feature is enabled. First, full duplex tells the switch to use port type point-to-point, with half duplex telling the switch to use port type shared. Enabling PortFast tells the switch to treat the port as an edge port. Table O-4 summarizes the combinations.

Table O-4 RSTP Port Types

Type	Current Duplex Status	Is Spanning-Tree PortFast Configured?
Point-to-point	Full	No
Point-to-point edge	Full	Yes
Shared	Half	No
Shared edge[1]	Half	Yes

[1] Cisco recommends against using this combination, to avoid causing loops.

You can easily find the RSTP port types in the output of several commands, including the same **show spanning-tree** command in Example O-16. Example O-16 lists output from switch SW2, with a hub added off SW2's F0/18 port (not shown in Figure O-8). The hub was added so that the output in Example O-16 lists a shared port (noted as Shr) to go along with the point-to-point ports (noted as P2p).

Example O-16 *RSTP Port Types*

```
SW2# show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol rstp
  Root ID     Priority    32778
              Address     1833.9d7b.0e80
              Cost        4
              Port        26 (GigabitEthernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     1833.9d7b.1380
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Fa0/12              Desg FWD 19        128.12   P2p Edge
Fa0/18              Desg FWD 19        128.18   Shr
Gi0/1               Desg FWD 4         128.25   P2p
Gi0/2               Root FWD 4         128.26   P2p
```

For exam prep, again note an odd fact about the highlighted output in Example O-16: The port type details appear in the output when using both STP and RSTP. For example, refer to Example O-12 again, which shows output from SW3 when using STP (when configured for PVST mode). The Type column also identifies point-to-point and edge interfaces.

Command References

Tables O-5 and O-6 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table O-5 Appendix O Configuration Command Reference

Command	Description
spanning-tree mode {**pvst** | **rapid-pvst** | **mst**}	Global configuration command to set the STP mode.
spanning-tree [**vlan** *vlan-number*] **root primary**	Global configuration command that changes this switch to the root switch. The switch's priority is changed to the lower of either 24,576 or 4096 less than the priority of the current root bridge when the command was issued.
spanning-tree [**vlan** *vlan-number*] **root secondary**	Global configuration command that sets this switch's STP base priority to 28,672.
spanning-tree [**vlan** *vlan-id*] {**priority** *priority*}	Global configuration command that changes the bridge priority of this switch for the specified VLAN.
spanning-tree [**vlan** *vlan-number*] **cost** *cost*	Interface subcommand that changes the STP cost to the configured value.
spanning-tree [**vlan** *vlan-number*] **port-priority** *priority*	Interface subcommand that changes the STP port priority in that VLAN (0 to 240, in increments of 16).
channel-group *channel-group-number* **mode** {**auto** | **desirable** | **active** | **passive** | **on**}	Interface subcommand that enables EtherChannel on the interface.
spanning-tree portfast	Interface subcommand that enables PortFast on the interface.
spanning-tree bpduguard enable	Interface subcommand that enables BPDU Guard on an interface.
spanning-tree portfast default	Global command that changes the switch default for PortFast on access interfaces from disabled to enabled.
spanning-tree portfast bpduguard default	Global command that changes the switch default for BPDU Guard on access interfaces from disabled to enabled.
no spanning-tree portfast default	Global command that changes the global setting for PortFast to disabled.
no spanning-tree portfast bpduguard default	Global command that changes the global setting for BPDU Guard to disabled.
spanning-tree portfast disable	Interface subcommand that disables PortFast on the interface.
spanning-tree bpduguard disable	Interface subcommand that disables BPDU Guard on an interface.

Table O-6 Appendix O EXEC Command Reference

Command	Description
show spanning-tree	Lists details about the state of STP on the switch, including the state of each port.
show spanning-tree interface *interface-id*	Lists STP information only for the specified port.
show spanning-tree vlan *vlan-id*	Lists STP information for the specified VLAN.
show spanning-tree [**vlan** *vlan-id*] **root**	Lists information about each VLAN's root or for just the specified VLAN.
show spanning-tree [**vlan** *vlan-id*] **bridge**	Lists STP information about the local switch for each VLAN or for just the specified VLAN.
show spanning-tree summary	Lists global STP settings for a switch, including the default PortFast and BPDU Guard settings, and the VLANs for which this switch is the root switch.
debug spanning-tree events	Causes the switch to provide informational messages about changes in the STP topology.
show spanning-tree interface *type number* **portfast**	Lists a one-line status message about PortFast on the listed interface.
show etherchannel [*channel-group-number*] {**brief** | **detail** | **port** | **port-channel** | **summary**}	Lists information about the state of EtherChannels on this switch.

APPENDIX P

LAN Troubleshooting

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 4 of the book *CCNA Routing and Switching ICND2 200-105 Official Cert Guide*, published in 2016.

This chapter discusses the LAN topics discussed in depth in the first three chapters, plus a few prerequisite topics, from a troubleshooting perspective.

Troubleshooting for any networking topic requires a slightly different mindset as compared to thinking about configuration and verification. When thinking about configuration and verification, it helps to think about basic designs, learn how to configure the feature correctly, and learn how to verify the correct configuration is indeed working correctly. However, to learn how to troubleshoot, you need to think about symptoms when the design is incorrect, or if the configuration does not match the good design. What symptoms occur when you make one type of mistake or another? This chapter looks at the common types of mistakes, and works through how to look at the status with **show** commands to find those mistakes.

This chapter breaks the material into four major sections. The first section tackles the largest topic, STP troubleshooting. STP is not likely to fail as a protocol; instead, STP may not be operating as designed, so the task is to find how STP is currently working and discover how to then make the configuration implement the correct design. The second major section then moves on to Layer 2 EtherChannels, which have a variety of small potential problems that can prevent the dynamic formation of an EtherChannel.

The third major section of the chapter focuses on the data plane forwarding of Ethernet frames on LAN switches, in light of VLANs, trunks, STP, and EtherChannels. That same section reviews the Layer 2 forwarding logic of a switch in light of these features. The fourth and final major section then examines VLAN and trunking issues, and how those issues impact switch forwarding.

Note that a few of the subtopics listed within the exam topics at the beginning of this chapter are not discussed in this chapter. This chapter does not discuss VTP beyond its basic features or Layer 3 EtherChannels.

Foundation Topics

Troubleshooting STP

STP questions tend to intimidate many test takers. STP uses many rules, with tiebreakers in case one rule ends with a tie. Without much experience with STP, people tend to distrust their own answers. Also, even those of us with networking jobs already probably do not troubleshoot STP very often, because STP works well. Often, troubleshooting STP is not about STP failing to do its job but rather about STP working differently than designed, with a different root switch, or different root ports (RP), and so on. Seldom does STP troubleshooting begin with a case in which STP has failed to prevent a loop.

This section reviews the rules for STP, while emphasizing some important troubleshooting points. In particular, this section takes a closer look at the tiebreakers that STP uses to make decisions. It also makes some practical suggestions about how to go about answering exam questions such as "which switch is the root switch?"

Determining the Root Switch

Determining the STP root switch is easy if you know all the switches' BIDs: Just pick the lowest value. If the question lists the priority and MAC address separately, as is common in some **show** command output, pick the switch with the lowest priority, or in the case of a tie, pick the lower MAC address value.

And just to be extra clear, STP does not have nor need a tiebreaker for electing the root switch. The BID uses a switch universal MAC address as the last 48 bits of the BID. These MAC addresses are unique in the universe, so there should never be identical BIDs or the need for a tiebreaker.

For the exam, a question that asks about the root switch might not be so simple as listing a bunch of BIDs and asking you which one is "best." A more likely question is a simulator (sim) question in which you have to do any **show** commands you like or a multiple choice question that lists the output from only one or two commands. Then you have to apply the STP algorithm to figure out the rest.

When faced with an exam question using a simulator, or just the output in an exhibit, use a simple strategy of ruling out switches, as follows:

Step 1. Begin with a list or diagram of switches, and consider all as possible root switches.

Step 2. Rule out any switches that have an RP (**show spanning-tree**, **show spanning-tree root**), because root switches do not have an RP.

Step 3. Always try **show spanning-tree**, because it identifies the local switch as root directly: "This switch is the root" on the fifth line of output.

Step 4. Always try **show spanning-tree root**, because it identifies the local switch as root indirectly: The RP column is empty if the local switch is the root.

Step 5. When using a sim, rather than try switches randomly, chase the RPs. For example, if starting with SW1, and SW1's G0/1 is an RP, next try the switch on the other end of SW1's G0/1 port.

Step 6. When using a sim, use **show spanning-tree vlan** *x* on a few switches and record the root switch, RP, and designated port (DP). This strategy can quickly show you most STP facts.

The one step in this list that most people ignore is the idea of ruling out switches that have an RP. Root switches do not have an RP, so any switch with an RP can be ruled out as not being the root switch for that VLAN. Example P-1 shows two commands on switch SW2 in some LAN that confirms that SW2 has an RP and is therefore not the root switch.

Example P-1 *Ruling Out Switches as Root Based on Having a Root Port*

```
SW2# show spanning-tree vlan 20 root

                                        Root Hello Max Fwd
Vlan                Root ID             Cost  Time Age Dly  Root Port
---------------- ------------------- --------- ----- --- ---  ------------
VLAN0020         32788 1833.9d7b.0e80          4    2   20  15  Gi0/2

SW2# show spanning-tree vlan 20

VLAN0020
  Spanning tree enabled protocol ieee
  Root ID     Priority    32788
              Address     1833.9d7b.0e80
              Cost        4
              Port        26 (GigabitEthernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32788  (priority 32768 sys-id-ext 20)
              Address     1833.9d7b.1380
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  15  sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi0/1               Desg FWD 4         128.25   P2p
Gi0/2               Root FWD 4         128.26   P2p
```

Both commands identify SW2's G0/2 port as its RP, so if you follow the suggestions, the next switch to try in a sim question would be the switch on the other end of SW2's G0/2 interface.

Determining the Root Port on Nonroot Switches

Determining the RP of a switch when **show** command output is available is relatively easy. As shown recently in Example P-1, both **show spanning-tree** and **show spanning-tree root** list the root port of the local switch, assuming it is not the root switch. The challenge comes more when an exam question makes you think through how the switches choose the RP based on the root cost of each path to the root switch, with some tiebreakers as necessary.

As a review, each nonroot switch has one, and only one, RP for a VLAN. To choose its RP, a switch listens for incoming Hello bridge protocol data units (BPDU). For each received Hello, the switch adds the cost listed in the hello BPDU to the cost of the incoming interface (the interface on which the Hello was received). That total is the root cost over that path. The lowest root cost wins, and the local switch uses its local port that is part of the least root cost path as its root port.

Most humans can analyze what STP chooses by using a network diagram and a slightly different algorithm. Instead of thinking about Hello messages and so on, approach the question as this: the sum of all outgoing port costs between the nonroot switch and the root. Repeating a familiar example, with a twist, Figure P-1 shows the calculation of the root cost. Note that SW3's Gi0/1 port has yet again had its cost configured to a different value.

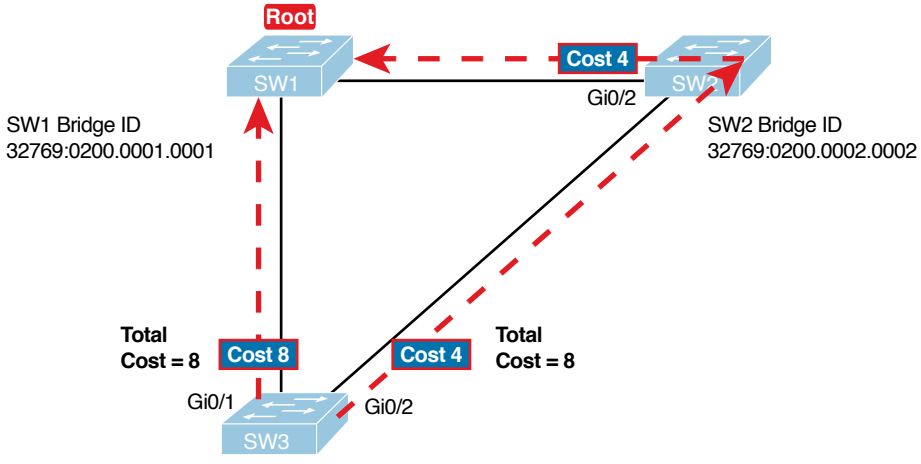


Figure P-1 *SW3's Root Cost Calculation Ends in a Tie*

STP Tiebreakers When Choosing the Root Port

Figure P-1 shows the easier process of adding the STP costs of the outgoing interfaces over each from SW3, a nonroot, to SW1, the root. It also shows a tie (on purpose), to talk about the tiebreakers.

When a switch chooses its root port, the first choice is to choose the local port that is part of the least root cost path. When those costs tie, the switch picks the port connected to the neighbor with the lowest BID. This tiebreaker usually breaks the tie, but not always. So, for completeness, the three tiebreakers are, in the order a switch uses them, as follows:

1. Choose based on the lowest neighbor bridge ID.
2. Choose based on the lowest neighbor port priority.
3. Choose based on the lowest neighbor internal port number.

(Note that the switch only considers the root paths that tie when thinking about these tiebreakers.)

For example, Figure P-1 shows that SW3 is not root and that its two paths to reach the root tie with their root costs of 8. The first tiebreaker is the lowest neighbor's BID. SW1's BID value is lower than SW2's, so SW3 chooses its G0/1 interface as its RP in this case.

The last two RP tiebreakers come into play only when two switches connect to each other with multiple links, as shown in Figure P-2. In that case, a switch receives Hellos on more than one port from the same neighboring switch, so the BIDs tie.

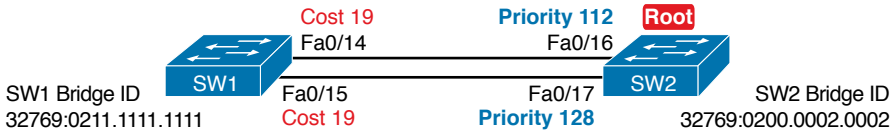


Figure P-2 *Topology Required for the Last Two Tiebreakers for Root Port*

In this particular example, SW2 becomes root, and SW1 needs to choose its RP. SW1's port costs tie, at 19 each, so SW1's root cost over each path will tie at 19. SW2 sends Hellos over each link to SW1, so SW1 cannot break the tie based on SW1's neighbor BID because both list SW2's BID. So, SW1 has to turn to the other two tiebreakers.

NOTE In real life, most engineers would put these two links into an EtherChannel.

The next tiebreaker is a configurable option: the neighboring switch's port priority on each neighboring switch interface. Cisco switch ports default to a setting of 128, with a range of values from 0 through 255, with lower being better (as usual). In this example, the network engineer has set SW2's F0/16 interface with the **spanning-tree vlan 10 port-priority 112** command. SW1 learns that the neighbor has a port priority of 112 on the top link and 128 on the bottom, so SW1 uses its top (F0/14) interface as the root port.

If the port priority ties, which it often does due to the default values, STP relies on an internal port numbering on the neighbor. Cisco switches assign an internal integer to identify each interface on the switch. The nonroot looks for the neighbor's lowest internal port number (as listed in the Hello messages) and chooses its RP based on the lower number.

Cisco switches use an obvious numbering, with Fa0/1 having the lowest number, then Fa0/2, then Fa0/3, and so on. So, in Figure P-2, SW2's Fa0/16 would have a lower internal port number than Fa0/17; SW1 would learn those numbers in the Hello; and SW1 would use its Fa0/14 port as its RP.

Suggestions for Attacking Root Port Problems on the Exam

Exam questions that make you think about the RP can be easy if you know where to look and the output of a few key commands is available. However, the more conceptual the question, the more you have to calculate the root cost over each path, correlate that to different **show** commands, and put the ideas together. The following list makes a few suggestions about how to approach STP problems on the exam:

Key Topic

1. If available, look at the **show spanning-tree** and **show spanning-tree root** commands. Both commands list the root port and the root cost (see Example P-1).
2. The **show spanning-tree** command lists cost in two places: the root cost at the top, in the section about the root switch; and the interface cost, at the bottom, in the per-interface section. Be careful, though; the cost at the bottom is the interface cost, not the root cost!

P

3. For problems where you have to calculate a switch's root cost:
   a. Memorize the default cost values: 100 for 10 Mbps, 19 for 100 Mbps, 4 for 1 Gbps, and 2 for 10 Gbps.
   b. Look for any evidence of the **spanning-tree cost** configuration command on an interface, because it overrides the default cost. Do not assume default costs are used.
   c. When you know a default cost is used, if you can, check the current actual speed as well. Cisco switches choose STP cost defaults based on the current speed, not the maximum speed.

Determining the Designated Port on Each LAN Segment

Each LAN segment has a single switch that acts as the designated port (DP) on that segment. On segments that connect a switch to a device that does not even use STP—for example, segments connecting a switch to a PC or a router—the switch always wins, because it is the only device sending a Hello onto the link. However, links with two switches require a little more work to discover which should be the DP. By definition:

Key Topic

Step 1. For switches connected to the same LAN segment, the switch with the lowest cost to reach the root, as advertised in the Hello they send onto the link, becomes the DP on that link.

Step 2. In case of a tie, among the switches that tied on cost, the switch with the lowest BID becomes the DP.

For example, consider Figure P-3. This figure notes the root, RPs, and DPs and each switch's least cost to reach the root over its respective RP.

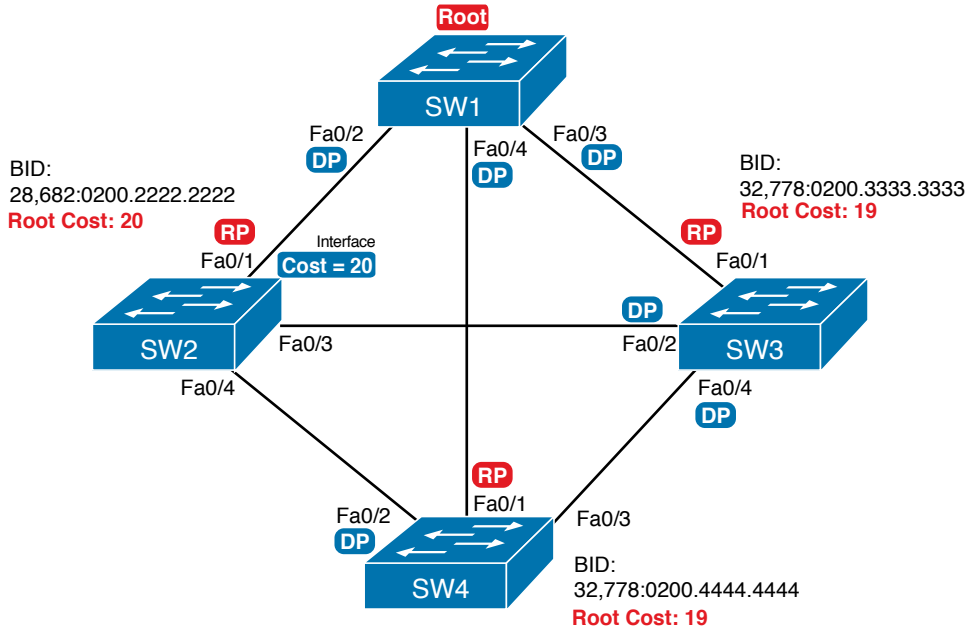


Figure P-3 *Picking the DPs*

Focus on the segments that connect the nonroot switches for a moment:

SW2–SW4 segment: SW4 wins because of its root cost of 19, compared to SW2's root cost of 20.

SW2–SW3 segment: SW3 wins because of its root cost of 19, compared to SW2's root cost of 20.

SW3–SW4 segment: SW3 and SW4 tie on root cost, both with root cost 19. SW3 wins due to its better (lower) BID value.

Interestingly, SW2 loses and does not become DP on the links to SW3 and SW4 even though SW2 has the better (lower) BID value. The DP tiebreaker does use the lowest BID, but the first DP criteria is the lowest root cost, and SW2's root cost happens to be higher than SW3's and SW4's.

NOTE A single switch can connect two or more interfaces to the same collision domain, and compete to become DP, if hubs are used. In such cases, two different switch ports on the same switch tie, the DP choice uses the same two final tiebreakers as used with the RP selection: the lowest interface STP priority, and if that ties, the lowest internal interface number.

Suggestions for Attacking Designated Port Problems on the Exam

As with exam questions asking about the RP, exam questions that make you think about the DP can be easy if you know where to look and the output of a few key commands is available. However, the more conceptual the question, the more you have to think about the criteria for choosing the DP: first the root cost of the competing switches, and then the better BID if they tie based on root cost.

The following list gives some tips to keep in mind when digging into a given DP issue. Some of this list repeats the suggestions for finding the RP, but to be complete, this list includes each idea as well.

Key Topic

1. If available, look at the **show spanning-tree** commands, at the list of interfaces at the end of the output. Then, look for the Role column, and look for Desg, to identify any DPs.
2. Identify the root cost of a switch directly by using the **show spanning-tree** command. But be careful! This command lists the cost in two places, and only the mention at the top, in the section about the root, lists the root cost.
3. For problems where you have to calculate a switch's root cost, do the following:
   a. Memorize the default cost values: 100 for 10 Mbps, 19 for 100 Mbps, 4 for 1 Gbps, and 2 for 10 Gbps.
   b. Look for any evidence of the **spanning-tree cost** configuration command on an interface, because it overrides the default cost. Do not assume default costs are used.
   c. When you know a default cost is used, if you can, check the current actual speed as well. Cisco switches choose STP cost defaults based on the current speed, not the maximum speed.

STP Convergence

STP puts each RP and DP into a forwarding state, and ports that are neither RP nor DP into a blocking state. Those states may remain as is for days, weeks, or months. But at some point, some switch or link will fail, a link may change speeds (changing the STP cost), or the STP configuration may change. Any of these events can cause switches to repeat their STP algorithm, which may in turn change their own RP and any ports that are DPs.

When STP converges based on some change, not all the ports have to change their state. For instance, a port that was forwarding, if it still needs to forward, just keeps on forwarding. Ports that were blocking that still need to block keep on blocking. But when a port needs to change state, something has to happen, based on the following rules:

Key Topic

- For interfaces that stay in the same STP state, nothing needs to change.
- For interfaces that need to move from a forwarding state to a blocking state, the switch immediately changes the state to blocking.
- For interfaces that need to move from a blocking state to a forwarding state, the switch first moves the interface to listening state, then learning state, each for the time specified by the forward delay timer (default 15 seconds). Only then is the interface placed into forwarding state.

Because the transition from blocking to forwarding does require some extra steps, you should be ready to respond to conceptual questions about the transition.

Troubleshooting Layer 2 EtherChannel

EtherChannels can prove particularly challenging to troubleshoot for a couple of reasons. First, you have to be careful to match the correct configuration, and there are many more incorrect configuration combinations than there are correct combinations. Second, many interface settings must match on the physical links, both on the local switch and on the neighboring switch, before a switch will add the physical link to the channel. This second major section in the chapter works through both sets of issues.

Incorrect Options on the channel-group Command

The rules for the small set of working configuration options on the **channel-group** command can be summarized as follows, for a single EtherChannel:

Key Topic

1. On the local switch, all the **channel-group** commands for all the physical interfaces must use the same channel-group number.
2. The channel-group number can be different on the neighboring switches.
3. If using the **on** keyword, you must use it on the corresponding interfaces of both switches.
4. If you use the **desirable** keyword on one switch, the switch uses PAgP; the other switch must use either **desirable** or **auto**.
5. If you use the **active** keyword on one switch, the switch uses LACP; the other switch must use either **active** or **passive**.

These rules summarize the correct configuration options, but the options actually leave many more incorrect choices. The following list shows some incorrect configurations that the switches allow, even though they would result in the EtherChannel not working. The list

compares the configuration on one switch to another based on the physical interface configuration. Each lists the reasons why the configuration is incorrect.

- Configuring the **on** keyword on one switch, and **desirable**, **auto**, **active**, or **passive** on the other switch. The **on** keyword does not enable PAgP, and does not enable LACP, and the other options rely on PAgP or LACP.
- Configuring the **auto** keyword on both switches. Both use PAgP, but both wait on the other switch to begin negotiations.
- Configuring the **passive** keyword on both switches. Both use LACP, but both wait on the other switch to begin negotiations.
- Configuring the **active** keyword on one switch and either **desirable** or **auto** on the other switch. The **active** keyword uses LACP, whereas the other keywords use PAgP.
- Configuring the **desirable** keyword on one switch and either **active** or **passive** on the other switch. The **desirable** keyword uses PAgP, whereas the other keywords use LACP.

Example P-2 shows an example that matches the last item in the list. In this case, SW1's two ports (F0/14 and F0/15) have been configured with the **desirable** keyword, and SW2's matching F0/16 and F0/17 have been configured with the **active** keyword. The example lists some telling status information about the failure, with notes following the example.

Example P-2 *Incorrect Configuration Using Mismatched PortChannel Protocols*

```
SW1# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------

1      Po1(SD)          PAgP      Fa0/14(I)    Fa0/15(I)

SW1# show interfaces status | include Po|14|15
Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/14                       connected    301        a-full  a-100 10/100BaseTX
Fa0/15                       connected    301        a-full  a-100 10/100BaseTX
Po1                          notconnect   unassigned   auto   auto
```

P

Start at the top, in the legend of the **show etherchannel summary** command. The *D* code letter means that the channel itself is down, with *S* meaning that the channel is a Layer 2 EtherChannel. Code *I* means that the physical interface is working independently from the PortChannel (described as "stand-alone"). Then, the bottom of that command's output highlights PortChannel 1 (Po1) as Layer 2 EtherChannel in a down state (SD), with F0/14 and F0/15 as stand-alone interfaces (I).

Interestingly, because the problem is a configuration mistake, the two physical interfaces still operate independently, as if the PortChannel did not exist. The last command in the example shows that while the PortChannel 1 interface is down, the two physical interfaces are in a connected state.

NOTE As a suggestion for attacking EtherChannel problems on the exam, rather than memorizing all the incorrect configuration options, concentrate on the list of correct configuration options. Then look for any differences between a given question's configuration as compared to the known correct configurations and work from there.

Configuration Checks Before Adding Interfaces to EtherChannels

Even when the **channel-group** commands have all been configured correctly, other configuration settings can cause problems as well. This last topic examines those configuration settings and their impact.

First, a local switch checks each new physical interface that is configured to be part of an EtherChannel, comparing each new link to the existing links. That new physical interface's settings must be the same as the existing links' settings; otherwise, the switch does not add the new link to the list of approved and working interfaces in the channel. That is, the physical interface remains configured as part of the PortChannel, but it is not used as part of the channel, often being placed into some nonworking state.

The list of items the switch checks includes the following:

Key Topic

- Speed
- Duplex
- Operational access or trunking state (all must be access, or all must be trunks)
- If an access port, the access VLAN
- If a trunk port, the allowed VLAN list (per the **switchport trunk allowed** command)
- If a trunk port, the native VLAN
- STP interface settings

In addition, switches check the settings on the neighboring switch. To do so, the switches either use PAgP or LACP (if already in use), or use Cisco Discovery Protocol (CDP) if using manual configuration. The neighbor must match on all parameters in this list except the STP settings.

As an example, SW1 and SW2 again use two links in one EtherChannel. Before configuring the EtherChannel, SW1's F0/15 was given a different STP port cost than F0/14. Example P-3 picks up the story just after configuring the correct **channel-group** commands, when the switch is deciding whether to use F0/14 and F0/15 in this EtherChannel.

Example P-3 *Local Interfaces Fail in EtherChannel Because of Mismatched STP Cost*

```
*Mar  1 23:18:56.132: %PM-P-ERR_DISABLE: channel-misconfig (STP) error detected on
  Po1, putting Fa0/14 in err-disable state
*Mar  1 23:18:56.132: %PM-P-ERR_DISABLE: channel-misconfig (STP) error detected on
  Po1, putting Fa0/15 in err-disable state
*Mar  1 23:18:56.132: %PM-P-ERR_DISABLE: channel-misconfig (STP) error detected on
  Po1, putting Po1 in err-disable state
*Mar  1 23:18:58.120: %LINK-3-UPDOWN: Interface FastEthernet0/14, changed state to
  down
*Mar  1 23:18:58.137: %LINK-3-UPDOWN: Interface Port-channel1, changed state to down
*Mar  1 23:18:58.137: %LINK-3-UPDOWN: Interface FastEthernet0/15, changed state to
  down

SW1# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SD)           -        Fa0/14(D)   Fa0/15(D)
```

The messages at the top of the example specifically state what the switch does when determining whether the interface settings match. In this case, SW1 detects the different STP costs. SW1 does not use F0/14, does not use F0/15, and even places them into an err-disabled state. The switch also puts the PortChannel into err-disabled state. As a result, the PortChannel is not operational, and the physical interfaces are also not operational.

To solve this problem, you must reconfigure the physical interfaces to use the same STP settings. In addition, the PortChannel and physical interfaces must be **shutdown**, and then **no shutdown**, to recover from the err-disabled state. (Note that when a switch applies the **shutdown** and **no shutdown** commands to a PortChannel, it applies those same commands to the physical interfaces, as well; so, just do the **shutdown/no shutdown** on the PortChannel interface.)

Analyzing the Switch Data Plane Forwarding

STP and EtherChannel both have an impact on what a switch's forwarding logic can use. STP limits which interfaces the data plane even considers using by placing some ports in a

blocking state (STP) or discarding state (RSTP), which in turn tells the data plane to simply not use that port. EtherChannel gives the data plane new ports to use in the switch's MAC address table—EtherChannels—while telling the data plane to not use the underlying physical interfaces in an EtherChannel in the MAC table.

This (short) third major section of the chapter explores the impact of STP and EtherChannel on data plane logic and a switch's MAC address table.

Predicting STP Impact on MAC Tables

Consider the small LAN shown in Figure P-4. The LAN has only three switches, with redundancy, just big enough to make the point for this next example. The LAN supports two VLANs, 1 and 2, and the engineer has configured STP such that SW3 blocks on a different port in each of the two VLANs. As a result, VLAN 1 traffic would flow from SW3 to SW1 next, and in VLAN 2, traffic would flow from SW3 to SW2 next instead.

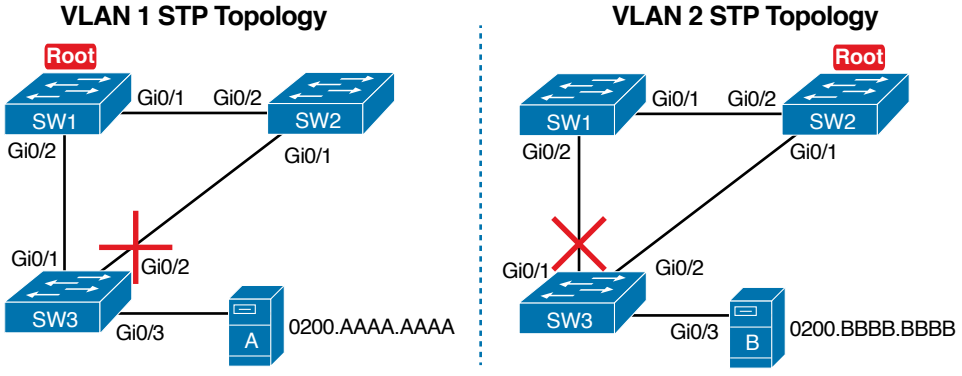


Figure P-4 *Two Different STP Topologies for Same Physical LAN, Two Different VLANs*

Looking at diagrams like those in Figure P-4 makes the forwarding path obvious. Although the figure shows the traffic path, that path is determined by switch MAC learning, which is then impacted by the ports on which STP has set a blocking or discarding state.

For example, consider VLAN 1's STP topology in Figure P-4. Remember, STP blocks on a port on one switch, not on both ends of the link. So, in the case of VLAN 1, SW3's G0/2 port blocks, but SW2's G0/1 does not. Even so, by blocking on a port on one end of the link, that act effectively stops any MAC learning from happening by either device on the link. That is, SW3 learns no MAC addresses on its G0/2 port, and SW2 learns no MAC addresses on its G0/1 port, for these reasons:

- **SW2 learns no MAC addresses on G0/1:** On the blocking (SW3) end of the SW3–SW2 trunk, SW3 will not send frames out that link to SW2, so SW2 will never receive frames from which to learn MAC addresses on SW2's G0/1.
- **SW3 learns no MAC addresses on G0/2:** On the not blocking (SW2) end of the SW3–SW2 trunk, SW2 will flood frames out that port. SW3 receives those frames, but because SW3 blocks, SW3 ignores those received frames and does not learn their MAC addresses.

Given that discussion, can you predict the MAC table entries on each of the three switches for the MAC addresses of servers A and B in Figure P-4? On switch SW2, the entry for

server A, in VLAN 1, should refer to SW2's G0/2 port, pointing to SW1 next, matching the figure. But SW2's entry for server B, in VLAN 2, references SW2's G0/1 port, again matching the figure. Example P-4 shows the MAC tables on SW1 and SW2 as a confirmation.

Example P-4 *Examining SW1 and SW2 Dynamic MAC Address Table Entries*

```
SW1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  1     0200.AAAA.AAAA    DYNAMIC     Gi0/2
  2     0200.BBBB.BBBB    DYNAMIC     Gi0/1
```

```
SW2# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  1     0200.AAAA.AAAA    DYNAMIC     Gi0/2
  2     0200.BBBB.BBBB    DYNAMIC     Gi0/1
```

Predicting EtherChannel Impact on MAC Tables

Most designs use multiple links between switches, with those links configured to be part of an EtherChannel. What does that do to the MAC forwarding logic? In short, the switch uses the PortChannel interfaces, and not the physical interfaces bundled into the EtherChannel, in the MAC address table. Specifically:

MAC learning: Frames received in a physical interface that is part of a PortChannel are considered to arrive on the PortChannel interface. So, MAC learning adds the PortChannel interface rather than the physical interface to the MAC address table.

MAC forwarding: The forwarding process will find a PortChannel port as an outgoing interface when matching the MAC address table. Then the switch must take the additional step to choose the outgoing physical interface, based on the load-balancing preferences configured for that PortChannel.

For example, consider Figure P-5, which updates previous Figure P-4 with two-link PortChannels between each pair of switches. With VLAN 1 blocking again on switch SW3, but this time on SW3's PortChannel3 interface, what MAC table entries would you expect to see in each switch? Similarly, what MAC table entries would you expect to see for VLAN 2, with SW3 blocking on its PortChannel2 interface?

The logic of which entries exist on which ports mirrors the logic with the earlier example surrounding Figure P-4. In this case, the interfaces just happen to be PortChannel interfaces. Example P-5 shows the same command from the same two switches as Example P-4: **show mac address-table dynamic** from both SW1 and SW2. (Note that to save length, the MAC table output shows only the entries for the two servers in Figure P-5.)

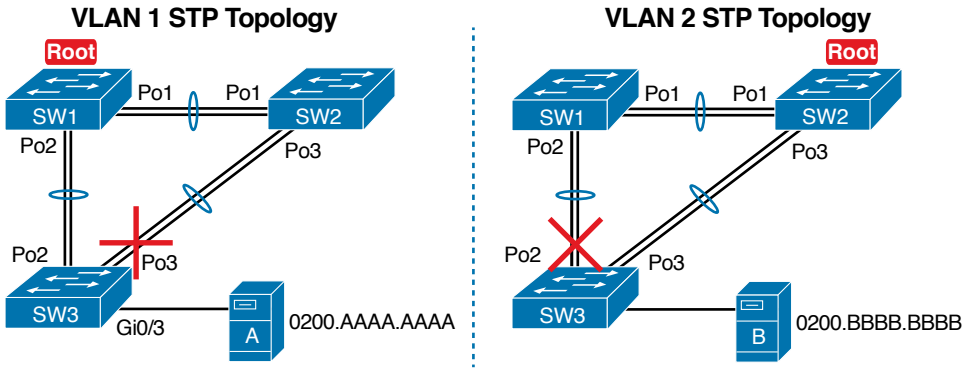


Figure P-5 *VLAN Topology with PortChannels Between Switches*

Example P-5 *SW1 and SW2 MAC Tables with PortChannel Ports Listed*

```
SW1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  1     0200.AAAA.AAAA    DYNAMIC     Po2
  2     0200.BBBB.BBBB    DYNAMIC     Po1
```

```
SW2# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  1     0200.AAAA.AAAA    DYNAMIC     Po1
  2     0200.BBBB.BBBB    DYNAMIC     Po3
```

Switches use one of many load-balancing options to then choose the physical interface to use after matching MAC table entries like those shown in Example P-5. By default, Cisco Layer 2 switches often default to use a balancing method based on the source MAC address. In particular, the switch looks at the low-order bits of the source MAC address (which are on the far right of the MAC address in written form). This approach increases the chances that the balancing will be spread somewhat evenly based on the source MAC addresses in use.

Choosing the VLAN of Incoming Frames

To wrap up the analysis of switch data plane forwarding, this section mostly reviews topics already discussed, but it serves to emphasize some important points. The topic is simply this: How does a switch know which VLAN a frame is a part of as the frame enters a switch? You have seen all the information needed to answer this question already, but take the time to review.

First, some interfaces trunk, and in those cases, the frame arrives with a VLAN ID listed in the incoming trunking header. In other cases, the frame does not arrive with a trunking

header, and the switch must look at local configuration. But because the switch will match both the destination MAC address and the frame VLAN ID when matching the MAC address table, knowing how the switch determines the VLAN ID is important.

The following list reviews and summarizes the key points of how a switch determines the VLAN ID to associate with an incoming frame:

Key Topic

Step 1. If the port is an access port, associate the frame with the configured access VLAN (**switchport access vlan** *vlan_id*).

Step 2. If the port is a voice port, or has both an IP Phone and PC (or other data device) connected to the phone:

A. Associate the frames from the data device with the configured access VLAN (as configured with the **switchport access vlan** *vlan_id* command).

B. Associate the frames from the phone with the VLAN ID in the 802.1Q header (as configured with the **switchport voice vlan** *vlan_id* command).

Step 3. If the port is a trunk, determine the frame's tagged VLAN, or if there is no tag, use that incoming interface's native VLAN ID (**switchport trunk native** *vlan_id*).

P

Troubleshooting VLANs and VLAN Trunks

A switch's data plane forwarding processes depend in part on VLANs and VLAN trunking. Before a switch can forward frames in a particular VLAN, the switch must know about a VLAN and the VLAN must be active. And before a switch can forward a frame over a VLAN trunk, the trunk must currently allow that VLAN to pass over the trunk.

This final major section in this chapter focuses on VLAN and VLAN trunking issues, specifically issues that impact the frame switching process. The issues are as follows:

Key Topic

Step 1. Identify all access interfaces and their assigned access VLANs and reassign into the correct VLANs if incorrect.

Step 2. Determine whether the VLANs both exist (either configured or learned with the VLAN Trunking Protocol [VTP]) and are active on each switch. If not, configure and activate the VLANs to resolve problems as needed.

Step 3. Check the allowed VLAN lists, on the switches on both ends of the trunk, and ensure that the lists of allowed VLANs are the same.

Step 4. Check for incorrect configuration settings that result in one switch operating as a trunk, with the neighboring switch not operating as a trunk.

Step 5. Check the allowed VLANs on each trunk, to make sure that the trunk has not administratively removed a VLAN from being supported on a trunk.

Access VLAN Configuration Incorrect

To ensure that each access interface has been assigned to the correct VLAN, engineers simply need to determine which switch interfaces are access interfaces instead of trunk

interfaces, determine the assigned access VLANs on each interface, and compare the information to the documentation. The **show** commands listed in Table P-1 can be particularly helpful in this process.

Key Topic

Table P-1 Commands That Can Find Access Ports and VLANs

EXEC Command	Description
show vlan brief **show vlan**	Lists each VLAN and all interfaces assigned to that VLAN (but does not include operational trunks)
show vlan id *num*	Lists both access and trunk ports in the VLAN
show interfaces *type number* **switchport**	Identifies the interface's access VLAN and voice VLAN, plus the configured and operational mode (access or trunk)
show mac address-table	Lists MAC table entries, including the associated VLAN

If possible, start this step with the **show vlan** and **show vlan brief** commands, because they list all the known VLANs and the access interfaces assigned to each VLAN. Be aware, however, that these two commands do not list operational trunks. The output does list all other interfaces (those not currently trunking), no matter whether the interface is in a working or nonworking state.

If the **show vlan** and **show interface switchport** commands are not available in a particular exam question, the **show mac address-table** command can also help identify the access VLAN. This command lists the MAC address table, with each entry including a MAC address, interface, and VLAN ID. If the exam question implies that a switch interface connects to a single device, you should only see one MAC table entry that lists that particular access interface; the VLAN ID listed for that same entry identifies the access VLAN. (You cannot make such assumptions for trunking interfaces.)

After you determine the access interfaces and associated VLANs, if the interface is assigned to the wrong VLAN, use the **switchport access vlan** *vlan-id* interface subcommand to assign the correct VLAN ID.

Access VLANs Undefined or Disabled

Switches do not forward frames for VLANs that are (a) not known because the VLAN is not configured or has not been learned with VTP or (b) the VLAN is known, but it is disabled (shut down). This section summarizes the best ways to confirm that a switch knows that a particular VLAN exists, and if it exists, determines the shutdown state of the VLAN.

First, on the issue of whether a VLAN exists on a switch, a VLAN can be defined to a switch in two ways: using the **vlan** *number* global configuration command, or it can be learned from another switch using VTP. For this discussion, consider that the only way for a switch to know about a VLAN is to have a **vlan** command configured on the local switch.

Next, the **show vlan** command always lists all VLANs known to the switch, but the **show running-config** command does not. Switches configured as VTP servers and clients do not list the **vlan** commands in the running-config file nor the startup-config file; on these

switches, you must use the **show vlan** command. Switches configured to use VTP transparent mode, or that disable VTP, list the **vlan** configuration commands in the configuration files. (Use the **show vtp status** command to learn the current VTP mode of a switch.)

After you determine that a VLAN does not exist on a switch, the problem might be that the VLAN simply needs to be configured.

Even for existing VLANs, you must also verify whether the VLAN is active. The **show vlan** command should list one of two VLAN state values, depending on the current state: either *active* or *act/lshut*. The second of these states means that the VLAN is shut down. Shutting down a VLAN disables the VLAN on that switch only, so that *the switch will not forward frames in that VLAN*.

Switch IOS gives you two similar configuration methods with which to disable (**shutdown**) and enable (**no shutdown**) a VLAN. Example P-6 shows how, first by using the global command **[no] shutdown vlan** *number* and then using the VLAN mode subcommand **[no] shutdown**. The example shows the global commands enabling and disabling VLANs 10 and 20, respectively, and using VLAN subcommands to enable and disable VLANs 30 and 40 (respectively).

Example P-6 *Enabling and Disabling VLANs on a Switch*

```
SW2# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                Fa0/14, Fa0/15, Fa0/16, Fa0/17
                                                Fa0/18, Fa0/19, Fa0/20, Fa0/21
                                                Fa0/22, Fa0/23, Fa0/24, Gi0/1
10   VLAN0010                         act/lshut Fa0/13
20   VLAN0020                         active
30   VLAN0030                         act/lshut
40   VLAN0040                         active
SW2# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
SW2(config)# no shutdown vlan 10
SW2(config)# shutdown vlan 20
SW2(config)# vlan 30
SW2(config-vlan)# no shutdown
SW2(config-vlan)# vlan 40
SW2(config-vlan)# shutdown
SW2(config-vlan)#
```

P

Mismatched Trunking Operational States

Trunking can be configured correctly so that both switches forward frames for the same set of VLANs. However, trunks can also be misconfigured, with a couple of different results. In some cases, both switches conclude that their interfaces do not trunk. In other cases, one switch believes that its interface is correctly trunking, while the other switch does not.

The most common incorrect configuration—which results in both switches not trunking—is a configuration that uses the **switchport mode dynamic auto** command on both switches on the link. The word "auto" just makes us all want to think that the link would trunk automatically, but this command is both automatic and passive. As a result, both switches passively wait on the other device on the link to begin negotiations.

With this particular incorrect configuration, the **show interfaces switchport** command on both switches confirms both the administrative state (auto) and the fact that both switches operate as "static access" ports. Example P-7 highlights those parts of the output from this command.

Example P-7 *Operational Trunking State*

```
SW2# show interfaces gigabit0/2 switchport
Name: Gi0/2
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
! lines omitted for brevity
```

A different incorrect trunking configuration results in one switch with an operational state of "trunk," while the other switch has an operational state of "static access." When this combination of events happens, the interface works a little. The status on each end will be up/up or connected. Traffic in the native VLAN will actually cross the link successfully. However, traffic in all the rest of the VLANs will not cross the link.

Figure P-6 shows the incorrect configuration along with which side trunks and which does not. The side that trunks (SW1 in this case) enables trunking always, using the command **switchport mode trunk**. However, this command does not disable Dynamic Trunking Protocol (DTP) negotiations. To cause this particular problem, SW1 also disables DTP negotiation using the **switchport nonegotiate** command. SW2's configuration also helps create the problem, by using a trunking option that relies on DTP. Because SW1 has disabled DTP, SW2's DTP negotiations fail, and SW2 does not trunk.

In this case, SW1 treats its G0/1 interface as a trunk, and SW2 treats its G0/2 interface as an access port (not a trunk). As shown in the figure at Step 1, SW1 could (for example) forward a frame in VLAN 10. However, SW2 would view any frame that arrives with an 802.1Q header as illegal, because SW2 treats its G0/2 port as an access port. So, SW2 discards any 802.1Q frames received on that port.

To deal with the possibility of this problem, always check the trunk's operational state on both sides of the trunk. The best commands to check trunking-related facts are **show interfaces trunk** and **show interfaces switchport**.

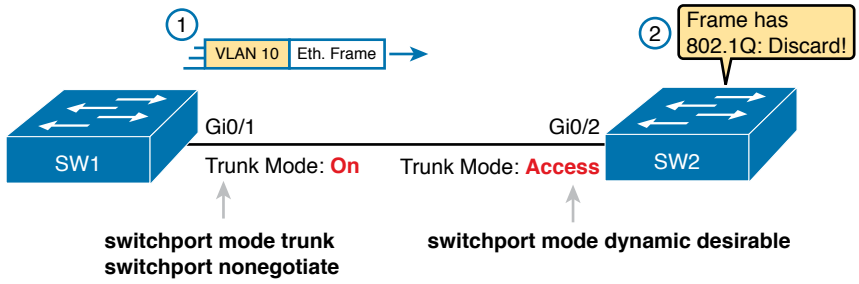


Figure P-6 *Mismatched Trunking Operational States*

NOTE Frankly, in real life, just avoid this kind of configuration. However, the switches do not prevent you from making these types of mistakes, so you need to be ready.

Mismatched Supported VLAN List on Trunks

VLAN trunks on Cisco switches can forward traffic for all defined and active VLANs. However, a particular trunk may not forward traffic for a defined and active VLAN for a variety of other reasons. You should know how to identify which VLANs a particular trunk port currently supports, and the reasons why the switch might not be forwarding frames for a VLAN on that trunk port.

The first category in this step can be easily done using the **show interfaces trunk** command, which only lists information about currently operational trunks. The best place to begin with this command is the last section of output, which lists the VLANs whose traffic will be forwarded over the trunk. Any VLANs that make it to this final list of VLANs in the command output meet the following criteria:

- The VLAN exists and is active on the local switch (as seen in the **show vlan** command).
- The VLAN has not been removed from the *allowed VLAN list* on the trunk (as configured with the **switchport trunk allowed vlan** interface subcommand).
- The VLAN has not been VTP-pruned from the trunk. (You can ignore this feature for the purposes of this book; it is mentioned here only because the **show** command mentions it.)
- The trunk is in an STP forwarding state in that VLAN (as also seen in the **show spanning-tree vlan** *vlan-id* command).

Example P-8 shows a sample of the command output from the **show interfaces trunk** command, with the final section of the command output shaded. In this case, the trunk only forwards traffic in VLANs 1 and 4.

Example P-8 *Allowed VLAN List and List of Active VLANs*

```
SW1# show interfaces trunk

Port        Mode         Encapsulation  Status        Native vlan
Gi0/1       desirable    802.1q         trunking      1

Port        Vlans allowed on trunk
Gi0/1       1-2,P-4094

Port        Vlans allowed and active in management domain
Gi0/1       1,4

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/1       1,4
```

The absence of a VLAN in this last part of the command's output does not necessarily mean that a problem has occurred. In fact, a VLAN might be legitimately excluded from a trunk for any of the reasons in the list just before Example P-8. However, for a given exam question, it can be useful to know why traffic for a VLAN will not be forwarded over a trunk, and the details inside the output identify the specific reasons.

The output of the **show interfaces trunk** command creates three separate lists of VLANs, each under a separate heading. These three lists show a progression of reasons why a VLAN is not forwarded over a trunk. Table P-2 summarizes the headings that precede each list and the reasons why a switch chooses to include or not include a VLAN in each list.

Key Topic

Table P-2 VLAN Lists in the **show interfaces trunk** Command

List Position	Heading	Reasons
First	VLANs allowed	VLANs 1–4094, minus those removed by the **switchport trunk allowed** command
Second	VLANs allowed and active…	The first list, minus VLANs not defined to the local switch (that is, there is not a **vlan** global configuration command or the switch has not learned of the VLAN with VTP), and also minus those VLANs in shutdown mode
Third	VLANs in spanning tree…	The second list, minus VLANs in an STP blocking state for that interface, and minus VLANs VTP pruned from that trunk

Mismatched Native VLAN on a Trunk

Closing with a brief mention of one other trunking topic, you should also check a trunk's native VLAN configuration at this step. Unfortunately, it is possible to set the native VLAN ID to different VLANs on either end of the trunk, using the **switchport trunk native vlan** *vlan-id* command. If the native VLANs differ according to the two neighboring switches, the switches will accidentally cause frames to leave one VLAN and enter another.

For example, if switch SW1 sends a frame using native VLAN 1 on an 802.1Q trunk, SW1 does not add a VLAN header, as is normal for the native VLAN. When switch SW2 receives the frame, noticing that no 802.1Q header exists, SW2 assumes that the frame is part of SW2's configured native VLAN. If SW2 has been configured to think VLAN 2 is the native VLAN on that trunk, SW2 will try to forward the received frame into VLAN 2.

This page intentionally left blank

APPENDIX Q

Troubleshooting IPv4 Routing Protocols

AUTHOR NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 11 of the book *CCNA Routing and Switching ICND2 200-105 Official Cert Guide*, published in 2016.

To troubleshoot a possible IPv4 routing protocol problem, first focus on interfaces, and then on neighbors. The routing protocol configuration identifies the interfaces on which the router should use the routing protocol. After identifying those interfaces, a network engineer can look at the neighbors each router finds on each interface, searching for neighbors that should exist but do not.

This chapter focuses on issues related to these two main branches of logic: on which interfaces should a router enable the routing protocol, and which neighbor relationships should each router create. This chapter's troubleshooting discussions emphasize how to find incorrect configuration problems by using only **show** and **debug** commands.

This chapter first briefly introduces a few broad concepts related to troubleshooting problems with routing protocols. The next major section examines problems related to which interfaces on which a router enables the routing protocol, with the final major section focusing of routing protocol neighbor relationships. Note that the entire chapter moves back and forth between discussing both Enhanced Interior Gateway Routing Protocol (EIGRP) and Open Shortest Path First Version 2 (OSPFv2).

Foundation Topics

Perspectives on Troubleshooting Routing Protocol Problems

Because a routing protocol's job is to fill a router's routing table with the currently best routes, it makes sense that troubleshooting potential problems with routing protocols could begin with the IP routing table. Given basic information about an internetwork, including the routers, their IP addresses and masks, and the routing protocol, you could calculate the subnet numbers that should be in the router's routing table and list the likely next-hop routers for each route. For example, Figure Q-1 shows an internetwork with six subnets. Router R1's routing table should list all six subnets, with three connected routes, two routes learned from R2 (172.16.4.0/24 and 172.16.5.0/24), and one route learned from R3 (172.16.6.0/24).

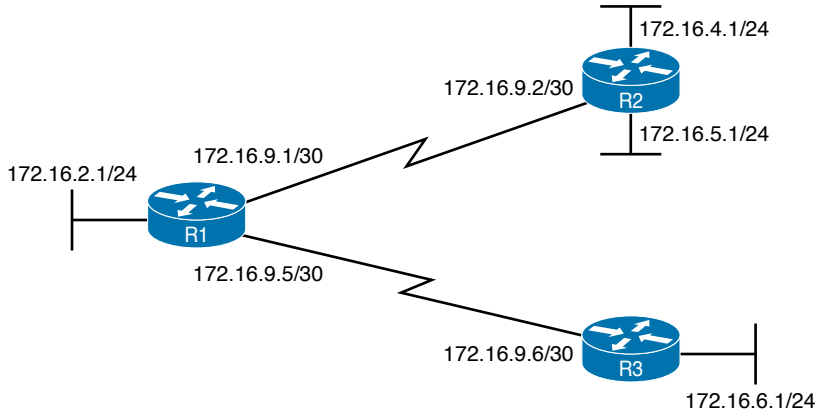


Figure Q-1 *Internetwork with Six Subnets*

So, one possible troubleshooting process is to analyze the internetwork, look at the routing table, and look for missing routes. If one or more expected routes are missing, the next step would be to determine whether that router has learned any routes from the expected next-hop (neighbor) router. The next steps to isolate the problem differ greatly if a router is having problems forming a neighbor relationship with another router, versus having a working neighbor relationship but not being able to learn all routes.

For example, suppose that R1 in Figure Q-1 has learned a route for subnet 172.16.4.0/24 in Figure Q-1 but not for subnet 172.16.5.0/24. In this case, it is clear that R1 has a working neighbor relationship with R2. In these cases, the root cause of this problem might still be related to the routing protocol, or it might not. For example, the problem may be that R2's lower LAN interface is down. However, if R1 did not have a route for both 172.16.4.0/24 and 172.16.5.0/24, R1's neighbor relationship with R2 could be the problem.

Troubleshooting routing protocol problems in real internetworks can be very complex—much more complex than even the most difficult CCNA R&S exam questions. Defining a generic troubleshooting process with which to attack both simple and complex routing protocol problems would require a lot of space and be counterproductive for preparing for

the CCNA exam. This chapter instead offers a straightforward process for attacking routing protocol problems—specifically, problems similar to the depth and complexity of the CCNA exam.

If an exam question appears to be related to a problem with a routing protocol, you can quickly identify some common configuration errors with the following process—even if the question does not list the configuration. The process has three main tasks:

Step 1. Examine the internetwork design to determine on which interfaces the routing protocol should be enabled and which routers are expected to become neighbors.

Step 2. Verify whether the routing protocol is enabled on each interface (as per Step 1). If it isn't, determine the root cause and fix the problem.

Step 3. Verify that each router has formed all expected neighbor relationships. If it hasn't, find the root cause and fix the problem.

For instance, as noted with asterisks in Figure Q-2, each router should enable the routing protocol on each of the interfaces shown in the figure. Also, routing protocol neighbor relationships should form between R1 and R2, and R1 and R3, but not between R2 and R3.

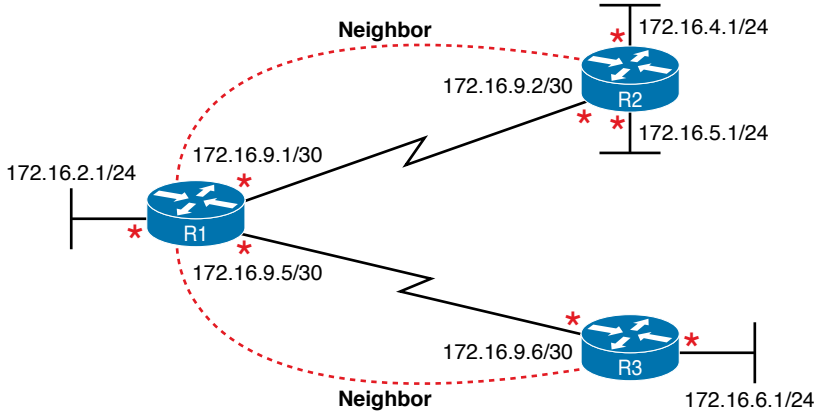


Figure Q-2 *Routing Protocol Interfaces and Neighbor Relationships*

While the concepts outlined in Figure Q-2 should be somewhat obvious by now, this chapter discusses how some of the most common configuration mistakes can impact the interfaces used by a routing protocol and whether a routing protocol creates neighbor relationships.

Interfaces Enabled with a Routing Protocol

This section examines the second major troubleshooting step outlined in the previous section of the chapter: how to verify the interfaces on which the routing protocol has been enabled. Both EIGRP and OSPF configuration enable the routing protocol on an interface by using the **network** router subcommand. For any interfaces matched by the **network** commands, the routing protocol tries the following two actions:

Key Topic

- Attempt to find potential neighbors on the subnet connected to the interface
- Advertise the subnet connected to that interface

At the same time, the **passive-interface** router subcommand can be configured so that the router does not attempt to find neighbors on the interface (the first action just listed), but still advertises the connected subnet (the second action).

Three **show** commands are all that is needed to know exactly which interfaces have been enabled with EIGRP and which interfaces are passive. In particular, the **show ip eigrp interfaces** command lists all EIGRP-enabled interfaces that are not passive interfaces. The **show ip protocols** command essentially lists the contents of the configured **network** commands for each routing protocol and a separate list of the passive interfaces. Comparing these two commands identifies all EIGRP-enabled interfaces and those that are passive.

For OSPF, the command works slightly differently, with the **show ip ospf interface brief** command listing all OSPF-enabled interfaces (including passive interfaces). Using this command, along with the list of passive interfaces listed by the **show ip protocols** command, again identifies all fully enabled OSPF interfaces as well as all passive interfaces.

Table Q-1 summarizes the commands that identify the interfaces on which OSPFv2 and EIGRP are enabled for easier reference.

Key Topic

Table Q-1 Key Commands to Find Routing Protocol-Enabled Interfaces

Command	Key Information	Lists Passive Interfaces?
show ip eigrp interfaces	Lists the interfaces on which EIGRP is enabled (based on the **network** commands), *excluding* passive interfaces.	No
show ip ospf interface brief	Lists the interfaces on which the OSPFv2 is enabled (based on the **network** router subcommands or **ip ospf** interface subcommands), *including* passive interfaces.	Yes
show ip protocols	Lists the contents of the **network** configuration commands for each routing process, and lists enabled but passive interfaces.	Yes

NOTE All the commands in Table Q-1 list the interfaces regardless of interface status, in effect telling you the results of the **network** and **passive-interface** configuration commands.

So, for the major troubleshooting step covered in this section, the task is to use the commands in Table Q-1 and analyze the output. First, an EIGRP example will be shown, followed by an OSPF example.

EIGRP Interface Troubleshooting

This section shows a few examples of the commands in the context of Figure Q-3, which is used in all the examples in this chapter.

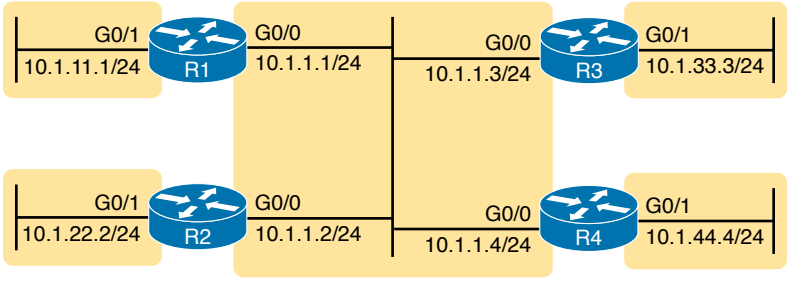


Figure Q-3 *Internetwork for EIGRP/OSPF Troubleshooting Examples*

This example includes four routers, with the following scenario in this case:

- R1 and R2 are configured correctly on both LAN interfaces.
- R3 is mistakenly not enabled with EIGRP on its G0/1 interface.
- R4 meant to use a **passive-interface G0/1** command because no other routers are off R4's G0/1 LAN. However, R4 has instead configured a **passive-interface G0/0** command.

This example begins by showing the working details between Routers R1 and R2, and then moves on to discuss the issues related to R3 and R4.

Examining Working EIGRP Interfaces

Examples Q-1 and Q-2 list configuration and **show** commands for R1 and R2, respectively. Each lists the related configuration, the **show ip eigrp interfaces** and **show ip protocols** command, and the EIGRP-learned routes on each router.

Example Q-1 *EIGRP Interfaces Problem: R1 Commands*

```
R1# show running-config
! only pertinent lines shown
router eigrp 99
 network 10.0.0.0
!
R1# show ip eigrp interfaces
EIGRP-IPv4 Interfaces for AS(99)
                   Xmit Queue   PeerQ        Mean   Pacing Time   Multicast    Pending
Interface    Peers  Un/Reliable  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Gi0/0          2        0/0         0/0          2       0/0           50           0
Gi0/1          0        0/0         0/0          0       0/0            0           0

R1# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "eigrp 99"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
```

```
  EIGRP-IPv4 Protocol for AS(99)
    Metric weight K1=1, K2=0, K3=1, K4=0, K5=0
    NSF-aware route hold timer is 240
    Router-ID: 1.1.1.1
    Topology : 0 (base)
      Active Timer: 3 min
      Distance: internal 90 external 170
      Maximum path: 4
      Maximum hopcount 100
      Maximum metric variance 1

  Automatic Summarization: disabled
  Maximum path: 4
  Routing for Networks:
    10.0.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    10.1.1.2              90      09:55:51
    10.1.1.3              90      00:02:00
  Distance: internal 90 external 170

R1# show ip route eigrp
! Legend omitted for brevity

      10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
D        10.1.22.0/24 [90/30720] via 10.1.1.2, 00:00:40, GigabitEthernet0/0
```

Example Q-2 *EIGRP Interfaces Problem: R2 Commands*

```
R2# show running-config
! only pertinent lines shown
router eigrp 99
 network 10.1.0.0 0.0.255.255

R2# show ip eigrp interfaces
EIGRP-IPv4 Interfaces for AS(99)
                   Xmit Queue   PeerQ        Mean   Pacing Time   Multicast    Pending
Interface   Peers  Un/Reliable  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Gi0/0           2        0/0        0/0          1       0/1           50           0
Gi0/1           0        0/0        0/0          0       0/0            0           0

R2# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "eigrp 99"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
```

```
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP-IPv4 Protocol for AS(99)
    Metric weight K1=1, K2=0, K3=1, K4=0, K5=0
    NSF-aware route hold timer is 240
    Router-ID: 2.2.2.2
    Topology : 0 (base)
      Active Timer: 3 min
      Distance: internal 90 external 170
      Maximum path: 4
      Maximum hopcount 100
      Maximum metric variance 1

  Automatic Summarization: disabled
  Maximum path: 4
  Routing for Networks:
    10.1.0.0/16
  Routing Information Sources:
    Gateway         Distance      Last Update
    10.1.1.3              90      00:02:30
    10.1.1.1              90      09:56:20
  Distance: internal 90 external 170

R2# show ip route eigrp
! Legend omitted for brevity
      10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
D        10.1.11.0/24 [90/30720] via 10.1.1.1, 00:03:25, GigabitEthernet0/0
```

The **show ip eigrp interfaces** command output on both R1 and R2 shows how both R1 and R2 have configured EIGRP using process ID 99, and that EIGRP has been enabled on both G0/0 and G0/1 on both these routers. This command lists only interfaces on which EIGRP has been enabled, excluding passive interfaces.

The highlighted parts of the **show ip protocols** command output on each router are particularly interesting. These sections show the parameters of the configured **network** commands. The **show ip protocols** command lists a separate line under the header "Routing for Networks," one for each configured **network** command. Example Q-1's output suggests R1 has a **network 10.0.0.0** configuration command (as shown at the beginning of the example), and Example Q-2's "10.1.0.0/16" suggests R2 has a **network 10.1.0.0 0.0.255.255** command.

Examining the Problems with EIGRP Interfaces

The next few pages now look at the problems caused by the configuration on Routers R3 and R4.

First, Example Q-2 gives brief insight into the current problem caused by R3. The end of R2's **show ip protocols** command (Example Q-2) lists two routing information sources: 10.1.1.1 (R1) and 10.1.1.3 (R3). However, R2 has learned only one EIGRP route (10.1.11.0/24), as shown in the **show ip route eigrp** command output. When working properly, R2 should learn three EIGRP routes—one for each of the other LAN subnets shown in Figure Q-3.

Example Q-3 shows the root cause on R3. First, R3's **show ip eigrp interfaces** command lists G0/0, but not G0/1, so a problem might exist with how EIGRP has been configured on G0/1. The configuration at the top of the example lists the root cause: an incorrect **network** command, which does not enable EIGRP on R3's G0/1 interface.

Example Q-3 *EIGRP Problems on R3*

```
R3# show running-config
! lines omitted for brevity
router eigrp 99
 network 10.1.1.3 0.0.0.0
 network 10.1.13.3 0.0.0.0
 auto-summary

R3# show ip eigrp interfaces
EIGRP-IPv4 Interfaces for AS(99)
                   Xmit Queue   PeerQ        Mean   Pacing Time   Multicast    Pending
Interface    Peers  Un/Reliable  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Gi0/0          2        0/0         0/0          1       0/1           50           0

R3# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "eigrp 99"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP-IPv4 Protocol for AS(99)
    Metric weight K1=1, K2=0, K3=1, K4=0, K5=0
    NSF-aware route hold timer is 240
    Router-ID: 3.3.3.3
    Topology : 0 (base)
      Active Timer: 3 min
      Distance: internal 90 external 170
      Maximum path: 4
      Maximum hopcount 100
      Maximum metric variance 1

  Automatic Summarization: disabled
  Maximum path: 4
  Routing for Networks:
    10.1.1.3/32
    10.1.13.3/32
  Routing Information Sources:
    Gateway         Distance      Last Update
    10.1.1.2              90      00:05:14
    10.1.1.1              90      00:05:14
  Distance: internal 90 external 170
```

The root cause of R3's problem is that R3 has a **network 10.1.13.3 0.0.0.0** configuration command, which does not match R3's 10.1.33.3 G0/1 IP address. If the configuration was not available in the exam question, the **show ip protocols** command could be used to essentially see the same configuration details. In this case, the **show ip protocols** command on R3 lists the text "10.1.13.3/32" as a reference to the contents of the incorrect **network** command's parameters, with "/32" translating to a wildcard mask of 32 binary 0s, or decimal 0.0.0.0.

R3's incorrect configuration means that two actions do not happen on R3's G0/1 interface. First, R3 does not try to find neighbors on its G0/1 interface, which is not a big deal in this case. However, R3 also does not advertise subnet 10.1.33.0/24, the connected subnet off R3's G0/1 interface.

Moving on to R4's problem, Example Q-4 shows why R1 and R2 do not learn R4's 10.1.44.0/24 subnet. In this case, on R4, the engineer could have correctly used a **passive-interface gigabitethernet0/1** router subcommand because no other routers should exist off R4's G0/1 interface. However, the engineer mistakenly made R4's G0/0 interface passive.

Example Q-4 *EIGRP Problems on R4*

```
R4# show running-config
! lines omitted for brevity
router eigrp 99
 passive-interface GigabitEthernet0/0
 network 10.0.0.0
 auto-summary

R4# show ip eigrp interfaces
EIGRP-IPv4 Interfaces for AS(99)
                   Xmit Queue   PeerQ        Mean   Pacing Time   Multicast    Pending
Interface    Peers  Un/Reliable  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Gi0/1          0        0/0          0/0         0       0/1             0           0

R4# show ip protocols | begin Routing for Networks
  Routing for Networks:
    10.0.0.0
  Passive Interface(s):
    GigabitEthernet0/0
  Routing Information Sources:
    Gateway         Distance      Last Update
  Distance: internal 90 external 170
```

NOTE The last command on the example, **show ip protocols | begin Routing for Networks**, lists the command output, but starting with the line with the literal case-sensitive string **Routing for Networks**. You can use this feature with any output from a command when you prefer to view only later lines of the command's output.

To find this mistake without the configuration, Example Q-4 lists two useful commands. R4's **show ip eigrp interfaces** command omits the (G0/0) passive interface, which means that R4 will not attempt to find EIGRP neighbors off that interface. Also, the highlighted part of R4's **show ip protocols** command output lists G0/0 as a passive interface, which again means that R4 does not even attempt to become neighbors with others off its G0/0 interface.

OSPF Interface Troubleshooting

OSPF has the same basic requirements as EIGRP for interfaces, with a few exceptions. First, EIGRP routers need to use the same autonomous system number (ASN) as their neighboring routers, as configured in the **router eigrp** *asn* global configuration command. OSPF routers can use any process ID on the **router ospf** *process-id* command, with no need to match their neighbors. Second, OSPF requires that the interfaces connected to the same subnet be assigned to the same OSPF area, whereas EIGRP has no concept of areas.

Example Q-5 shows a mostly working OSPF internetwork, again based on Figure Q-3. The problem in this case relates to the area design, as shown in Figure Q-4, the revised version of Figure Q-3. All subnets should be placed into area 0. However, the engineer made a configuration mistake on R2, putting both its interfaces into area 1. As a result, R2's G0/0 interface breaks the OSPF design rule of being in the same subnet as R1, R3, and R4, but not being in the same OSPF area.

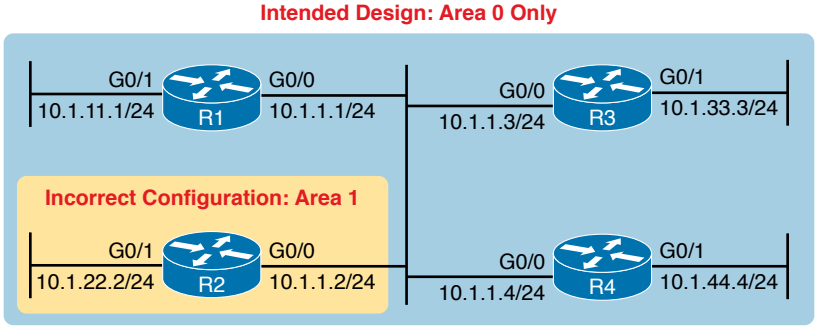


Figure Q-4 *Intended Area Design Using Only Area 0, with R2 Breaking the Design*

Example Q-5 begins to break down the problem by looking at the status of OSPF on the router interfaces of R1 and R2, using the **show ip ospf interface brief** command.

Example Q-5 **show ip interface brief** *on R1 and R2*

```
R1> show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1     0               10.1.11.1/24       1     DR    0/0
Gi0/0        1     0               10.1.1.1/24        1     DROTH 2/2

! The following command is from R2
R2> show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        2     1               10.1.22.2/24       1     WAIT  0/0
Gi0/0        2     1               10.1.1.2/24        1     WAIT  0/0
```

From a general perspective, the **show ip ospf interface brief** command lists output similar to the **show ip eigrp interface** command, with one line for each enabled interface. The **show ip ospf interface** command, not shown in the example, lists detailed OSPF information for each interface.

Specific to this problem, the output in Example Q-5 shows that R1 and R2 both have OSPF enabled on both LAN interfaces. However, this command also lists the area number for each interface, with R2 having both LAN interfaces in area 1. Also, these commands repeat the IP address and mask of the interfaces, so together, you can see that R1's 10.1.1.1/24 address is in the same subnet as R2's 10.1.1.2/24 address, putting these two routers in the same subnet but in different OSPF areas.

Example Q-6 shows another way to look at the problem, with the **show ip protocols** commands on both R1 and R2. Because this command lists the OSPF **network** commands in shorthand form, it can point toward a possible configuration error, even if the configuration is not available.

Example Q-6 *Finding OSPF Configuration Errors with* **show ip protocols** *R1 and R2*

```
R1> show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "ospf 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 1.1.1.1
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
    10.0.0.0 0.255.255.255 area 0
  Routing Information Sources:
    Gateway         Distance      Last Update
    2.2.2.2              110      00:14:32
    3.3.3.3              110      00:14:32
    10.1.44.4            110      00:14:42
  Distance: (default is 110)

R1> show ip route ospf
! Legend omitted for brevity

      10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O        10.1.33.0/24 [110/2] via 10.1.1.3, 00:15:32, GigabitEthernet0/0
O        10.1.44.0/24 [110/2] via 10.1.1.4, 00:15:42, GigabitEthernet0/0
```

```
! Now moving to Router R2

R2> show ip protocols
*** IP Routing is NSF aware ***
```

```
Routing Protocol is "ospf 2"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 2.2.2.2
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
    10.0.0.0 0.255.255.255 area 1
  Routing Information Sources:
    Gateway         Distance      Last Update
  Distance: (default is 110)

R2>
Nov 15 12:16:39.377: %OSPF-4-ERRRCV: Received invalid packet: mismatched area
ID, from backbone area must be virtual-link but not found from 10.1.1.1,
GigabitEthernet0/0
```

Interestingly, a closer look at R2's **show ip protocols** command output, particularly the highlighted portion, points out the configuration error. As usual, the section with the heading "Routing for Networks:" points to a shorthand version of the configuration. In this case, the highlighted phrase "10.0.0.0 0.255.255.255 area 1" is actually the exact syntax of the one **network** command on Router R2, minus the word *network*, or **network 10.0.0.0 0.255.255.255 area 1**. Because Figure Q-4 shows the design should put all interfaces in area 0, reconfiguring this command to instead be **network 10.0.0.0 0.255.255.255 area 0** would solve this particular problem.

The end of the example also shows an unsolicited log message generated by Router R2, notifying the console user that this router has received a Hello from a router in a different area.

As you check the interfaces, you could also check several other details. It makes sense to go ahead and check the interface IP addresses, masks, and interface status values by using the **show interfaces** and **show ip interface brief** commands. In particular, it is helpful to note which interfaces are up/up, because a router will send no packets (including routing protocol packets) out interfaces that are not in an up/up state.

Neighbor Relationships

This final major section of the chapter examines the large number of facts that each router must check with each potential neighbor before the two routers become neighbors.

At a very basic level, routing protocols can easily create neighbor relationships using a Hello protocol. First, the routing protocol must be enabled on an interface. In addition, the interface may not be configured as a passive interface, because that stops the routing protocol from sending the Hello messages.

Beyond this basic process, the routing protocols actually check several other parameters to find out whether the routers should become neighbors. Both OSPF and EIGRP use Hello messages, and these messages each list information used to perform some basic verification

checks. For example, as just shown in earlier Example Q-5, an OSPF router should not become neighbors with another router in another area because all routers on a common subnet should be in the same OSPF area by design.

After an EIGRP or OSPF router hears a Hello from a new neighbor, the routing protocol examines the information in the Hello, and compares that information with the local router's own settings. If the settings match, great. If not, the routers do not become neighbors. Because there is no formal term for all these items that a routing protocol considers, this book just calls them *neighbor requirements*.

Table Q-2 lists the neighbor requirements for both EIGRP and OSPF. Following the table, the next few pages examine some of these settings for both EIGRP and OSPF, again using examples based on Figure Q-3.

NOTE Even though it is important to study and remember the items in this table, when reading this chapter the first time, just keep reading. When later reviewing the chapter or part, make sure you remember the details in the table.

Key Topic

Table Q-2 Neighbor Requirements for EIGRP and OSPF

Requirement	EIGRP	OSPF
Interfaces must be in an up/up state.	Yes	Yes
Interfaces must be in the same subnet.	Yes	Yes
Access control lists (ACL) must not filter routing protocol messages.	Yes	Yes
Must pass routing protocol neighbor authentication (if configured).	Yes	Yes
Must use the same ASN/PID on the **router** configuration command.	Yes	No
Hello and hold/dead timers must match.	No	Yes
Router IDs (RID) must be unique.	No[1]	Yes
K-values must match.	Yes	N/A
Must be in the same area.	N/A	Yes

[1] Having duplicate EIGRP RIDs does not prevent routers from becoming neighbors, but it can cause problems when external EIGRP routes are added to the routing table.

Q

Unlike most of the neighbor requirements listed in Table Q-2, the first three requirements have very little to do with the routing protocols themselves. The two routers must be able to send packets to each other over the physical network to which they are both connected. To do that, the router interfaces must be up/up, and they must be in the same subnet. In addition, the routers must not be using an ACL that filters the routing protocol traffic.

For instance, OSPF sends many messages to the well-known multicast IP addresses 224.0.0.5 and 224.0.0.6, whereas EIGRP uses 224.0.0.10. An ACL command like **access-list 101 deny ip any host 224.0.0.10**, in an inbound ACL on a router interface, would filter incoming EIGRP packets. Or, an ACL command like **access-list 102 deny ospf any any** could filter all OSPF traffic. Even more difficult to notice is an ACL that has lots of **permit** commands that match different TCP and UDP port numbers, but does not match the routing protocol explicitly, so the routing protocol packets match the implicit deny any at the end of the ACL. So, take extra care to watch for ACLs, especially when it seems like all the routing protocol configuration looks good.

In practice, before examining the rest of the details of why two routers do not become neighbors, confirm that the two routers can ping each other on the local subnet. If the ping fails, investigate all the Layer 1, 2, and 3 issues that could prevent the ping from working (such as an interface not being up/up).

Now, on to the specific discussions about EIGRP and OSPF. Because the details differ slightly between the two routing protocols, this section first examines EIGRP, followed by OSPF.

NOTE This section assumes that the routing protocol has actually been enabled on each required interface, as covered earlier in this chapter in the "Interfaces Enabled with a Routing Protocol" section.

EIGRP Neighbor Verification Checks

Any two EIGRP routers that connect to the same data link, and whose interfaces have been enabled for EIGRP and are not passive, will at least consider becoming neighbors. To quickly and definitively know which potential neighbors have passed all the neighbor requirements for EIGRP, just look at the output of the **show ip eigrp neighbors** command. This command lists only neighbors that have passed all the neighbor verification checks.

Example Q-7 shows an example of the **show ip eigrp neighbors** command, with the four routers from Figure Q-3 again. In this case, all the routers have been configured correctly, so each has a neighbor relationship with the other three routers on the same LAN subnet.

Example Q-7 *R1* **show ip eigrp neighbors** *Command with All Problems Fixed*

```
R1# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(99)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
1   10.1.1.3                Gi0/0                    13 00:00:20    1   100  0  31
2   10.1.1.4                Gi0/0                    13 00:00:43   80   480  0  10
0   10.1.1.2                Gi0/0                    13 00:13:52    1   100  0  20
```

If the **show ip eigrp neighbors** command does not list one or more expected neighbors, the first problem isolation step should be to find out if the two routers can ping each other's IP addresses on the same subnet. If that works, start looking at the list of neighbor verification checks, as relisted for EIGRP here in Table Q-3. Table Q-3 summarizes the EIGRP neighbor requirements, while noting the best commands with which to determine which requirement is the root cause of the problem.

Key Topic

Table Q-3 EIGRP Neighbor Requirements and the Best **show/debug** Commands

Requirement	Best Commands to Isolate the Problem
Must be in the same subnet.	**show interfaces**, **show ip interface**
Must use the same ASN on the **router** configuration command.	**show ip eigrp interfaces**, **show ip protocols**
Must pass EIGRP neighbor authentication.	**debug eigrp packets**
K-values must match.	**show ip protocols**

Of the four rows of requirements listed in Table Q-3, the first two have already been discussed in this chapter, and do not need further discussion.

For EIGRP authentication (the third item in the table), EIGRP supports the capability for routers to trust routers as EIGRP neighbors only if the routers share the same security key (password); if that check fails, the neighbor relationship fails. By default, routers do not attempt EIGRP authentication, which allows the routers to form EIGRP neighbor relationships. If one router uses authentication, and the other does not, they will not become neighbors. If both use authentication, they must use the same authentication key to become neighbors.

The last item in the table, EIGRP K-values, refers to the EIGRP metric components and the metric calculation. These K-values are variables that basically enable or disable the use of the different components in the EIGRP composite metric. Cisco recommends leaving these values at their default settings, using only bandwidth and delay in the metric calculation. The K-value settings must match before two routers will become neighbors; you can check the K-values on both routers with the **show ip protocols** command.

EIGRP Neighbor Troubleshooting Example

Example Q-8 shows three problems that can cause EIGRP routers to fail to become neighbors. This example uses the usual design for this chapter, as repeated in Figure Q-5. The figure shows the same routers, and same interfaces, but with the following problems:

- R2 has been configured with IP address 10.1.2.2/24 in a different subnet than R1, R3, and R4.
- R3 has been configured to use ASN 199 with the **router eigrp 199** command instead of ASN 99, as used on the other three routers.
- R4 has been configured to use message digest 5 (MD5) authentication, whereas the other routers use no authentication.

R1 can actually detect two of the problems using local commands and messages, as shown in Example Q-8. R1 generates an unsolicited log message for the mismatched subnet problem, and a **debug** command on R1 can reveal the authentication failure. The example shows some running commentary inside the example.

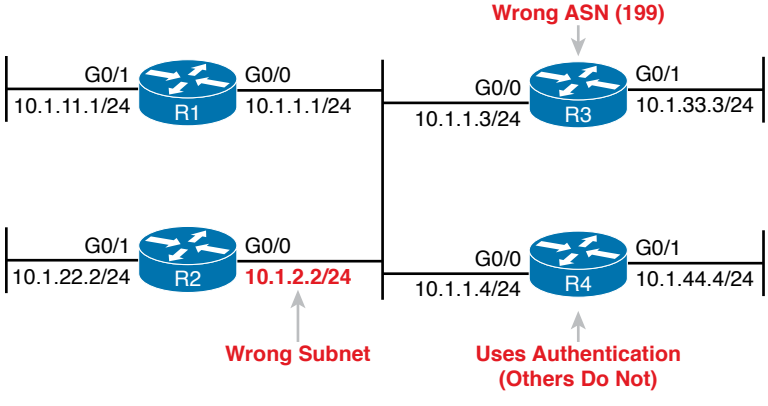


Figure Q-5 *Summary of Problems That Prevent EIGRP Neighbors on the Central LAN*

Example Q-8 *Common Problems Preventing the Formation of EIGRP Neighbors (R1)*

```
! First, R1 has no neighbor relationships yet. R1 uses ASN (process) 99.
R1# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(99)

R1#
! Next, R1 generates a log message, which shows up at the console, stating
! that the router with IP address 10.1.2.2 is not on the same subnet as R1.
!
*Nov 15 16:19:14.740: %DUAL-6-NBRINFO: EIGRP-IPv4 99: Neighbor 10.1.2.2
(GigabitEthernet0/0) is blocked: not on common subnet (10.1.1.1/24)

! Next, R1 enables a debug that shows messages for each packet received from R4,
! which uses the wrong password (authentication key string)
!
R1# debug eigrp packets
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
     SIAREPLY)
R1#

*Nov 15 16:20:30.865: EIGRP: Gi0/0: ignored packet from 10.1.1.4, opcode = 5
  (authentication off or key-chain missing)
```

Example Q-8 shows some evidence of the mismatched subnet with R2, and the invalid authentication problem with R4. Even without knowing the details, it is easy to imagine that if one router's EIGRP process uses authentication with a defined password, and the other does not, that authentication will fail. The result? Neighbor relationships do not form.

Example Q-8 shows details about two of the problems, but not any details about the incorrect ASN configured on R3. Example Q-9 shows those details by listing excerpts from two **show** commands on R3, both of which identify the ASN configured on that router. By using these same commands on all the routers, you could note that R1, R2, and R4 use ASN 99, whereas R3 uses 199, as shown in Example Q-9.

Example Q-9 *Displaying the Incorrect ASN (199) on R3*

```
R3# show ip protocols
Routing Protocol is "eigrp 199"
!
! The first line of output from show ip eigrp interfaces lists ASN 199
!
R3# show ip eigrp interfaces
EIGRP-IPv4 Interfaces for AS(199)
                        Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Gi0/0              0        0/0         0       0/1            0           0
Gi0/1              0        0/0         0       0/1            0           0
```

OSPF Neighbor Troubleshooting

Similar to EIGRP, a router's **show ip ospf neighbor** command lists all the neighboring routers that have met all the requirements to become an OSPF neighbor as listed in Table Q-2. So, the first step in troubleshooting OSPF neighbors is to look at the list of neighbors.

Example Q-10 lists the output of a **show ip ospf neighbor** command on Router R2, from Figure Q-4. All four routers sit on the same LAN subnet, in area 0, with correct configurations, so all four routers form a valid OSPF neighbor relationship.

Example Q-10 *Normal Working* **show ip ospf neighbors** *Command on Router R2*

```
R2# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           1   FULL/BDR        00:00:37    10.1.1.1        GigabitEthernet0/0
3.3.3.3           1   2WAY/DROTHER    00:00:37    10.1.1.3        GigabitEthernet0/0
4.4.4.4           1   FULL/DR         00:00:31    10.1.1.4        GigabitEthernet0/0
```

First, note that the neighbor IDs, listed in the first column, identify neighbors by their router ID (RID). For this example network, all four routers use an easily guessed RID. Further to the right, the Address column lists the interface IP address used by that neighbor on the common subnet.

A brief review of OSPF neighbor states can help you understand a few of the subtleties of the output in the example. A router's listed status for each of its OSPF neighbors—the neighbor's state—should settle into either a 2-way or full state under normal operation. For neighbors that do not need to directly exchange their databases, typically two non-designated router (DR) routers on a LAN, the routers should settle into a 2-way neighbor state. In most cases, two neighboring routers need to directly exchange their full link-state databases (LSDB) with each other. As soon as that process has been completed, the two routers settle into a full neighbor state.

In Example Q-10, Router R4 is the DR, and R1 is the backup DR (BDR), so R2 and R3 (as non-DRs) do not need to directly exchange routes. Therefore, R2's neighbor state for R3 (RID 3.3.3.3) in Example Q-10 is listed as 2-way.

NOTE Notably, OSPF neighbors do not have to use the same process ID on the **router ospf** *process-id* command to become neighbors. In Example Q-10, all four routers use different PIDs.

If the **show ip ospf neighbor** command does not list one or more expected neighbors, you should confirm, even before moving on to look at OSPF neighbor requirements, that the two routers can ping each other on the local subnet. But if the two neighboring routers can ping each other, and the two routers still do not become OSPF neighbors, the next step is to examine each of the OSPF neighbor requirements. Table Q-4 summarizes the requirements, listing the most useful commands with which to find the answers.

Key Topic

Table Q-4 OSPF Neighbor Requirements and the Best **show/debug** Commands

Requirement	Best show Command	Best debug Command
Must be in the same subnet.	**show interfaces**	**debug ip ospf hello**
Hello and dead timers must match.	**show ip ospf interface**	**debug ip ospf hello**
Must be in the same area.	**show ip ospf interface brief**	**debug ip ospf adj**
RIDs must be unique.	**show ip ospf**	(N/A; log messages identify this problem)
Must pass any neighbor authentication.	**show ip ospf interface**	**debug ip ospf adj**

This topic looks at a couple of OSPF neighbor problems using the usual four-router network from Figure Q-4, with all interfaces in area 0. However, the following problems have been introduced into the design:

- R2 has been configured with both LAN interfaces in area 1, whereas the other three routers' G0/0 interfaces are assigned to area 0.
- R3 is using the same RID (1.1.1.1) as R1.
- R4 has been configured with a Hello/dead timer of 5/20 on its G0/0 interface, instead of the 10/40 used (by default) on R1, R2, and R3.

Figure Q-6 shows these same problems for reference.

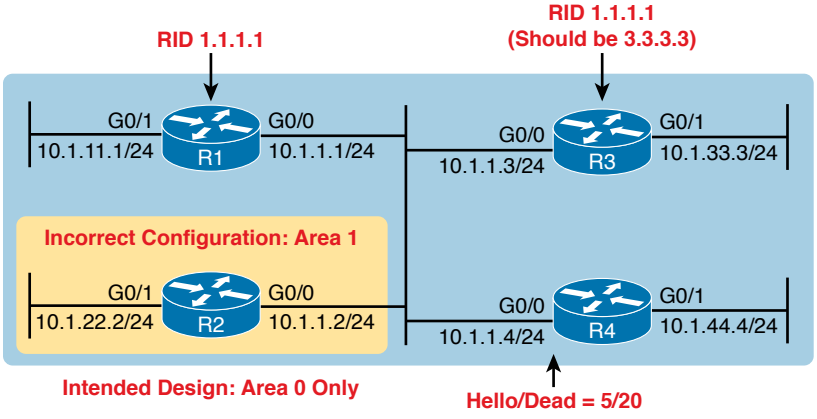


Figure Q-6 *Summary of Problems That Prevent OSPF Neighbors on the Central LAN*

Finding Area Mismatches

Earlier in this chapter, the "OSPF Interface Troubleshooting" section showed how to use the **show ip ospf interface** command to list the area numbers and find OSPF area mismatches. This next topic shows how to see that same issue using the **debug ip ospf adj** command, as shown in Example Q-11. This command lists messages related to OSPF neighbor adjacency events, and shows messages that identify the area mismatch (with R2).

Example Q-11 *Finding Mismatched Area Problem with R1* **debug**

```
R1# debug ip ospf adj
OSPF adjacency events debugging is on
R1#
*Nov 15 13:42:02.288: OSPF-1 ADJ   Gi0/0: Rcv pkt from 10.1.1.2, area 0.0.0.0,
  mismatched area 0.0.0.1 in the header
R1#
R1# undebug all
All possible debugging has been turned off
```

As noted in Table Q-4, the **debug ip ospf adj** command helps troubleshoot mismatched OSPF area problems. The first part of the highlighted message in the example lists shorthand about a received packet ("Rcv pkt") from 10.1.1.2, which is R2's IP address. The rest of the message mentions R1's area (0.0.0.0), and the area claimed by the other router (0.0.0.1). (Note that the message lists the 32-bit area number as a dotted-decimal number.)

This particular example focuses on the symptom (that a neighbor relationship does not start), and the debug messages that identify the problem (mismatched areas). However, finding the configuration error may take some work, because the problem could be more complex than just having the wrong area number configured on a command.

One harder-to-notice configuration error happens when the configuration has multiple **network** commands, with different area numbers, that all happen to match one interface's IP address. IOS stores the OSPF **network** commands to the configuration in the same order they are configured (which is the same order listed in the output of **show running-config**). IOS processes the commands in sequence, so that the first **network** command that matches a particular interface is used to set the OSPF area number.

For instance, imagine a router with interface G0/1 configured with IP address 1.1.1.1. The OSPF configuration lists the following two **network** commands, in that order. Both would match the interface IP address of 1.1.1.1, so IOS uses the first command, which lists area 1. IOS would not use the second command, even though it uses a wildcard mask that is more specific.

- **network 1.0.0.0 0.255.255.255 area 1**
- **network 1.1.1.1 0.0.0.0 area 0**

Another tricky configuration error that can result in an area mismatch occurs when configuring both the **network** OSPF subcommand and the **ip ospf** interface subcommand on the same router. IOS supports using both on the same router at the same time. However, IOS does not prevent a case in which a **network** command attempts to enable OSPF in one area, and the **ip ospf** interface subcommand attempts to enable OSPF in a different area. When that happens, IOS uses the area number defined in the **ip ospf** interface subcommand.

For instance, with the two **network** commands just listed, if the **ip ospf 1 area 5** command was configured on that router's interface, that interface would be in area 5; IOS would prefer that setting over any OSPF **network** command.

NOTE Using both **network** router subcommands and **ip ospf** interface subcommands allows an easier migration from the older to newer style OSPF configuration. However, most enterprises today would use either **network** commands or **ip ospf** commands in one router.

Finding Duplicate OSPF Router IDs

Next, Example Q-12 shows R1 and R3 both trying to use RID 1.1.1.1. Interestingly, both routers automatically generate a log message for the duplicate OSPF RID problem between R1 and R3; the end of Example Q-12 shows one such message. For the exams, just use the **show ip ospf** commands on both R3 and R1 to easily list the RID on each router, noting that they both use the same value.

Example Q-12 *Comparing OSPF Router IDs on R1 and R3*

```
! Next, on R3: R3 lists the RID of 1.1.1.1
!
R3# show ip ospf
 Routing Process "ospf 3" with ID 1.1.1.1
 Start time: 00:00:37.136, Time elapsed: 02:20:37.200
! lines omitted for brevity
```

```
! Back to R1: R1 also uses RID 1.1.1.1

R1# show ip ospf
 Routing Process "ospf 1" with ID 1.1.1.1
 Start time: 00:01:51.864, Time elapsed: 12:13:50.904
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Supports NSSA (compatible with RFC 3101)
 Event-log enabled, Maximum number of events: 1000, Mode: cyclic
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 0. Checksum Sum 0x000000
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
```

```
 Number of areas transit capable is 0
 External flood list length 0
 IETF NSF helper support enabled
 Cisco NSF helper support enabled
 Reference bandwidth unit is 100 mbps
    Area BACKBONE(0) (Inactive)
        Number of interfaces in this area is 3
        Area has no authentication
        SPF algorithm last executed 00:52:42.956 ago
        SPF algorithm executed 9 times
        Area ranges are
        Number of LSA 1. Checksum Sum 0x00C728
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0

*May 29 00:01:25.679: %OSPF-4-DUP_RTRID_NBR: OSPF detected duplicate router-id
1.1.1.1 from 10.1.1.3 on interface GigabitEthernet0/0
```

First, focus on the problem: the duplicate RIDs. The first line of the **show ip ospf** command on the two routers quickly shows the duplicate use of 1.1.1.1. To solve the problem, assuming R1 should use 1.1.1.1 and R3 should use another RID (maybe 3.3.3.3), change the RID on R3, and restart the OSPF process. To do so, use the **router-id 3.3.3.3** OSPF subcommand and use the EXEC mode command **clear ip ospf process**.

Also, take a moment to read over the log message generated on each router when a duplicate RID exists.

Finally, note that the **show ip ospf** commands in Example Q-12 also show a common false positive for a root cause of OSPF neighbor problems. OSPF PIDs—the number of the **router ospf** command—do not have to match. Note that in Example Q-12 that same first line of output shows that R3 uses the **router ospf 3** command, per the phrase "Process ospf 3," whereas R1 uses the **router ospf 1** command, as noted with the phrase "Process ospf 1." These mismatched numbers are not a problem.

Finding OSPF Hello and Dead Timer Mismatches

Finally, consider the problem created on R4, with the configuration of a different Hello timer and dead timer as compared with the default settings on R1, R2, and R3. Whereas EIGRP allows neighbors to use a different Hello timer, OSPF does not, so this mismatch prevents R4 from becoming neighbors with any of the other three OSPF routers.

Example Q-13 shows the easiest way to find the mismatch, using the **show ip ospf interface** command on both R1 and R4. This command lists the Hello and dead timers for each interface, as highlighted in the example. Note that R1 uses 10 and 40 (Hello and dead), whereas R4 uses 5 and 20.

Example Q-13 *Finding Mismatched Hello/Dead Timers*

```
R1# show ip ospf interface G0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0, Attached via Network Statement
  Process ID 1, Router ID 1.1.1.1, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 1.1.1.1, Interface address 10.1.1.1
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
! lines omitted for brevity
```

```
! Moving on to R4 next
!
R4# show ip ospf interface Gi0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.1.4/24, Area 0, Attached via Network Statement
  Process ID 4, Router ID 10.1.44.4, Network Type BROADCAST, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.1.44.4, Interface address 10.1.1.4
  No backup designated router on this network
  Timer intervals configured, Hello 5, Dead 20, Wait 20, Retransmit 5
! lines omitted for brevity
```

The **debug ip ospf hello** command can also uncover this problem because it lists a message for each Hello that reveals the Hello/dead timer mismatch, as shown in Example Q-14.

Example Q-14 *Finding Mismatched Hello/Dead Timers with* **debug**

```
R1# debug ip ospf hello
OSPF hello events debugging is on
R1#
*Nov 15 14:05:10.616: OSPF-1 HELLO Gi0/0: Rcv hello from 10.1.44.4 area 0 10.1.1.4
*Nov 15 14:05:10.616: OSPF-1 HELLO Gi0/0: Mismatched hello parameters from 10.1.1.4
*Nov 15 14:05:10.616: OSPF-1 HELLO Gi0/0: Dead R 20 C 40, Hello R 5 C 10 Mask R
  255.255.255.0 C 255.255.255.0
```

Although debug messages can be a little difficult to understand, a few comments make the meaning of these messages much clearer. The highlighted message uses a *C* to mean "configured value"—in other words, the value on the local router, or R1 in this case. The *R* in the message means "received value," or the value listed in the received Hello. In this case

- "Dead R 20 C 40" means that R1 received a Hello with a dead timer set to 20, while R1's configured value is set to 40.

- "Hello R 5 C 10" means that R1 received a Hello with the Hello timer set to 5, while R1's configured value is set to 10.

Note that any IP subnet mismatch problems could also be found with this same debug, based on the received and configured subnet masks.

Other OSPF Issues

This last short discussion in this chapter looks at these two additional topics: shutting down the routing protocol process and the interface maximum transmission unit (MTU) size.

Shutting Down the OSPF Process

Cisco uses the IOS **shutdown** command in several contexts. You can use the **shutdown** command in interface configuration mode to disable the interface so that it no longer sends and receives packets. Cisco IOS switches allow the **shutdown** command in VLAN configuration mode, causing the switch to stop forwarding frames in that VLAN. In both cases, the **shutdown** command does not remove any configuration; it simply causes IOS to stop a particular function. Then, the **no shutdown** command in the same command mode re-enables that function.

IOS allows both the OSPFv2 and EIGRP routing protocol processes to be disabled and enabled with the **shutdown** and **no shutdown** commands, respectively, in routing protocol configuration mode. When a routing protocol process is shut down, IOS

- Brings down any existing neighbor relationships
- Does not form new neighbor relationships
- Quits sending Hello messages
- Does not remove routing protocol configuration

Basically, shutting down the routing protocol process gives the network engineer a way to stop using the routing protocol on that router, without having to remove all the configuration.

From a troubleshooting perspective, on the exam, what would you expect to see if a small design was configured perfectly, except that one router's OSPF process was shut down? First, the router with the shutdown routing protocol process would not have any OSPF neighbors, and other routers would not list that router as a neighbor. But because the OSPF **shutdown** subcommand does not remove any configuration, the **show ip ospf interfaces** command still shows evidence that OSPF is configured on the interfaces.

Example Q-15 shows an example on Router R5, as shown in Figure Q-7. R5 is a different router than the one used in earlier examples, but it begins the example with two OSPF neighbors, R2 and R3, with router IDs 2.2.2.2 and 3.3.3.3. The example shows the OSPF process being shut down, the neighbors failing, and those two key OSPF **show** commands: **show ip ospf neighbor** and **show ip ospf interface brief**.

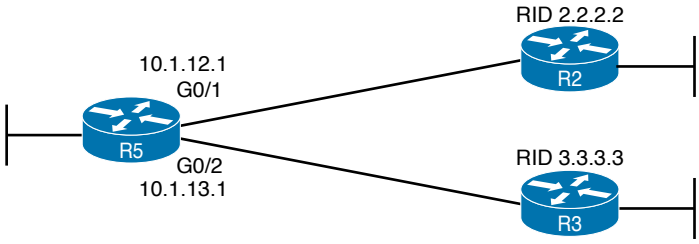


Figure Q-7 *Example Network to Demonstrate OSPF Process Shutdown*

Example Q-15 *Shutting Down an OSPF Process, and the Resulting Neighbor States*

```
R5# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   FULL/DR         00:00:35    10.1.12.2       GigabitEthernet0/1
3.3.3.3           1   FULL/DR         00:00:33    10.1.13.3       GigabitEthernet0/2
R5# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R5(config)# router ospf 1
R5(config-router)# shutdown
R5(config-router)# ^Z
R5#
*Mar 23 12:43:30.634: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on GigabitEthernet0/1
  from FULL to DOWN, Neighbor Down: Interface down or detached
*Mar 23 12:43:30.635: %OSPF-5-ADJCHG: Process 1, Nbr 3.3.3.3 on GigabitEthernet0/2
  from FULL to DOWN, Neighbor Down: Interface down or detached
R5#
R5# show ip ospf neighbor
R5#
R5# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Gi0/1        1     0               10.1.12.1/24       1     DOWN  0/0
Gi0/2        1     0               10.1.13.1/24       1     DOWN  0/0
```

The two **show** commands point out a couple of particularly important facts. First, before the **shutdown**, the **show ip ospf neighbor** command lists two neighbors. After the **shutdown**, the same command lists no neighbors at all. Second, the **show ip ospf interface brief** command does list the interfaces on which OSPF is enabled, on the local router's own IP addresses. However, it lists a state of DOWN, which is a reference to the neighbor's state.

Mismatched MTU Settings

The MTU size defines a per-interface setting used by the router for its Layer 3 forwarding logic, defining the largest network layer packet that the router will forward out each interface. For instance, the IPv4 MTU size of an interface defines the maximum size IPv4 packet that the router can forward out an interface.

Routers often use a default MTU size of 1500 bytes, with the ability to set the value as well. The **ip mtu** *size* interface subcommand defines the IPv4 MTU setting, and the **ipv6 mtu** *size* command sets the equivalent for IPv6 packets.

In an odd twist, two OSPFv2 routers can actually become OSPF neighbors, and reach 2-way state, even if they happen to use different IPv4 MTU settings on their interfaces. However, they fail to exchange their LSDBs. Eventually, after trying and failing to exchange their LSDBs, the neighbor relationship also fails.

The concepts behind what happens with an MTU mismatch work the same with both OSPFv2 and OSPFv3.

Command References

Tables Q-5, Q-6, and Q-7 list configuration, verification, and debug commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table Q-5 Appendix Q Configuration Command Reference

Command	Description
ip hello-interval eigrp *as-number timer-value*	Interface subcommand that sets the EIGRP Hello interval for that EIGRP process
ip hold-time eigrp *as-number seconds*	Interface subcommand that sets the EIGRP hold time for the interface
ip ospf hello-interval *seconds*	Interface subcommand that sets the interval for periodic Hellos
ip ospf dead-interval *number*	Interface subcommand that sets the OSPF dead timer
passive-interface *type number*	Router subcommand, for both OSPF and EIGRP that tells the routing protocol to stop sending Hellos and stop trying to discover neighbors on that interface

Table Q-6 Appendix Q **show** Command Reference

Command	Description
show ip protocols	Shows routing protocol parameters and current timer values, including an effective copy of the routing protocols' **network** commands and a list of passive interfaces
show ip eigrp interfaces	Lists the interfaces on which EIGRP has been enabled for each EIGRP process, except passive interfaces
show ip route eigrp	Lists only EIGRP-learned routes from the routing table
show ip eigrp neighbors	Lists EIGRP neighbors and status
show ip ospf interface brief	Lists the interfaces on which the OSPF protocol is enabled (based on the **network** commands), including passive interfaces
show ip ospf interface [*type number*]	Lists detailed OSPF settings for all interfaces, or the listed interface, including Hello and dead timers and OSPF area
show ip route ospf	Lists routes in the routing table learned by OSPF

Command	Description
show ip ospf neighbor	Lists neighbors and current status with neighbors, per interface
show ip ospf	Lists a group of messages about the OSPF process itself, listing the OSPF Router ID in the first line
show interfaces	Lists a long set of messages, per interface, that lists configuration, state, and counter information
show interfaces description	Lists one line of output per interface with brief status information

Table Q-7 Appendix Q **debug** Command Reference

Command	Description
debug eigrp packets	Lists log messages for EIGRP packets that flow in and out of the router
debug ip ospf adj	Issues log messages for adjacency events, meaning events related to routers becoming neighbors
debug ip ospf events	Issues log messages for each action taken by OSPF, including the receipt of messages
debug ip ospf packet	Issues log messages describing the contents of all OSPF packets
debug ip ospf hello	Issues log messages describing Hellos and Hello failures
undebug all	EXEC command used to disable all current debugs

APPENDIX R

Exam Topics Cross Reference

This appendix lists the exam topics associated with the CCNA 200-301 exam. Cisco lists the exam topics on its website. Even though changes to the exam topics are rare, you should always review those exam topics for any updates; check www.cisco.com/go/certifications and navigate to the correct exam.

Cisco organizes each list of exam topics by domains, which are major topic areas. Cisco states the percentage of the exam that should come from each domain, so you get some idea of the areas of importance. Traditionally, the score report you receive after taking the exam shows your percentage score in each domain.

This appendix includes two separate types of indices to exam topics:

- **CCNA 200-301 Exam Topic Order:** This section lists the CCNA 200-301 exam topics in the same order Cisco lists them on its website, with a list of associated book chapters. This first list shows a cross reference from each exam topic to the chapters that include at least some material about each topic.
- **Book Chapter Order Versus CCNA 200-301 Exam Topics:** This lists the same CCNA 200-301 exam topics but indexed by chapter instead of exam topic. This section lists the chapters in this book, along with the exam topics that the chapter includes. This section basically relists the kind of information found on the first page of each chapter, just in condensed form in one place.

CCNA 200-301 Exam Topic Order

The CCNA 200-301 exam includes six major topic areas (domains), each with a percentage listed. Table R-1 lists the domains and their percentages.

Table R-1 CCNA 200-301 Exam Topic Domains

Domain	Percentage
Domain 1: Network Fundamentals	20%
Domain 2: Network Access	20%
Domain 3: IP Connectivity	25%
Domain 4: IP Services	10%
Domain 5: Security Fundamentals	15%
Domain 6: Automation and Programmability	10%

Tables R-2 through R-7 list the exam topics within each of the six domains. Note that the *CCNA 200-301 Official Cert Guide, Volume 2*, covers some of the exam topics. These tables show where this book explains exam topics. Exam topics with no chapter listed are covered in Volume 2 only.

Table R-2 CCNA 200-301 Domain 1 Exam Topics (Network Fundamentals)

Exam Topic	Chapter(s)
1.1 Explain the role of network components	2, 3, 5, 7, 26
1.1.a Routers	3, 15
1.1.b L2 and L3 switches	2, 5, 7
1.1.c Next-generation firewalls and IPS	
1.1.d Access points	26
1.1.e Controllers (Cisco DNA Center and WLC)	29
1.1.f Endpoints	
1.1.g Servers	
1.2 Describe characteristics of network topology architectures	2, 3
1.2.a 2 tier	
1.2.b 3 tier	
1.2.c Spine-leaf	
1.2.d WAN	3
1.2.e Small office/home office (SOHO)	2, 15
1.2.f On-premises and cloud	
1.3 Compare physical interface and cabling types	1, 2
1.3.a Single-mode fiber, multimode fiber, copper	1, 2
1.3.b Connections (Ethernet shared media and point-to-point)	1, 2
1.3.c Concepts of PoE	
1.4 Identify interface and cable issues (collisions, errors, mismatch duplex, and/or speed)	7
1.5 Compare TCP to UDP	
1.6 Configure and verify IPv4 addressing and subnetting	6, 11, 12, 13, 14, 15, 17, 18, 22
1.7 Describe the need for private IPv4 addressing	11, 16
1.8 Configure and verify IPv6 addressing and prefix	23, 24
1.9 Compare IPv6 address types	23, 24

Exam Topic	Chapter(s)
1.9.a Global unicast	23, 24
1.9.b Unique local	23, 24
1.9.c Link local	24
1.9.d Anycast	24
1.9.e Multicast	24
1.9.f Modified EUI 64	24
1.10 Verify IP parameters for Client OS (Windows, Mac OS, Linux)	
1.11 Describe wireless principles	26
1.11.a Nonoverlapping Wi-Fi channels	26
1.11.b SSID	26
1.11.c RF	26
1.11.d Encryption	28
1.12 Explain virtualization fundamentals (virtual machines)	
1.13 Describe switching concepts	5, 8
1.13.a MAC learning and aging	5, 8
1.13.b Frame switching	5, 8
1.13.c Frame flooding	5, 8
1.13.d MAC address table	5, 8

Table R-3 CCNA 200-301 Domain 2 Exam Topics (Network Access)

Exam Topic	Chapter(s)
2.1 Configure and verify VLANs (normal range) spanning multiple switches	8
2.1.a Access ports (data and voice)	8
2.1.b Default VLAN	8
2.1.c Connectivity	8
2.2 Configure and verify interswitch connectivity	8
2.2.a Trunk ports	8
2.2.b 802.1Q	8
2.2.c Native VLAN	8
2.3 Configure and verify Layer 2 discovery protocols (Cisco Discovery Protocol and LLDP)	
2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)	8, 9, 10, 17

Exam Topic	Chapter(s)
2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations	5, 9, 10
2.5.a Root port, root bridge (primary/secondary), and other port names	9, 10
2.5.b Port states (forwarding/blocking)	9, 10
2.5.c PortFast benefits	9, 10
2.6 Compare Cisco Wireless Architectures and AP modes	27
2.7 Describe physical infrastructure connections of WLAN components (AP, WLC, access/trunk ports, and LAG)	29
2.8 Describe AP and WLC management access connections (Telnet, SSH, HTTP, HTTPS, console, and TACACS+/RADIUS)	29
2.9 Configure the components of a wireless LAN access for client connectivity using GUI only such as WLAN creation, security settings, QoS profiles, and advanced WLAN settings	29

Table R-4 CCNA 200-301 Domain 3 Exam Topics (IP Connectivity)

Exam Topic	Chapter(s)
3.1 Interpret the components of routing table	16
3.1.a Routing protocol code	16
3.1.b Prefix	16
3.1.c Network mask	16
3.1.d Next hop	16
3.1.e Administrative distance	16
3.1.f Metric	16
3.1.g Gateway of last resort	16
3.2 Determine how a router makes a forwarding decision by default	16
3.2.a Longest match	16
3.2.b Administrative distance	16, 19, 20
3.2.c Routing protocol metric	19, 20
3.3 Configure and verify IPv4 and IPv6 static routing	16, 18, 25
3.3.a Default route	16, 18, 25
3.3.b Network route	16, 18, 25
3.3.c Host route	16, 18, 25
3.3.d Floating static	16, 18, 25

Exam Topic	Chapter(s)
3.4 Configure and verify single area OSPFv2	19, 20, 21
3.4.a Neighbor adjacencies	19, 20, 21
3.4.b Point-to-point	19, 20, 21
3.4.c Broadcast (DR/BDR selection)	19, 20, 21
3.4.d Router ID	19, 20, 21
3.5 Describe the purpose of first hop redundancy protocol	

Table R-5 CCNA 200-301 Domain 4 Exam Topics (IP Services)

Exam Topics	Chapter(s)
4.1 Configure and verify inside source NAT using static and pools	
4.2 Configure and verify NTP operating in a client and server mode	
4.3 Explain the role of DHCP and DNS within the network	
4.4 Explain the function of SNMP in network operations	
4.5 Describe the use of syslog features including facilities and levels	
4.6 Configure and verify DHCP client and relay	6
4.7 Explain the forwarding per-hop behavior (PHB) for QoS such as classification, marking, queuing, congestion, policing, shaping	
4.8 Configure network devices for remote access using SSH	6
4.9 Describe the capabilities and function of TFTP/FTP in the network	

Table R-6 CCNA 200-301 Domain 5 Exam Topics (Security Fundamentals)

Exam Topics	Chapter(s)
5.1 Define key security concepts (threats, vulnerabilities, exploits, and mitigation techniques)	
5.2 Describe security program elements (user awareness, training, and physical access control)	
5.3 Configure device access control using local passwords	6
5.4 Describe security password policies elements, such as management, complexity, and password alternatives (multifactor authentication, certificates, and biometrics)	
5.5 Describe remote access and site-to-site VPNs	
5.6 Configure and verify access control lists	

Exam Topics	Chapter(s)
5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)	
5.8 Differentiate authentication, authorization, and accounting concepts	
5.9 Describe wireless security protocols (WPA, WPA2, and WPA3)	28
5.10 Configure WLAN using WPA2 PSK using the GUI	29

Table R-7 CCNA 200-301 Domain 6 Exam Topics (Programmability and Automation)

Exam Topics	Chapter(s)
6.1 Explain how automation impacts network management	
6.2 Compare traditional networks with controller-based networking	
6.3 Describe controller-based and software defined architectures (overlay, underlay, and fabric)	
6.3.a Separation of control plane and data plane	
6.3.b North-bound and south-bound APIs	
6.4 Compare traditional campus device management with Cisco DNA Center enabled device management	
6.5 Describe characteristics of REST-based APIs (CRUD, HTTP verbs, and data encoding)	
6.6 Recognize the capabilities of configuration management mechanisms Puppet, Chef, and Ansible	
6.7 Interpret JSON encoded data	

Book Chapter Order Versus CCNA 200-301 Exam Topics

Cisco organizes its exam topics based on the outcome of your learning experience, which is typically not a reasonable order for building the content of a book or course. This section lists the book chapters in sequence, with the exam topics covered in each chapter.

Book Chapter	Exam Topics Covered
Part I: Introduction to Networking	
Chapter 1: Introduction to TCP/IP Networking	**1.0 Network Fundamentals** 1.3 Compare physical interface and cabling types *1.3.a Single-mode fiber, multimode fiber, copper* *1.3.b Connections (Ethernet shared media and point-to-point)*

Book Chapter	Exam Topics Covered
Chapter 2: Fundamentals of Ethernet LANs	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.b L2 and L3 switches* 1.2 Describe characteristics of network topology architectures *1.2.e Small office/home office (SOHO)* 1.3 Compare physical interface and cabling types *1.3.a Single-mode fiber, multimode fiber, copper* *1.3.b Connections (Ethernet shared media and point-to-point)*
Chapter 3: Fundamentals of WANs and IP Routing	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.a Routers* 1.2 Describe characteristics of network topology architectures *1.2.d WAN*
Part II: Implementing Ethernet LANs	
Chapter 4: Using the Command-Line Interface	**None**
Chapter 5: Analyzing Ethernet LAN Switching	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.b L2 and L3 switches* 1.13 Describe switching concepts *1.13.a MAC learning and aging* *1.13.b Frame switching* *1.13.c Frame flooding* *1.13.d MAC address table* **2.0 Network Access** 2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations
Chapter 6: Configuring Basic Switch Management	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting **4.0 IP Services** 4.6 Configure and verify DHCP client and relay 4.8 Configure network devices for remote access using SSH **5.0 Security Fundamentals** 5.3 Configure device access control using local passwords

Book Chapter	Exam Topics Covered
Chapter 7: Configuring and Verifying Switch Interfaces	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.b L2 and L3 switches* 1.4 Describe switching concepts
Part III: Implementing VLANs and STP	
Chapter 8: Implementing Ethernet Virtual LANs	**1.0 Network Fundamentals** 1.13 Describe switching concepts *1.13.a MAC learning and aging* *1.13.b Frame switching* *1.13.c Frame flooding* *1.13.d MAC address table* **2.0 Network Access** 2.1 Configure and verify VLANs (normal range) spanning multiple switches *2.1.a Access ports (data and voice)* *2.1.b Default VLAN* *2.1.c Connectivity* 2.2 Configure and verify interswitch connectivity *2.2.a Trunk ports* *2.2.b 802.1Q* *2.2.c Native VLAN*
Chapter 9: Spanning Tree Protocol Concepts	**2.0 Network Access** 2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP) 2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations *2.5.a Root port, root bridge (primary/secondary), and other port names* *2.5.b Port states (forwarding/blocking)* *2.5.c PortFast benefits*
Chapter 10: RSTP and EtherChannel Configuration	**2.0 Network Access** 2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP) 2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations *2.5.a Root port, root bridge (primary/secondary), and other port names* *2.5.b Port states (forwarding/blocking)* *2.5.c PortFast benefits*

Book Chapter	Exam Topics Covered
Part IV: IPv4 Addressing	
Chapter 11: Perspectives on IPv4 Subnetting	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting 1.7 Describe the need for private IPv4 addressing
Chapter 12: Analyzing Classful IPv4 Networks	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting
Chapter 13: Analyzing Subnet Masks	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting
Chapter 14: Analyzing Existing Subnets	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting
Part V: IPv4 Routing	
Chapter 15: Operating Cisco Routers	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.a Routers* 1.2 Describe characteristics of network topology architectures *1.2.e Small office/home office (SOHO)* 1.6 Configure and verify IPv4 addressing and subnetting
Chapter 16: Configuring IPv4 Addressing and Static Routes	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting **3.0 IP Connectivity** 3.1 Interpret the components of routing table *3.1.a Routing protocol code* *3.1.b Prefix* *3.1.c Network mask* *3.1.d Next hop* *3.1.e Administrative distance* *3.1.f Metric* *3.1.g Gateway of last resort* 3.2 Determine how a router makes a forwarding decision by default *3.2.a Longest match* *3.2.b Administrative distance* 3.3 Configure and verify IPv4 and IPv6 static routing *3.3.a Default route* *3.3.b Network route* *3.3.c Host route* *3.3.d Floating static*

Book Chapter	Exam Topics Covered
Chapter 17: IP Routing in the LAN	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting **2.0 Network Access** 2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)
Chapter 18: Troubleshooting IPv4 Routing	**1.0 Network Fundamentals** 1.6 Configure and verify IPv4 addressing and subnetting **3.0 IP Connectivity** 3.3 Configure and verify IPv4 and IPv6 static routing *3.3.a Default route* *3.3.b Network route* *3.3.c Host route* *3.3.d Floating static*
Part VI: OSPF	
Chapter 19: Understanding OSPF Concepts	**3.0 IP Connectivity** 3.2 Determine how a router makes a forwarding decision by default *3.2.b Administrative distance* *3.2.c Routing protocol metric* 3.4 Configure and verify single area OSPFv2 *3.4.a Neighbor adjacencies* *3.4.b Point-to-point* *3.4.c Broadcast (DR/BR selection)* *3.4.d (Router ID)*
Chapter 20: Implementing OSPF	**3.0 IP Connectivity** 3.2 Determine how a router makes a forwarding decision by default *3.2.b Administrative distance* *3.2.c Routing protocol metric* 3.4 Configure and verify single area OSPFv2 *3.4.a Neighbor adjacencies* *3.4.b Point-to-point* *3.4.c Broadcast (DR/BR selection)* *3.4.d (Router ID)*
Chapter 21: OSPF Network Types and Neighbors	**3.0 IP Connectivity** 3.4 Configure and verify single area OSPFv2 *3.4.a Neighbor adjacencies* *3.4.b Point-to-point* *3.4.c Broadcast (DR/BR selection)* *3.4.d (Router ID)*

Book Chapter	Exam Topics Covered
Part VII: IP Version 6	
Chapter 22: Fundamentals of IP Version 6	**1.0 Network Fundamentals** 1.8 Configure and verify IPv6 addressing and prefix
Chapter 23: IPv6 Addressing and Subnetting	**1.0 Network Fundamentals** 1.8 Configure and verify IPv6 addressing and prefix 1.9 Compare and contrast IPv6 address types *1.9.a Global unicast* *1.9.b Unique local*
Chapter 24: Implementing IPv6 Addressing on Routers	**1.0 Network Fundamentals** 1.8 Configure and verify IPv6 addressing and prefix 1.9 Compare and contrast IPv6 address types *1.9.a Global unicast* *1.9.b Unique local* *1.9.c Link local* *1.9.d Anycast* *1.9.e Multicast* *1.9.f Modified EUI 64*
Chapter 25: Implementing IPv6 Routing	**3.0 IP Connectivity** 3.3 Configure and verify IPv4 and IPv6 static routing *3.3.a Default route* *3.3.b Network route* *3.3.c Host route* *3.3.d Floating static*
Part VIII: Wireless LANs	
Chapter 26: Fundamentals of Wireless Networks	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components *1.1.d Access Points* 1.11 Describe wireless principles *1.11.a Nonoverlapping Wi-Fi Channels* *1.11.b SSID* *1.11.c RF*
Chapter 27: Analyzing Cisco Wireless Architectures	**2.0 Network Access** 2.6 Compare Cisco Wireless Architectures and AP modes

Book Chapter	Exam Topics Covered
Chapter 28: Securing Wireless Networks	**1.0 Network Fundamentals** 1.11 Describe wireless principles 1.11.d Encryption **5.0 Security Fundamentals** 5.9 Describe wireless security protocols (WPA, WPA2, and WPA3)
Chapter 29: Building a Wireless LAN	**2.0 Network Access** 2.7 Describe physical infrastructure connections of WLAN components (AP, WLC, access/trunk ports, and LAG) 2.8 Describe AP and WLC management access connections (Telnet, SSH, HTTP, HTTPS, console, and TACACS+/RADIUS) 2.9 Configure the components of a wireless LAN access for client connectivity using GUI only such as WLAN creation, security settings, QoS profiles, and advanced WLAN settings **5.0 Security Fundamentals** 5.10 Configure WLAN using WPA2 PSK using the GUI

Official Cert Guide

Advance your IT career with hands-on learning

Hands-On Labs

Review Exercises

Config Checklists

Study Planner

CCNA 200-301

Volume 2

WENDELL ODOM,
CCIE® NO. 1624 EMERITUS

ciscopress.com

CCNA 200-301, Volume 2
Official Cert Guide

In addition to the wealth of updated content, this new edition includes a series of free hands-on exercises to help you master several real-world configuration activities. These exercises can be performed on the CCNA 200-301 Network Simulator Lite, Volume 2 software included for free on the companion website that accompanies this book. This software, which simulates the experience of working on actual Cisco routers and switches, contains the following 13 free lab exercises, covering ACL topics in Part I:

1. ACL I
2. ACL II
3. ACL III
4. ACL IV
5. ACL V
6. ACL VI
7. ACL Analysis I
8. Named ACL I
9. Named ACL II
10. Named ACL III
11. Standard ACL Configuration Scenario
12. Extended ACL I Configuration Scenario
13. Extended ACL II Configuration Scenario

If you are interested in exploring more hands-on labs and practice configuration and troubleshooting with more router and switch commands, go to **www.pearsonitcertification.com/networksimulator** for demos and to review the latest products for sale.

CCNA 200-301 Network Simulator Lite, Volume 2 system requirements:

Windows system requirements (minimum):

- Windows 10 (32/64-bit), Windows 8.1 (32/64-bit), or Windows 7 (32/64-bit)
- 1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor
- 1 GB RAM (32-bit) or 2 GB RAM (64-bit)
- 16 GB available hard disk space (32-bit) or 20 GB (64-bit)
- DirectX 9 graphics device with WDDM 1.0 or higher driver
- Adobe Acrobat Reader version 8 and above

Mac system requirements (minimum):

- macOS 10.15, 10.14, 10.13, 10.12, or 10.11
- Intel core Duo 1.83 GHz
- 512 MB RAM (1 GB recommended)
- 1.5 GB hard disk space
- 32-bit color depth at 1024 x 768 resolution
- Adobe Acrobat Reader version 8 and above

CCNA
200-301

Official Cert Guide, Volume 2

WENDELL ODOM, CCIE No. 1624 Emeritus

Cisco Press

CCNA 200-301 Official Cert Guide, Volume 2

Wendell Odom

Published by: Cisco Press

ScoutAutomatedPrintCode

Library of Congress Control Number: 2019949625

ISBN-13: 978-1-58714-713-5
ISBN-10: 1-58714-713-0

Warning and Disclaimer

This book is designed to provide information about the Cisco CCNA 200-301 exam. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The authors, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc., cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Microsoft and/or its respective suppliers make no representations about the suitability of the information contained in the documents and related graphics published as part of the services for any purpose. All such documents and related graphics are provided "as is" without warranty of any kind. Microsoft and/ or its respective suppliers hereby disclaim all warranties and conditions with regard to this information, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement. In no event shall Microsoft and/or its respective suppliers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of information available from the services.

The documents and related graphics contained herein could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Microsoft and/or its respective suppliers may make improvements and/or changes in the product(s) and/or the program(s) described herein at any time. Partial screenshots may be viewed in full within the software version specified.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screenshots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Editor-in-Chief: Mark Taub

Business Operation Manager, Cisco Press: Ronald Fligge

Director, ITP Product Management: Brett Bartow

Managing Editor: Sandra Schroeder

Development Editor: Christopher Cleveland

Senior Project Editor: Tonya Simpson

Copy Editor: Chuck Hutchinson

Technical Editor: Elan Beer

Editorial Assistant: Cindy Teeters

Cover Designer: Chuti Prasertsith

Composition: Tricia Bronkella

Indexer: Ken Johnson

Proofreader: Debbie Williams

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.

About the Author

Wendell Odom, CCIE No. 1624 Emeritus, has been in the networking industry since 1981. He has worked as a network engineer, consultant, systems engineer, instructor, and course developer; he currently works writing and creating certification study tools. This book is his 29th edition of some product for Pearson, and he is the author of all editions of the CCNA Cert Guides about Routing and Switching from Cisco Press. He has written books about topics from networking basics, certification guides throughout the years for CCENT, CCNA R&S, CCNA DC, CCNP ROUTE, CCNP QoS, and CCIE R&S. He maintains study tools, links to his blogs, and other resources at www.certskills.com.

About the Contributing Author

David Hucaby, CCIE No. 4594, CWNE No. 292, is a network engineer for University of Kentucky Healthcare. He has been authoring Cisco Press titles for 20 years, with a focus on wireless and LAN switching topics. David has bachelor of science and master of science degrees in electrical engineering. He lives in Kentucky with his wife, Marci, and two daughters.

About the Technical Reviewer

Elan Beer, CCIE No. 1837, is a senior consultant and Cisco instructor specializing in data center architecture and multiprotocol network design. For the past 27 years, Elan has designed networks and trained thousands of industry experts in data center architecture, routing, and switching. Elan has been instrumental in large-scale professional service efforts designing and troubleshooting internetworks, performing data center and network audits, and assisting clients with their short- and long-term design objectives. Elan has a global perspective of network architectures via his international clientele. Elan has used his expertise to design and troubleshoot data centers and internetworks in Malaysia, North America, Europe, Australia, Africa, China, and the Middle East. Most recently, Elan has been focused on data center design, configuration, and troubleshooting as well as service provider technologies. In 1993, Elan was among the first to obtain the Cisco Certified System Instructor (CCSI) certification, and in 1996, he was among the first to attain the Cisco System highest technical certification, the Cisco Certified Internetworking Expert. Since then, Elan has been involved in numerous large-scale data center and telecommunications networking projects worldwide.

Acknowledgments

Brett Bartow continues to be the backbone of the Cisco Press brand, guiding the entire author team through the big transition in 2019–2020 with all the changes Cisco introduced to its certifications. Simply the best! Thanks for all you do, Brett!

Dave Hucaby teamed up again to write this book, contributing one chapter here to go along with his four chapters in the CCNA Volume 1 book. It's such a joy to review his work and see such polished material from the first draft. It's been a joy to work with such a consummate professional—thanks, Dave!

Chris Cleveland developed the book—again—and made it much better—again—and did it with more juggling than ever before, I think. Five months, roughly 50 technology chapters and another 50 other book elements, and countless online elements, all done with apparent ease. Kudos to Chris, yet again!

I so look forward to reading Elan Beer's tech edits of the chapters. That may seem strange to hear, but Elan has truly amazing technical editing skills. His insights range from the details of technology, to the mind of the new learner, to wording and clarity, to holes in networking logic as compared to the wording, to tiny typos that impact the meaning. Thanks again Elan for improving the chapters so much!

Tonya Simpson managed this book, along with the CCNA Volume 1 book, all in that same compressed timeframe again. As usual, on both projects, Tonya has kept the production processes rolling along and getting through the idiosyncrasies of the content. Thanks for shepherding the book through the wild again, Tonya!

As always, thanks to the production team that works with Tonya. From fixing all my grammar and passive-voice sentences to pulling the design and layout together, they do it all; thanks for putting it all together and making it look easy. And Tonya got to juggle two books of mine at the same time (again)—thanks for managing the whole production process again.

Mike Tanamachi, illustrator and mind reader, did a great job on the figures again. Mike came through again with some beautiful finished products. Thanks again, Mike.

I could not have made the timeline for this book without Chris Burns of Certskills Professional. Chris owns much of the PTP question support and administration process, works on the labs we put on my blog, and then catches anything I need to toss over my shoulder so I can focus on the books. Chris, you are the man!

A special thank you to you readers who write in with suggestions and possible errors, and especially those of you who post online at the Cisco Learning Network and at my blog (https://blog.certskills.com). Without question, the comments I receive directly and overhear by participating at CLN made this edition a better book.

Thanks to my wonderful wife, Kris, who helps make this sometimes challenging work lifestyle a breeze. I love walking this journey with you, doll. Thanks to my daughter Hannah, who actually helped a bit with the book this summer before heading off to college (go Jackets!). And thanks to Jesus Christ, Lord of everything in my life.

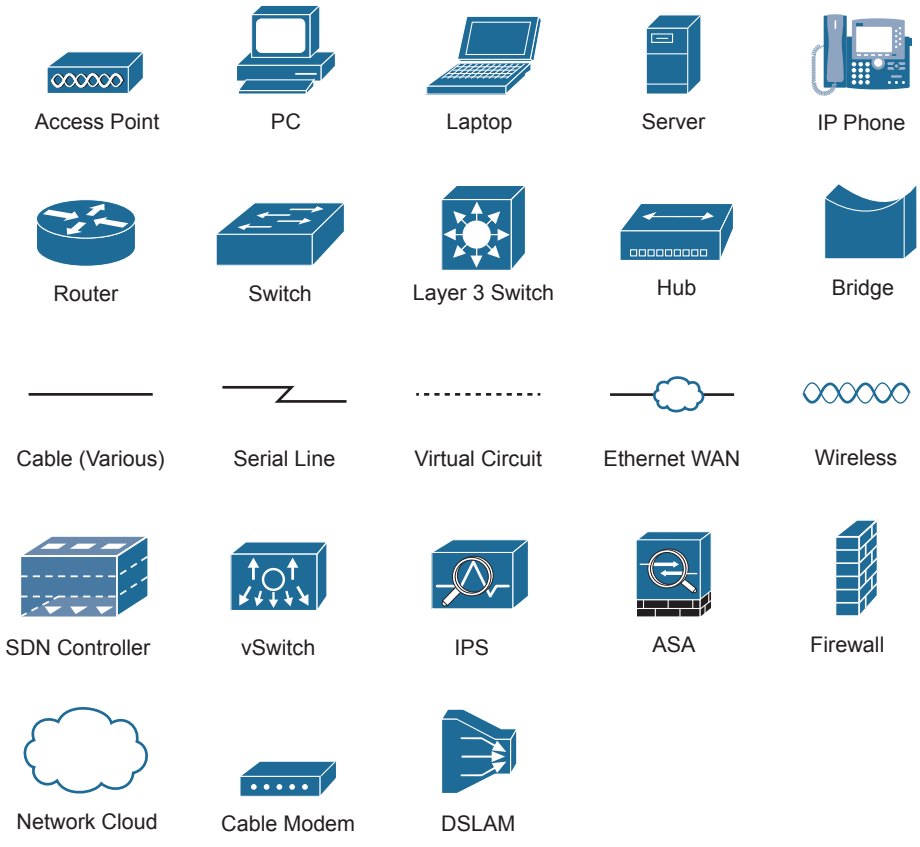
Contents at a Glance

Reader Services

To access additional content for this book, simply register your product. To start the registration process, go to www.ciscopress.com/register and log in or create an account*. Enter the product ISBN 9781587147135 and click Submit. After the process is complete, you will find any available bonus content under Registered Products.

*Be sure to check the box that you would like to hear from us to receive exclusive discounts on future editions of this product.

Icons Used in This Book



Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([ ]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

Contents

This page intentionally left blank

Introduction

About Cisco Certifications and CCNA

Congratulations! If you're reading far enough to look at this book's Introduction, you've probably already decided to go for your Cisco certification, and the CCNA certification is the one place to begin that journey. If you want to succeed as a technical person in the networking industry at all, you need to know Cisco. Cisco has a ridiculously high market share in the router and switch marketplace, with more than 80 percent market share in some markets. In many geographies and markets around the world, networking equals Cisco. If you want to be taken seriously as a network engineer, Cisco certification makes perfect sense.

NOTE This book discusses part of the content Cisco includes in the CCNA 200-301 exam, with the *CCNA 200-301 Official Cert Guide, Volume 1*, covering the rest. You will need both the Volume 1 and Volume 2 books to have all the content necessary for the exam.

The first few pages of this Introduction explain the core features of the Cisco Career Certification program, of which the Cisco Certified Network Associate (CCNA) serves as the foundation for all the other certifications in the program. This section begins with a comparison of the old to the new certifications due to some huge program changes in 2019. It then gives the key features of CCNA, how to get it, and what's on the exam.

The Big Changes to Cisco Certifications in 2019

Cisco announced sweeping changes to its career certification program around mid-year 2019. Because so many of you will have read and heard about the old versions of the CCNA certification, this Introduction begins with a few comparisons between the old and new CCNA as well as some of the other Cisco career certifications.

First, consider the Cisco career certifications before 2019, as shown in Figure I-1. At that time, Cisco offered 10 separate CCNA certifications in different technology tracks. Cisco also had eight Professional-level (CCNP, or Cisco Certified Network Professional) certifications.

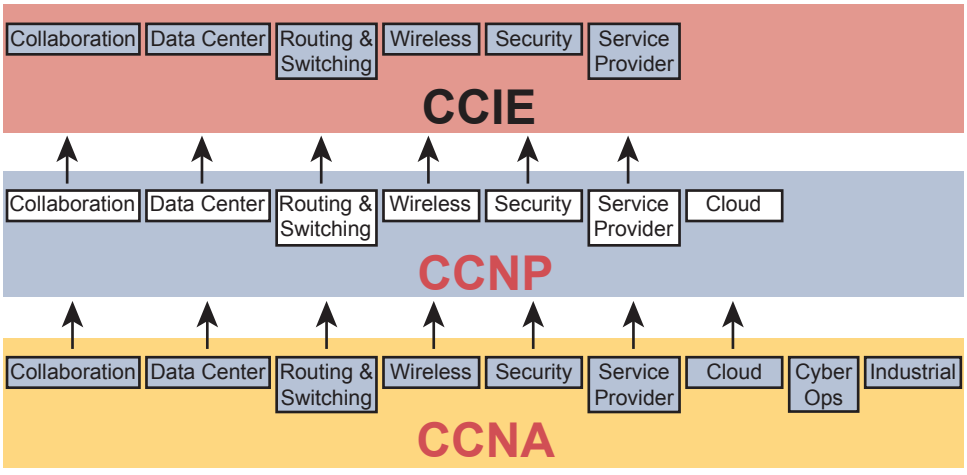


Figure I-1 *Old Cisco Certification Silo Concepts*

Why so many? Cisco began with one track—Routing and Switching—back in 1998. Over time, Cisco identified more and more technology areas that had grown to have enough content to justify another set of CCNA and CCNP certifications on those topics, so Cisco added more tracks. Many of those also grew to support expert-level topics with CCIE (Cisco Certified Internetwork Expert).

In 2019, Cisco consolidated the tracks and moved the topics around quite a bit, as shown in Figure I-2.

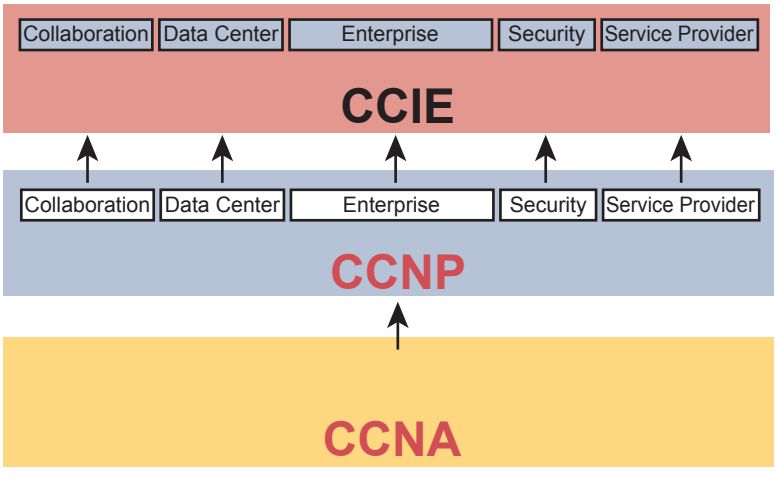


Figure I-2 *New Cisco Certification Tracks and Structure*

All the tracks now begin with the content in the one remaining CCNA certification. For CCNP, you now have a choice of five technology areas for your next steps, as shown in Figure I-2. (Note that Cisco replaced "Routing and Switching" with "Enterprise.")

Cisco made the following changes with the 2019 announcements:

CCENT: Retired the only entry-level certification (CCENT, or Cisco Certified Entry Network Technician), with no replacement.

CCNA: Retired all the CCNA certifications except what was then known as "CCNA Routing and Switching," which became simply "CCNA."

CCNP: Consolidated the professional-level (CCNP) certifications to five tracks, including merging CCNP Routing and Switching and CCNP Wireless into CCNP Enterprise.

CCIE: Achieved better alignment with CCNP tracks through the consolidations.

Cisco needed to move many of the individual exam topics from one exam to another because of the number of changes. For instance, Cisco announced the retirement of all the associate certifications—nine CCNA certifications plus the CCDA (Design Associate) certification—but those technologies didn't disappear! Cisco just moved the topics around to different exams in different certifications. (Note that Cisco later announced that CCNA Cyber Ops would remain, and not be retired, with details to be announced.)

Consider wireless LANs as an example. The 2019 announcements retired both CCNA Wireless and CCNP Wireless as certifications. Some of the old CCNA Wireless topics landed in the new CCNA, whereas others landed in the two CCNP Enterprise exams about wireless LANs.

For those of you who want to learn more about the transition, check out my blog (https://blog.certskills.com) and look for posts in the News category from around June 2019. Now on to the details about CCNA as it exists starting in 2019!

How to Get Your CCNA Certification

As you saw in Figure I-2, all career certification paths now begin with CCNA. So how do you get it? Today, you have one and only one option to achieve CCNA certification:

Take and pass one exam: the Cisco 200-301 CCNA exam.

To take the 200-301 exam, or any Cisco exam, you will use the services of Pearson VUE (vue.com). The process works something like this:

1. Establish a login at https://home.pearsonvue.com/ (or use your existing login).
2. Register for, schedule a time and place, and pay for the Cisco 200-301 exam, all from the VUE website.
3. Take the exam at the VUE testing center.
4. You will receive a notice of your score, and whether you passed, before you leave the testing center.

Types of Questions on the CCNA 200-301 Exam

The Cisco CCNA and CCNP exams all follow the same general format, with these types of questions:

- Multiple-choice, single-answer
- Multiple-choice, multiple-answer

- Testlet (one scenario with multiple multiple-choice questions)
- Drag-and-drop
- Simulated lab (sim)
- Simlet

Although the first four types of questions in the list should be somewhat familiar to you from other tests in school, the last two are more common to IT tests and Cisco exams in particular. Both use a network simulator to ask questions so that you control and use simulated Cisco devices. In particular:

Sim questions: You see a network topology and lab scenario, and can access the devices. Your job is to fix a problem with the configuration.

Simlet questions: This style combines sim and testlet question formats. As with a sim question, you see a network topology and lab scenario, and can access the devices. However, as with a testlet, you also see multiple multiple-choice questions. Instead of changing or fixing the configuration, you answer questions about the current state of the network.

These two question styles with the simulator give Cisco the ability to test your configuration skills with sim questions, and your verification and troubleshooting skills with simlet questions.

Before taking the test, learn the exam user interface by watching some videos Cisco provides about the interface. To find the videos, just go to www.cisco.com and search for "Cisco Certification Exam Tutorial Videos."

CCNA 200-301 Exam Content, Per Cisco

Ever since I was in grade school, whenever a teacher announced that we were having a test soon, someone would always ask, "What's on the test?" We all want to know, and we all want to study what matters and avoid studying what doesn't matter.

Cisco tells the world the topics on each of its exams. Cisco wants the public to know the variety of topics and get an idea about the kinds of knowledge and skills required for each topic for every Cisco certification exam. To find the details, go to www.cisco.com/go/certifications, look for the CCNA page, and navigate until you see the exam topics.

This book also lists those same exam topics in several places. From one perspective, every chapter sets about to explain a small set of exam topics, so each chapter begins with the list of exam topics covered in that chapter. However, you might want to also see the exam topics in one place, so Appendix G, "Exam Topics Cross-Reference," lists all the exam topics. You may want to download Appendix G in PDF form and keep it handy. The appendix lists the exam topics with two different cross-references:

- A list of exam topics and the chapter(s) that covers each topic
- A list of chapters and the exam topics covered in each chapter

Exam Topic Verbs and Depth

Reading and understanding the exam topics, especially deciding the depth of skills required for each exam topic, require some thought. Each exam topic mentions the name of some technology, but it also lists a verb that implies the depth to which you must master the topic. The primary exam topics each list one or more verbs that describe the skill level required. For example, consider the following exam topic:

Configure and **verify** IPv4 addressing and subnetting

Note that this one exam topic has two verbs (*configure* and *verify*). Per this exam topic, you should be able to not only configure IPv4 addresses and subnets, but you also should understand them well enough to verify that the configuration works. In contrast, the following exam topic asks you to describe a technology but does not ask you to configure it:

Describe the purpose of first hop redundancy protocol

The *describe* verb tells you to be ready to describe whatever a "first hop redundancy protocol" is. That exam topic also implies that you do not then need to be ready to configure or verify any first hop redundancy protocols (HSRP, VRRP, and GLBP).

Finally, note that the configure and verify exam topics imply that you should be able to describe and explain and otherwise master the concepts so that you understand what you have configured. The earlier "Configure and verify IPv4 addressing and subnetting" does not mean that you should know how to type commands but have no clue as to what you configured. You must first master the conceptual exam topic verbs. The progression runs something like this:

Describe, Identify, Explain, Compare/Contrast, Configure, Verify, Troubleshoot

For instance, an exam topic that lists "compare and contrast" means that you should be able to describe, identify, and explain the technology. Also, an exam topic with "configure and verify" tells you to also be ready to describe, explain, and compare/contrast.

The Context Surrounding the Exam Topics

Take a moment to navigate to www.cisco.com/go/certifications and find the list of exam topics for the CCNA 200-301 exam. Did your eyes go straight to the list of exam topics? Or did you take the time to read the paragraphs above the exam topics first?

That list of exam topics for the CCNA 200-301 exam includes a little over 50 primary exam topics and about 50 more secondary exam topics. The primary topics have those verbs as just discussed, which tell you something about the depth of skill required. The secondary topics list only the names of more technologies to know.

However, the top of the web page that lists the exam topics also lists some important information that tells us some important facts about the exam topics. In particular, that leading text, found at the beginning of Cisco exam topic pages of most every exam, tells us these important points:

- The guidelines may change over time.
- The exam topics are general guidelines about what may be on the exam.
- The actual exam may include "other related topics."

Interpreting these three facts in order, I would not expect to see a change to the published list of exam topics for the exam. I've been writing the Cisco Press CCNA Cert Guides since Cisco announced CCNA back in 1998, and I've never seen Cisco change the official exam topics in the middle of an exam—not even to fix typos. But the introductory words say that they might change the exam topics, so it's worth checking.

As for the second item in the preceding list, even before you know what the acronyms mean, you can see that the exam topics give you a general but not detailed idea about each topic. The exam topics do not attempt to clarify every nook and cranny or to list every command and parameter; however, this book serves as a great tool in that it acts as a much more detailed interpretation of the exam topics. We examine every exam topic, and if we think a concept or command is possibly within an exam topic, we put it into the book. So, the exam topics give us general guidance, and these books give us much more detailed guidance.

The third item in the list uses literal wording that runs something like this: "However, other related topics may also appear on any specific delivery of the exam." That one statement can be a bit jarring to test takers, but what does it really mean? Unpacking the statement, it says that such questions may appear on any one exam but may not; in other words, they don't set about to ask every test taker some questions that include concepts not mentioned in the exam topics. Second, the phrase "…other **related** topics…" emphasizes that any such questions would be related to some exam topic, rather than being far afield—a fact that helps us in how we respond to this particular program policy.

For instance, the CCNA 200-301 exam includes configuring and verifying the OSPF routing protocol, but it does not mention the EIGRP routing protocol. I personally would be unsurprised to see an OSPF question that required a term or fact not specifically mentioned in the exam topics, but not one that's some feature that (in my opinion) ventures far away from the OSPF features in the exam topics. Also, I would not expect to see a question about how to configure and verify EIGRP.

And just as one final side point, note that Cisco does on occasion ask a test taker some unscored questions, and those may appear to be in this vein of questions from outside topics. When you sit down to take the exam, the small print mentions that you may see unscored questions and you won't know which ones are unscored. (These questions give Cisco a way to test possible new questions.) Yet some of these might be ones that fall into the "other related topics" category but then not affect your score.

You should prepare a little differently for any Cisco exam, in comparison to, say, an exam back in school, in light of Cisco's "other related questions" policy:

- Do not approach an exam topic with an "I'll learn the core concepts and ignore the edges" approach.
- Instead, approach each exam topic with a "pick up all the points I can" approach by mastering each exam topic, both in breadth and in depth.
- Go beyond each exam topic when practicing configuration and verification by taking a little extra time to look for additional **show** commands and configuration options, and make sure you understand as much of the **show** command output that you can.

By mastering the known topics, and looking for places to go a little deeper, you will hopefully pick up the most points you can from questions about the exam topics. Then the extra practice you do with commands may happen to help you learn beyond the exam topics in a way that can help you pick up other points as well.

CCNA 200-301 Exam Content, Per This Book

When we created the Official Cert Guide content for the CCNA 200-301 exam, we considered a few options for how to package the content, and we landed on releasing a two-book set. Figure I-3 shows the setup of the content, with roughly 60 percent of the content in Volume 1 and the rest in Volume 2.

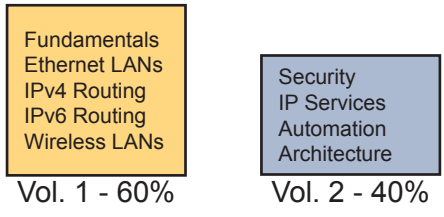


Figure I-3 *Two Books for CCNA 200-301*

The two books together cover all the exam topics in the CCNA 200-301 exam. Each chapter in each book develops the concepts and commands related to an exam topic, with clear and detailed explanations, frequent figures, and many examples that build your understanding of how Cisco networks work.

As for choosing what content to put into the books, note that we begin and finish with Cisco's exam topics, but with an eye toward predicting as many of the "other related topics" as we can. We start with the list of exam topics and apply a fair amount of experience, discussion, and other secret sauce to come up with an interpretation of what specific concepts and commands are worthy of being in the books or not. At the end of the writing process, the books should cover all the published exam topics, with additional depth and breadth that I choose based on the analysis of the exam. As we have done from the very first edition of the *CCNA Official Cert Guide*, we intend to cover each and every topic in depth. But as you would expect, we cannot predict every single fact on the exam given the nature of the exam policies, but we do our best to cover all known topics.

Book Features

This book includes many study features beyond the core explanations and examples in each chapter. This section acts as a reference to the various features in the book.

Chapter Features and How to Use Each Chapter

Each chapter of this book is a self-contained short course about one small topic area, organized for reading and study, as follows:

"Do I Know This Already?" quizzes: Each chapter begins with a pre-chapter quiz.

Foundation Topics: This is the heading for the core content section of the chapter.

Chapter Review: This section includes a list of study tasks useful to help you remember concepts, connect ideas, and practice skills-based content in the chapter.

Figure I-4 shows how each chapter uses these three key elements. You start with the DIKTA quiz. You can use the score to determine whether you already know a lot, or not so much, and determine how to approach reading the Foundation Topics (that is, the technology content in the chapter). When finished, use the Chapter Review tasks to start working on mastering your memory of the facts and skills with configuration, verification, and troubleshooting.

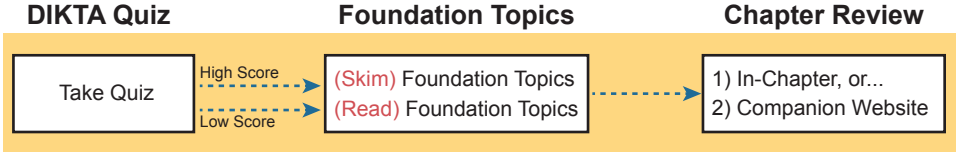


Figure I-4 *Three Primary Tasks for a First Pass Through Each Chapter*

In addition to these three main chapter features, each "Chapter Review" section uses a variety of other book features, including the following:

- **Review Key Topics:** Inside the "Foundation Topics" section, the Key Topic icon appears next to the most important items, for the purpose of later review and mastery. While all content matters, some is, of course, more important to learn, or needs more review to master, so these items are noted as key topics. The Chapter Review lists the key topics in a table. Scan the chapter for these items to review them. Or review the key topics interactively using the companion website.
- **Complete Tables from Memory:** Instead of just rereading an important table of information, you will find some tables have been turned into memory tables, an interactive exercise found on the companion website. Memory tables repeat the table but with parts of the table removed. You can then fill in the table to exercise your memory and click to check your work.
- **Key Terms You Should Know:** You do not need to be able to write a formal definition of all terms from scratch; however, you do need to understand each term well enough to understand exam questions and answers. The Chapter Review lists the key terminology from the chapter. Make sure you have a good understanding of each term and use the Glossary to cross-check your own mental definitions. You can also review key terms with the "Key Terms Flashcards" app on the companion website.

- **Labs:** Many exam topics use verbs such as *configure* and *verify*; all these refer to skills you should practice at the user interface (CLI) of a router or switch. The Chapter and Part Reviews refer you to these other tools. The upcoming section titled "About Building Hands-On Skills" discusses your options.
- **Command References:** Some book chapters cover a large number of router and switch commands. The Chapter Review includes reference tables for the commands used in that chapter, along with an explanation. Use these tables for reference, but also use them for study. Just cover one column of the table and see how much you can remember and complete mentally.
- **Review DIKTA Questions:** Although you have already seen the DIKTA questions from the chapters, re-answering those questions can prove a useful way to review facts. The Part Review suggests that you repeat the DIKTA questions but using the Pearson Test Prep (PTP) exam.

Part Features and How to Use the Part Review

The book organizes the chapters into parts for the purpose of helping you study for the exam. Each part groups a small number of related chapters together. Then the study process (described just before Chapter 1) suggests that you pause after each part to do a review of all chapters in the part. Figure I-5 lists the titles of the eight parts and the chapters in those parts (by chapter number) for this book.

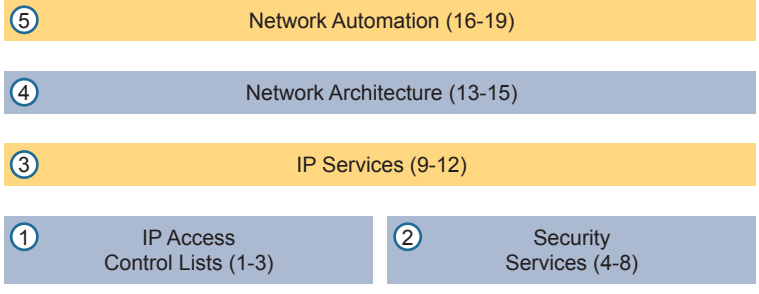


Figure I-5 *The Book Parts (by Title), and Chapter Numbers in Each Part*

The Part Review that ends each part acts as a tool to help you with spaced review sessions. Spaced reviews—that is, reviewing content several times over the course of your study—help improve retention. The Part Review activities include many of the same kinds of activities seen in the Chapter Review. Avoid skipping the Part Review, and take the time to do the review; it will help you in the long run.

The Companion Website for Online Content Review

We created an electronic version of every Chapter and Part Review task that could be improved though an interactive version of the tool. For instance, you can take a "Do I Know This Already?" quiz by reading the pages of the book, but you can also use our testing software. As another example, when you want to review the key topics from a chapter, you can find all those in electronic form as well.

All the electronic review elements, as well as other electronic components of the book, exist on this book's companion website. The companion website gives you a big advantage: you can do most of your Chapter and Part Review work from anywhere using the interactive tools on the site. The advantages include

- **Easier to use:** Instead of having to print out copies of the appendixes and do the work on paper, you can use these new apps, which provide you with an easy-to-use, interactive experience that you can easily run over and over.
- **Convenient:** When you have a spare 5–10 minutes, go to the book's website and review content from one of your recently finished chapters.
- **Untethered from the book:** You can access your review activities from anywhere—no need to have the book with you.
- **Good for tactile learners:** Sometimes looking at a static page after reading a chapter lets your mind wander. Tactile learners might do better by at least typing answers into an app, or clicking inside an app to navigate, to help keep you focused on the activity.

The interactive Chapter Review elements should improve your chances of passing as well. Our in-depth reader surveys over the years show that those who do the Chapter and Part Reviews learn more. Those who use the interactive versions of the review elements also tend to do more of the Chapter and Part Review work. So take advantage of the tools and maybe you will be more successful as well. Table I-1 summarizes these interactive applications and the traditional book features that cover the same content.

Table I-1 *Book Features with Both Traditional and App Options*

Feature	Traditional	App
Key Topic	Table with list; flip pages to find	Key Topics Table app
Config Checklist	Just one of many types of key topics	Config Checklist app
Key Terms	Listed in each "Chapter Review" section, with the Glossary in the back of the book	Glossary Flash Cards app

The companion website also includes links to download, navigate, or stream for these types of content:

- Pearson Sim Lite Desktop App
- Pearson Test Prep (PTP) Desktop App
- Pearson Test Prep (PTP) Web App
- Videos as mentioned in book chapters

How to Access the Companion Website

To access the companion website, which gives you access to the electronic content with this book, start by establishing a login at www.ciscopress.com and register your book. To do so, simply go to www.ciscopress.com/register and enter the ISBN of the print book: 9781587147135. After you have registered your book, go to your account page and click the **Registered Products** tab. From there, click the **Access Bonus Content** link to get access to the book's companion website.

Note that if you buy the *Premium Edition eBook and Practice Test* version of this book from Cisco Press, your book will automatically be registered on your account page. Simply go to your account page, click the **Registered Products** tab, and select **Access Bonus Content** to access the book's companion website.

How to Access the Pearson Test Prep (PTP) App

You have two options for installing and using the Pearson Test Prep application: a web app and a desktop app.

To use the Pearson Test Prep application, start by finding the registration code that comes with the book. You can find the code in these ways:

- **Print book:** Look in the cardboard sleeve in the back of the book for a piece of paper with your book's unique PTP code.
- **Premium Edition:** If you purchase the Premium Edition eBook and Practice Test directly from the Cisco Press website, the code will be populated on your account page after purchase. Just log in at www.ciscopress.com, click **account** to see details of your account, and click the **digital purchases** tab.
- **Amazon Kindle:** For those who purchase a Kindle edition from Amazon, the access code will be supplied directly from Amazon.
- **Other bookseller e-books:** Note that if you purchase an e-book version from any other source, the practice test is not included because other vendors to date have not chosen to vend the required unique access code.

NOTE Do not lose the activation code because it is the only means with which you can access the QA content with the book.

Once you have the access code, to find instructions about both the PTP web app and the desktop app, follow these steps:

Step 1. Open this book's companion website, as was shown earlier in this Introduction under the heading "How to Access the Companion Website."

Step 2. Click the **Practice Exams** button.

Step 3. Follow the instructions listed there both for installing the desktop app and for using the web app.

Note that if you want to use the web app only at this point, just navigate to www.pearsontestprep.com, establish a free login if you do not already have one, and register this book's practice tests using the registration code you just found. The process should take only a couple of minutes.

NOTE Amazon e-book (Kindle) customers: It is easy to miss Amazon's email that lists your PTP access code. Soon after you purchase the Kindle e-book, Amazon should send an email. However, the email uses very generic text and makes no specific mention of PTP or practice exams. To find your code, read every email from Amazon after you purchase the book. Also, do the usual checks (such as checking your spam folder) for ensuring your email arrives.

NOTE Other e-book customers: As of the time of publication, only the publisher and Amazon supply PTP access codes when you purchase their e-book editions of this book.

Feature Reference

The following list provides an easy reference to get the basic idea behind each book feature:

- **Practice exam:** The book gives you the rights to the Pearson Test Prep (PTP) testing software, available as a web app and desktop app. Use the access code on a piece of cardboard in the sleeve in the back of the book, and use the companion website to download the desktop app or navigate to the web app (or just go to www.pearsontestprep.com).
- **E-book:** Pearson offers an e-book version of this book that includes extra practice tests. If interested, look for the special offer on a coupon card inserted in the sleeve in the back of the book. This offer enables you to purchase the *CCNA 200-301 Official Cert Guide, Volume 2, Premium Edition eBook and Practice Test* at a 70 percent discount off the list price. The product includes three versions of the e-book: PDF (for reading on your computer), EPUB (for reading on your tablet, mobile device, or Nook or other e-reader), and Mobi (the native Kindle version). It also includes additional practice test questions and enhanced practice test features.
- **Mentoring videos:** The companion website also includes a number of videos about other topics as mentioned in individual chapters.
- **CCNA 200-301 Network Simulator Lite:** This lite version of the best-selling CCNA Network Simulator from Pearson provides you with a means, right now, to experience the Cisco command-line interface (CLI). No need to go buy real gear or buy a full simulator to start learning the CLI. Just install it from the companion website.
- **CCNA Simulator:** If you are looking for more hands-on practice, you might want to consider purchasing the CCNA Network Simulator. You can purchase a copy of this software from Pearson at http://pearsonitcertification.com/networksimulator or other

retail outlets. To help you with your studies, Pearson has created a mapping guide that maps each of the labs in the simulator to the specific sections in each volume of the CCNA Cert Guide. You can get this mapping guide free on the Extras tab on the book product page: www.ciscopress.com/title/9781587147135.

- **PearsonITCertification.com:** The website www.pearsonitcertification.com is a great resource for all things IT-certification related. Check out the great CCNA articles, videos, blogs, and other certification preparation tools from the industry's best authors and trainers.
- **Author's website and blogs:** The author maintains a website that hosts tools and links useful when studying for CCNA. In particular, the site has a large number of free lab exercises about CCNA content, additional sample questions, and other exercises. Additionally, the site indexes all content so you can study based on the book chapters and parts. To find it, navigate to https://blog.certskills.com.

Book Organization, Chapters, and Appendixes

The *CCNA 200-301 Official Cert Guide, Volume 1,* contains 29 chapters, while this book has 19 core chapters. Each chapter covers a subset of the topics on the CCNA exam. The book organizes its chapters into parts of three to five chapters as follows:

- **Part I: IP Access Control Lists**
  - **Chapter 1, "Introduction to TCP/IP Transport and Applications,"** completes most of the detailed discussion of the upper two layers of the TCP/IP model (transport and application), focusing on TCP and applications.
  - **Chapter 2, "Basic IPv4 Access Control Lists,"** examines how standard IP ACLs can filter packets based on the source IP address so that a router will not forward the packet.
  - **Chapter 3, "Advanced IPv4 Access Control Lists,"** examines both named and numbered ACLs, and both standard and extended IP ACLs.
- **Part II: Security Services**
  - **Chapter 4, "Security Architectures,"** discusses a wide range of fundamental concepts in network security.
  - **Chapter 5, "Securing Network Devices,"** shows how to use the router and switch CLI and introduces the concepts behind firewalls and intrusion prevention systems (IPSs).
  - **Chapter 6, "Implementing Switch Port Security,"** explains the concepts as well as how to configure and verify switch port security, a switch feature that does basic MAC-based monitoring of the devices that send data into a switch.
  - **Chapter 7, "Implementing DHCP,"** discusses how hosts can be configured with their IPv4 settings and how they can learn those settings with DHCP.
  - **Chapter 8, "DHCP Snooping and ARP Inspection,"** shows how to implement two related switch security features, with one focusing on reacting to suspicious DHCP messages and the other reacting to suspicious ARP messages.

- **Part III: IP Services**
  - **Chapter 9, "Device Management Protocols,"** discusses the concepts and configuration of some common network management tools: syslog, NTP, CDP, and LLDP.
  - **Chapter 10, "Network Address Translation,"** works through the complete concept, configuration, verification, and troubleshooting sequence for the router NAT feature, including how it helps conserve public IPv4 addresses.
  - **Chapter 11, "Quality of Service (QoS),"** discusses a wide variety of concepts all related to the broad topic of QoS.
  - **Chapter 12, "Miscellaneous IP Services,"** discusses several topics for which the exam requires conceptual knowledge but no configuration knowledge, including FHRPs (including HSRP), SNMP, TFTP, and FTP.
- **Part IV: Network Architecture**
  - **Chapter 13, "LAN Architecture,"** examines various ways to design Ethernet LANs, discussing the pros and cons, and explains common design terminology, including Power over Ethernet (PoE).
  - **Chapter 14, "WAN Architecture,"** discusses the concepts behind three WAN alternatives: Metro Ethernet, MPLS VPNs, and Internet VPNs.
  - **Chapter 15, "Cloud Architecture,"** explains the basic concepts and then generally discusses the impact that cloud computing has on a typical enterprise network, including the foundational concepts of server virtualization.
- **Part V: Network Automation**
  - **Chapter 16, "Introduction to Controller-Based Networking,"** discusses many concepts and terms related to how Software-Defined Networking (SDN) and network programmability are impacting typical enterprise networks.
  - **Chapter 17, "Cisco Software-Defined Access (SDA),"** discusses Cisco's Software-Defined Networking (SDN) offering for the enterprise, including the DNA Center controller.
  - **Chapter 18, "Understanding REST and JSON,"** explains the foundational concepts of REST APIs, data structures, and how JSON can be useful for exchanging data using APIs.
  - **Chapter 19, "Understanding Ansible, Puppet, and Chef,"** discusses the need for configuration management software and introduces the basics of each of these three configuration management tools.
- **Part VI: Final Review**
  - **Chapter 20, "Final Review,"** suggests a plan for final preparation after you have finished the core parts of the book, in particular explaining the many study options available in the book.
- **Part VII: Appendixes**
  - **Appendix A, "Numeric Reference Tables,"** lists several tables of numeric information, including a binary-to-decimal conversion table and a list of powers of 2.

- **Appendix B, "CCNA 200-301 Volume 2 Exam Updates,"** is a place for the author to add book content mid-edition. Always check online for the latest PDF version of this appendix; the appendix lists download instructions.
- **Appendix C, "Answers to the 'Do I Know This Already?' Quizzes,"** includes the explanations to all the "Do I Know This Already" quizzes.
- The **Glossary** contains definitions for many of the terms used in the book, including the terms listed in the "Key Terms You Should Know" sections at the conclusion of the chapters.

- **Online Appendixes**
  - **Appendix D, "Topics from Previous Editions**
  - **Appendix E, "Practice for Chapter 2: Basic IPv4 Access Control Lists"**
  - **Appendix F, "Previous Edition ICND1 Chapter 35: Managing IOS Files"**
  - **Appendix G, "Exam Topics Cross-Reference,"** provides some tables to help you find where each exam objective is covered in the book.
  - **Appendix H, "Study Planner,"** is a spreadsheet with major study milestones, where you can track your progress through your study.

About Building Hands-On Skills

You need skills in using Cisco routers and switches, specifically the Cisco command-line interface (CLI). The Cisco CLI is a text-based command-and-response user interface: you type a command, and the device (a router or switch) displays messages in response. To answer sim and simlet questions on the exams, you need to know a lot of commands, and you need to be able to navigate to the right place in the CLI to use those commands.

This next section walks through the options of what is included in the book, with a brief description of lab options outside the book.

Config Lab Exercises

Some router and switch features require multiple configuration commands. Part of the skill you need to learn is to remember which configuration commands work together, which ones are required, and which ones are optional. So, the challenge level goes beyond just picking the right parameters on one command. You have to choose which commands to use, in which combination, typically on multiple devices. And getting good at that kind of task requires practice.

Each Config Lab lists details about a straightforward lab exercise for which you should create a small set of configuration commands for a few devices. Each lab presents a sample lab topology, with some requirements, and you have to decide what to configure on each device. The answer then shows a sample configuration. Your job is to create the configuration and then check your answer versus the supplied answer.

Config Lab content resides outside the book at the author's blog site (https://blog.certskills.com). You can navigate to the Config Lab in a couple of ways from the site, or just go directly to https://blog.certskills.com/category/hands-on/config-lab/ to reach a list of all Config Labs. Figure I-6 shows the logo that you will see with each Config Lab.

Figure I-6 *Config Lab Logo in the Author's Blogs*

These Config Labs have several benefits, including the following:

Untethered and responsive: Do them from anywhere, from any web browser, from your phone or tablet, untethered from the book.

Designed for idle moments: Each lab is designed as a 5- to 10-minute exercise if all you are doing is typing in a text editor or writing your answer on paper.

Two outcomes, both good: Practice getting better and faster with basic configuration, or if you get lost, you have discovered a topic that you can now go back and reread to complete your knowledge. Either way, you are a step closer to being ready for the exam!

Blog format: The format allows easy adds and changes by me and easy comments by you.

Self-assessment: As part of final review, you should be able to do all the Config Labs, without help, and with confidence.

Note that the blog organizes these Config Lab posts by book chapter, so you can easily use these at both Chapter Review and Part Review.

A Quick Start with Pearson Network Simulator Lite

The decision of how to get hands-on skills can be a little scary at first. The good news: You have a free and simple first step to experience the CLI: install and use the Pearson Network Simulator Lite (or NetSim Lite) that comes with this book.

This book comes with a lite version of the best-selling CCNA Network Simulator from Pearson, which provides you with a means, right now, to experience the Cisco CLI. No need to go buy real gear or buy a full simulator to start learning the CLI. Just install it from the companion website.

The CCNA 200-301 Network Simulator Lite Volume 2 software contains 13 labs covering ACL topics from Part I in the book. So, make sure to use the NetSim Lite to learn the basics of the CLI to get a good start.

Of course, one reason that you get access to the NetSim Lite is that the publisher hopes you will buy the full product. However, even if you do not use the full product, you can still learn from the labs that come with NetSim Lite while deciding about what options to pursue.

The Pearson Network Simulator

The Config Labs and the Pearson Network Simulator Lite both fill specific needs, and they both come with the book. However, you need more than those two tools.

The single best option for lab work to do along with this book is the paid version of the Pearson Network Simulator. This simulator product simulates Cisco routers and switches so that you can learn for CCNA certification. But more importantly, it focuses on learning for the exam by providing a large number of useful lab exercises. Reader surveys tell us that those people who use the Simulator along with the book love the learning process and rave about how the book and Simulator work well together.

Of course, you need to make a decision for yourself and consider all the options. Thankfully, you can get a great idea of how the full Simulator product works by using the Pearson Network Simulator Lite product included with the book. Both have the same base code, same user interface, and same types of labs. Try the Lite version to decide if you want to buy the full product.

Note that the Simulator and the books work on a different release schedule. For a time in 2020, the Simulator will be the one created for the previous versions of the exams (ICND1 100-101, ICND2 200-101, and CCNA 200-120). Interestingly, Cisco did not add a large number of new topics that require CLI skills to the CCNA 200-301 exam as compared with its predecessor, so the old Simulator covers most of the CCNA 200-301 CLI topics. So, during the interim before the products based on the 200-301 exam come out, the old Simulator products should be quite useful.

On a practical note, when you want to do labs when reading a chapter or doing Part Review, the Simulator organizes the labs to match the book. Just look for the Sort by Chapter tab in the Simulator's user interface. However, during the months in 2020 for which the Simulator is the older edition listing the older exams in the title, you will need to refer to a PDF that lists those labs versus this book's organization. You can find that PDF on the book product page under the Downloads tab here: www.ciscopress.com/title/9781587147135.

More Lab Options

If you decide against using the full Pearson Network Simulator, you still need hands-on experience. You should plan to use some lab environment to practice as much CLI as possible.

First, you can use real Cisco routers and switches. You can buy them, new or used, or borrow them at work. You can rent them for a fee. If you have the right mix of gear, you could even do the Config Lab exercises from my blog on that gear or try to re-create examples from the book.

Cisco also makes a simulator that works very well as a learning tool: Cisco Packet Tracer. Cisco now makes Packet Tracer available for free. However, unlike the Pearson Network Simulator, it does not include lab exercises that direct you as to how to go about learning each topic. If interested in more information about Packet Tracer, check out my series about using Packet Tracer at my blog (https://blog.certskills.com); just search for "Packet Tracer."

Cisco offers a virtualization product that lets you run router and switch operating system (OS) images in a virtual environment. This tool, the Virtual Internet Routing Lab (VIRL), lets you create a lab topology, start the topology, and connect to real router and switch OS images. Check out http://virl.cisco.com for more information.

You can even rent virtual Cisco router and switch lab pods from Cisco, in an offering called Cisco Learning Labs (https://learningnetworkstore.cisco.com/cisco-learning-labs).

This book does not tell you what option to use, but you should plan on getting some hands-on practice somehow. The important thing to know is that most people need to practice using the Cisco CLI to be ready to pass these exams.

For More Information

If you have any comments about the book, submit them via www.ciscopress.com. Just go to the website, select **Contact Us**, and type your message.

Cisco might make changes that affect the CCNA certification from time to time. You should always check www.cisco.com/go/ccna for the latest details.

The *CCNA 200-301 Official Cert Guide*, *Volume 2*, helps you attain CCNA certification. This is the CCNA certification book from the only Cisco-authorized publisher. We at Cisco Press believe that this book certainly can help you achieve CCNA certification, but the real work is up to you! I trust that your time will be well spent.

Figure Credits

Figure 7-9, screenshot of network connection details © Microsoft, 2019

Figure 7-10, screenshot(s) reprinted with permission from Apple, Inc.

Figure 7-11, screenshot of Linux © The Linux Foundation

Figure 12-16, screenshot of CS Blogfigs 2018 © FileZila

Figure 13-9, electric outlet © Mike McDonald/Shutterstock

Figure 15-10, screenshot of Set Up VM with Different CPU/RAM/OS © 2019, Amazon Web Services, Inc

Figure 16-13, illustration of man icon © AlexHliv/Shutterstock

Figure 17-1, illustration of man icon © AlexHliv/Shutterstock

Figure 17-11, illustration of man icon © AlexHliv/Shutterstock

Figure 18-9, screenshot of REST GET Request © 2019 Postman, Inc.

Figure 20-1, screenshot of PTP Grading © 2019 Pearson Education

Figure 20-2, screenshot of PTP Grading © 2019 Pearson Education

Figure D-1, ribbon set © petrnutil/123RF

The *CCNA Official Cert Guide, Volume 2* includes the topics that help you build an enterprise network so all devices can communicate with all other devices. Parts I and II of this book focus on how to secure that enterprise network so that only the appropriate devices and users can communicate.

Part I focuses on IP Version 4 (IPv4) access control lists (ACLs). ACLs are IPv4 packet filters that can be programmed to look at IPv4 packet headers, make choices, and either allow a packet through or discard the packet. Because you can implement IPv4 ACLs on any router, a network engineer has a large number of options of where to use ACLs, without adding additional hardware or software, making ACLs a very flexible and useful tool.

Chapter 1 begins this part with an introduction to the TCP/IP transport layer protocols TCP and UDP, along with an introduction to several TCP/IP applications. This chapter provides the necessary background to understand the ACL chapters and to better prepare you for upcoming discussions of additional security topics in Part II and IP services topics in Part III.

Chapters 2 and 3 get into details about ACLs. Chapter 2 discusses ACL basics, avoiding some of the detail to ensure that you master several key concepts. Chapter 3 then looks at the much wider array of ACL features to make you ready to take advantage of the power of ACLs and to be ready to better manage those ACLs.

Part I

IP Access Control Lists

CHAPTER 1

Introduction to TCP/IP Transport and Applications

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.5 Compare TCP to UDP

4.0 IP Services

4.3 Explain the role of DHCP and DNS in the network

The CCNA exam focuses mostly on functions at the lower layers of TCP/IP, which define how IP networks can send IP packets from host to host using LANs and WANs. This chapter explains the basics of a few topics that receive less attention on the exams: the TCP/IP transport layer and the TCP/IP application layer. The functions of these higher layers play a big role in real TCP/IP networks. Additionally, many of the security topics in Parts I and II of this book, and some of the IP services topics in Part III, require you to know the basics of how the transport and application layers of TCP/IP work. This chapter serves as that introduction.

This chapter begins by examining the functions of two transport layer protocols: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The second major section of the chapter examines the TCP/IP application layer, including some discussion of how Domain Name System (DNS) name resolution works.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 1-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
TCP/IP Layer 4 Protocols: TCP and UDP	1–4
TCP/IP Applications	5–6

1. Which of the following header fields identify which TCP/IP application gets data received by the computer? (Choose two answers.)
   a. Ethernet Type
   b. SNAP Protocol Type
   c. IP Protocol
   d. TCP Port Number
   e. UDP Port Number
2. Which of the following are typical functions of TCP? (Choose four answers.)
   a. Flow control (windowing)
   b. Error recovery
   c. Multiplexing using port numbers
   d. Routing
   e. Encryption
   f. Ordered data transfer
3. Which of the following functions is performed by both TCP and UDP?
   a. Windowing
   b. Error recovery
   c. Multiplexing using port numbers
   d. Routing
   e. Encryption
   f. Ordered data transfer
4. What do you call data that includes the Layer 4 protocol header, and data given to Layer 4 by the upper layers, not including any headers and trailers from Layers 1 to 3? (Choose two answers.)
   a. L3PDU
   b. Chunk
   c. Segment
   d. Packet
   e. Frame
   f. L4PDU
5. In the URI http://blog.certskills.com/config-labs, which part identifies the web server?
   a. http
   b. blog.certskills.com
   c. certskills.com
   d. http://blog.certskills.com
   e. The file name.html includes the hostname.

6. Fred opens a web browser and connects to the www.certskills.com website. Which of the following are typically true about what happens between Fred's web browser and the web server? (Choose two answers.)
   a. Messages flowing toward the server use UDP destination port 80.
   b. Messages flowing from the server typically use RTP.
   c. Messages flowing to the client typically use a source TCP port number of 80.
   d. Messages flowing to the server typically use TCP.

Foundation Topics

TCP/IP Layer 4 Protocols: TCP and UDP

The OSI transport layer (Layer 4) defines several functions, the most important of which are error recovery and flow control. Likewise, the TCP/IP transport layer protocols also implement these same types of features. Note that both the OSI model and the TCP/IP model call this layer the transport layer. But as usual, when referring to the TCP/IP model, the layer name and number are based on OSI, so any TCP/IP transport layer protocols are considered Layer 4 protocols.

The key difference between TCP and UDP is that TCP provides a wide variety of services to applications, whereas UDP does not. For example, routers discard packets for many reasons, including bit errors, congestion, and instances in which no correct routes are known. As you have read already, most data-link protocols notice errors (a process called *error detection*) but then discard frames that have errors. TCP provides retransmission (error recovery) and helps to avoid congestion (flow control), whereas UDP does not. As a result, many application protocols choose to use TCP.

However, do not let UDP's lack of services make you think that UDP is worse than TCP. By providing fewer services, UDP needs fewer bytes in its header compared to TCP, resulting in fewer bytes of overhead in the network. UDP software does not slow down data transfer in cases where TCP can purposefully slow down. Also, some applications, notably today Voice over IP (VoIP) and video over IP, do not need error recovery, so they use UDP. So, UDP also has an important place in TCP/IP networks today.

Table 1-2 lists the main features supported by TCP/UDP. Note that only the first item listed in the table is supported by UDP, whereas all items in the table are supported by TCP.

Key Topic

Table 1-2 TCP/IP Transport Layer Features

Function	Description
Multiplexing using ports	Function that allows receiving hosts to choose the correct application for which the data is destined, based on the port number
Error recovery (reliability)	Process of numbering and acknowledging data with Sequence and Acknowledgment header fields
Flow control using windowing	Process that uses window sizes to protect buffer space and routing devices from being overloaded with traffic

Function	Description
Connection establishment and termination	Process used to initialize port numbers and Sequence and Acknowledgment fields
Ordered data transfer and data segmentation	Continuous stream of bytes from an upper-layer process that is "segmented" for transmission and delivered to upper-layer processes at the receiving device, with the bytes in the same order

Next, this section describes the features of TCP, followed by a brief comparison to UDP.

Transmission Control Protocol

Each TCP/IP application typically chooses to use either TCP or UDP based on the application's requirements. For example, TCP provides error recovery, but to do so, it consumes more bandwidth and uses more processing cycles. UDP does not perform error recovery, but it takes less bandwidth and uses fewer processing cycles. Regardless of which of these two TCP/IP transport layer protocols the application chooses to use, you should understand the basics of how each of these transport layer protocols works.

TCP, as defined in Request For Comments (RFC) 793, accomplishes the functions listed in Table 1-2 through mechanisms at the endpoint computers. TCP relies on IP for end-to-end delivery of the data, including routing issues. In other words, TCP performs only part of the functions necessary to deliver the data between applications. Also, the role that it plays is directed toward providing services for the applications that sit at the endpoint computers. Regardless of whether two computers are on the same Ethernet, or are separated by the entire Internet, TCP performs its functions the same way.

Figure 1-1 shows the fields in the TCP header. Although you don't need to memorize the names of the fields or their locations, the rest of this section refers to several of the fields, so the entire header is included here for reference.

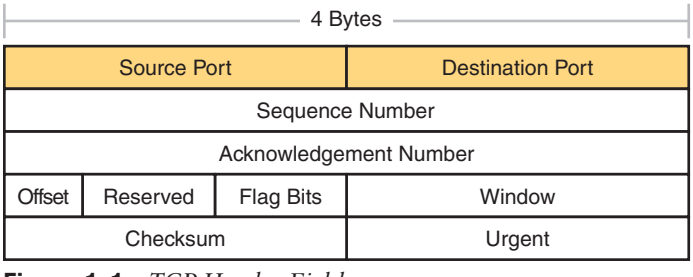


Figure 1-1 *TCP Header Fields*

The message created by TCP that begins with the TCP header, followed by any application data, is called a *TCP segment*. Alternatively, the more generic term *Layer 4 PDU*, or *L4PDU*, can also be used.

Multiplexing Using TCP Port Numbers

TCP and UDP both use a concept called *multiplexing*. Therefore, this section begins with an explanation of multiplexing with TCP and UDP. Afterward, the unique features of TCP are explored.

Multiplexing by TCP and UDP involves the process of how a computer thinks when receiving data. The computer might be running many applications, such as a web browser, an email package, or an Internet VoIP application (for example, Skype). TCP and UDP multiplexing tells the receiving computer to which application to give the received data.

Some examples will help make the need for multiplexing obvious. The sample network consists of two PCs, labeled Hannah and George. Hannah uses an application that she wrote to send advertisements that appear on George's screen. The application sends a new ad to George every 10 seconds. Hannah uses a second application, a wire-transfer application, to send George some money. Finally, Hannah uses a web browser to access the web server that runs on George's PC. The ad application and wire-transfer application are imaginary, just for this example. The web application works just like it would in real life.

Figure 1-2 shows the sample network, with George running three applications:

- A UDP-based advertisement application
- A TCP-based wire-transfer application
- A TCP web server application

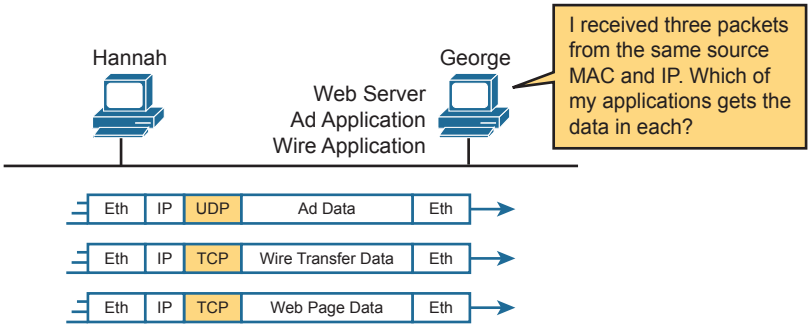


Figure 1-2 *Hannah Sending Packets to George, with Three Applications*

George needs to know which application to give the data to, but *all three packets are from the same Ethernet and IP address*. You might think that George could look at whether the packet contains a UDP or TCP header, but as you see in the figure, two applications (wire transfer and web) are using TCP.

TCP and UDP solve this problem by using a port number field in the TCP or UDP header, respectively. Each of Hannah's TCP and UDP segments uses a different *destination port number* so that George knows which application to give the data to. Figure 1-3 shows an example.

Multiplexing relies on a concept called a *socket*. A socket consists of three things:

- An IP address
- A transport protocol
- A port number

Answers to the "Do I Know This Already?" quiz:

1 D, E **2** A, B, C, F **3** C **4** C, F **5** B **6** C, D

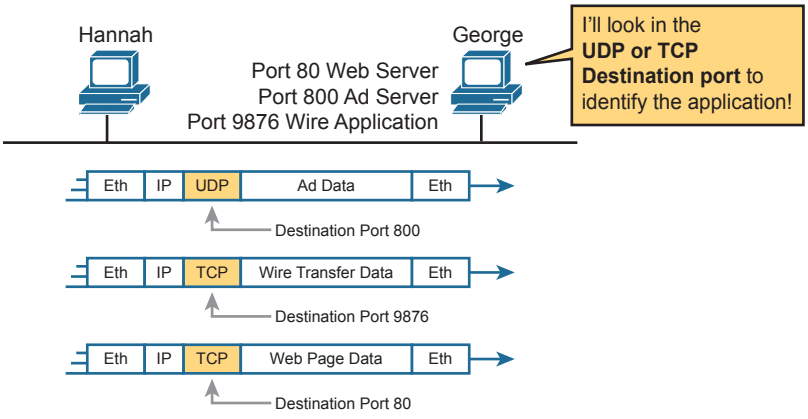


Figure 1-3 *Hannah Sending Packets to George, with Three Applications Using Port Numbers to Multiplex*

So, for a web server application on George, the socket would be (10.1.1.2, TCP, port 80) because, by default, web servers use the well-known port 80. When Hannah's web browser connects to the web server, Hannah uses a socket as well—possibly one like this: (10.1.1.1, TCP, 49160). Why 49160? Well, Hannah just needs a port number that is unique on Hannah, so Hannah sees that port 49160.

The Internet Assigned Numbers Authority (IANA), the same organization that manages IP address allocation worldwide, subdivides the port number ranges into three main ranges. The first two ranges reserve numbers that IANA can then allocate to specific application protocols through an application and review process, with the third category reserving ports to be dynamically allocated as used for clients, as with the port 49160 example in the previous paragraph. The names and ranges of port numbers (as detailed in RFC 6335) are

- **Well Known (System) Ports:** Numbers from 0 to 1023, assigned by IANA, with a stricter review process to assign new ports than user ports.
- **User (Registered) Ports:** Numbers from 1024 to 49151, assigned by IANA with a less strict process to assign new ports compared to well-known ports.
- **Ephemeral (Dynamic, Private) Ports:** Numbers from 49152 to 65535, not assigned and intended to be dynamically allocated and used temporarily for a client application while the app is running.

Figure 1-4 shows an example that uses three ephemeral ports on the user device on the left, with the server on the right using two well-known ports and one user port. The computers use three applications at the same time; hence, three socket connections are open. Because a socket on a single computer should be unique, a connection between two sockets should identify a unique connection between two computers. This uniqueness means that you can use multiple applications at the same time, talking to applications running on the same or different computers. Multiplexing, based on sockets, ensures that the data is delivered to the correct applications.

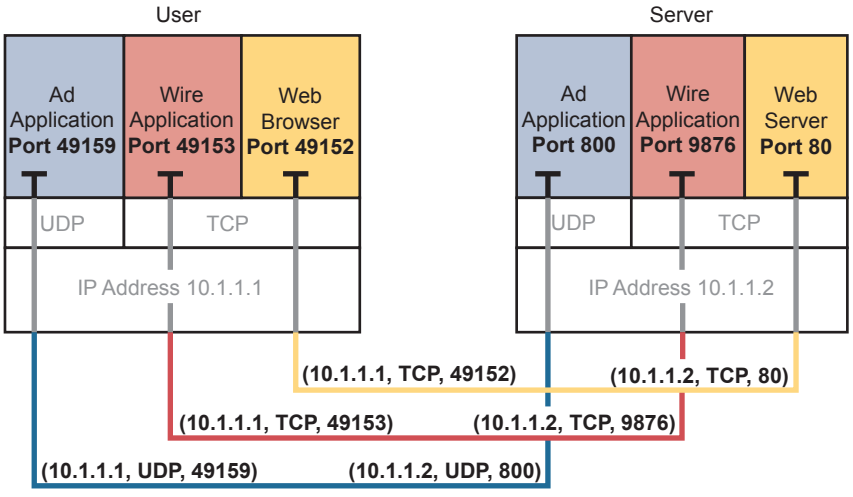


Figure 1-4 *Connections Between Sockets*

Port numbers are a vital part of the socket concept. Servers use well-known ports (or user ports), whereas clients use dynamic ports. Applications that provide a service, such as FTP, Telnet, and web servers, open a socket using a well-known port and listen for connection requests. Because these connection requests from clients are required to include both the source and destination port numbers, the port numbers used by the servers must be known beforehand. Therefore, each service uses a specific well-known port number or user port number. Both well-known and user ports are listed at www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt.

On client machines, where the requests originate, any locally unused port number can be allocated. The result is that each client on the same host uses a different port number, but a server uses the same port number for all connections. For example, 100 web browsers on the same host computer could each connect to a web server, but the web server with 100 clients connected to it would have only one socket and, therefore, only one port number (port 80, in this case). The server can tell which packets are sent from which of the 100 clients by looking at the source port of received TCP segments. The server can send data to the correct web client (browser) by sending data to that same port number listed as a destination port. The combination of source and destination sockets allows all participating hosts to distinguish between the data's source and destination. Although the example explains the concept using 100 TCP connections, the same port-numbering concept applies to UDP sessions in the same way.

NOTE You can find all RFCs online at www.rfc-editor.org/rfc/rfc*xxxx*.txt, where *xxxx* is the number of the RFC. If you do not know the number of the RFC, you can try searching by topic at www.rfc-editor.org.

Popular TCP/IP Applications

Throughout your preparation for the CCNA exam, you will come across a variety of TCP/IP applications. You should at least be aware of some of the applications that can be used to help manage and control a network.

The World Wide Web (WWW) application exists through web browsers accessing the content available on web servers. Although it is often thought of as an end-user application, you can actually use WWW to manage a router or switch. You enable a web server function in the router or switch and use a browser to access the router or switch.

The Domain Name System (DNS) allows users to use names to refer to computers, with DNS being used to find the corresponding IP addresses. DNS also uses a client/server model, with DNS servers being controlled by networking personnel and DNS client functions being part of most any device that uses TCP/IP today. The client simply asks the DNS server to supply the IP address that corresponds to a given name.

Simple Network Management Protocol (SNMP) is an application layer protocol used specifically for network device management. For example, Cisco supplies a large variety of network management products, many of them in the Cisco Prime network management software product family. They can be used to query, compile, store, and display information about a network's operation. To query the network devices, Cisco Prime software mainly uses SNMP protocols.

Traditionally, to move files to and from a router or switch, Cisco used Trivial File Transfer Protocol (TFTP). TFTP defines a protocol for basic file transfer—hence the word *trivial*. Alternatively, routers and switches can use File Transfer Protocol (FTP), which is a much more functional protocol, to transfer files. Both work well for moving files into and out of Cisco devices. FTP allows many more features, making it a good choice for the general end-user population. TFTP client and server applications are very simple, making them good tools as embedded parts of networking devices.

Some of these applications use TCP, and some use UDP. For example, Simple Mail Transfer Protocol (SMTP) and Post Office Protocol version 3 (POP3), both used for transferring mail, require guaranteed delivery, so they use TCP.

Regardless of which transport layer protocol is used, applications use a well-known port number so that clients know which port to attempt to connect to. Table 1-3 lists several popular applications and their well-known port numbers.

Key Topic

Table 1-3 Popular Applications and Their Well-Known Port Numbers

Port Number	Protocol	Application
20	TCP	FTP data
21	TCP	FTP control
22	TCP	SSH
23	TCP	Telnet
25	TCP	SMTP
53	UDP, TCP[1]	DNS
67	UDP	DHCP Server
68	UDP	DHCP Client
69	UDP	TFTP
80	TCP	HTTP (WWW)
110	TCP	POP3

Port Number	Protocol	Application
161	UDP	SNMP
443	TCP	SSL
514	UDP	Syslog

[1] DNS uses both UDP and TCP in different instances. It uses port 53 for both TCP and UDP.

Connection Establishment and Termination

TCP connection establishment occurs before any of the other TCP features can begin their work. Connection establishment refers to the process of initializing Sequence and Acknowledgment fields and agreeing on the port numbers used. Figure 1-5 shows an example of connection establishment flow.

Key Topic

Figure 1-5 *TCP Connection Establishment*

This three-way connection establishment flow (also called a three-way handshake) must complete before data transfer can begin. The connection exists between the two sockets, although the TCP header has no single socket field. Of the three parts of a socket, the IP addresses are implied based on the source and destination IP addresses in the IP header. TCP is implied because a TCP header is in use, as specified by the protocol field value in the IP header. Therefore, the only parts of the socket that need to be encoded in the TCP header are the port numbers.

TCP signals connection establishment using 2 bits inside the flag fields of the TCP header. Called the SYN and ACK flags, these bits have a particularly interesting meaning. SYN means "synchronize the sequence numbers," which is one necessary component in initialization for TCP.

Figure 1-6 shows TCP connection termination. This four-way termination sequence is straightforward and uses an additional flag, called the *FIN bit*. (FIN is short for "finished," as you might guess.) One interesting note: Before the device on the right sends the third TCP segment in the sequence, it notifies the application that the connection is coming down. It then waits on an acknowledgment from the application before sending the third segment in the figure. Just in case the application takes some time to reply, the PC on the right sends the second flow in the figure, acknowledging that the other PC wants to take down the connection. Otherwise, the PC on the left might resend the first segment repeatedly.

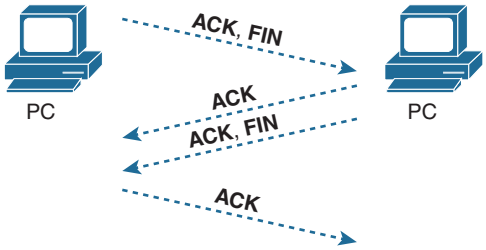


Figure 1-6 *TCP Connection Termination*

TCP establishes and terminates connections between the endpoints, whereas UDP does not. Many protocols operate under these same concepts, so the terms *connection-oriented* and *connectionless* are used to refer to the general idea of each. More formally, these terms can be defined as follows:

Key Topic

- **Connection-oriented protocol:** A protocol that requires an exchange of messages before data transfer begins, or that has a required pre-established correlation between two endpoints.
- **Connectionless protocol:** A protocol that does not require an exchange of messages and that does not require a pre-established correlation between two endpoints.

Error Recovery and Reliability

TCP provides for reliable data transfer, which is also called *reliability* or *error recovery*, depending on what document you read. To accomplish reliability, TCP numbers data bytes using the Sequence and Acknowledgment fields in the TCP header. TCP achieves reliability in both directions, using the Sequence Number field of one direction combined with the Acknowledgment field in the opposite direction.

Figure 1-7 shows an example of how the TCP Sequence and Acknowledgment fields allow the PC to send 3000 bytes of data to the server, with the server acknowledging receipt of the data. The TCP segments in the figure occur in order, from top to bottom. For simplicity's sake, all messages happen to have 1000 bytes of data in the data portion of the TCP segment. The first Sequence number is a nice round number (1000), again for simplicity's sake. The top of the figure shows three segments, with each sequence number being 1000 more than the previous, identifying the first of the 1000 bytes in the message. (That is, in this example, the first segment holds bytes 1000–1999; the second holds bytes 2000–2999; and the third holds bytes 3000–3999.)

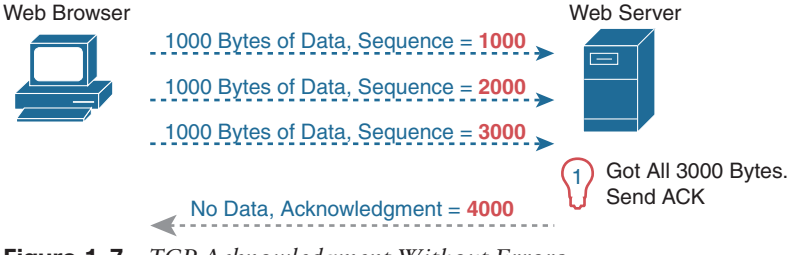


Figure 1-7 *TCP Acknowledgment Without Errors*

The fourth TCP segment in the figure—the only one flowing back from the server to the web browser—acknowledges the receipt of all three segments. How? The acknowledgment

value of 4000 means "I received all data with sequence numbers up through one less than 4000, so I am ready to receive your byte 4000 next." (Note that this convention of acknowledging by listing the next expected byte, rather than the number of the last byte received, is called *forward acknowledgment*.)

This first example does not recover from any errors, however; it simply shows the basics of how the sending host uses the sequence number field to identify the data, with the receiving host using forward acknowledgments to acknowledge the data. The more interesting discussion revolves around how to use these same tools to do error recovery. TCP uses the Sequence and Acknowledgment fields so that the receiving host can notice lost data, ask the sending host to resend, and then acknowledge that the re-sent data arrived.

Many variations exist for how TCP does error recovery. Figure 1-8 shows just one such example, with similar details compared to the previous figure. The web browser again sends three TCP segments, again 1000 bytes each, again with easy-to-remember sequence numbers. However, in this example, the second TCP segment fails to cross the network.

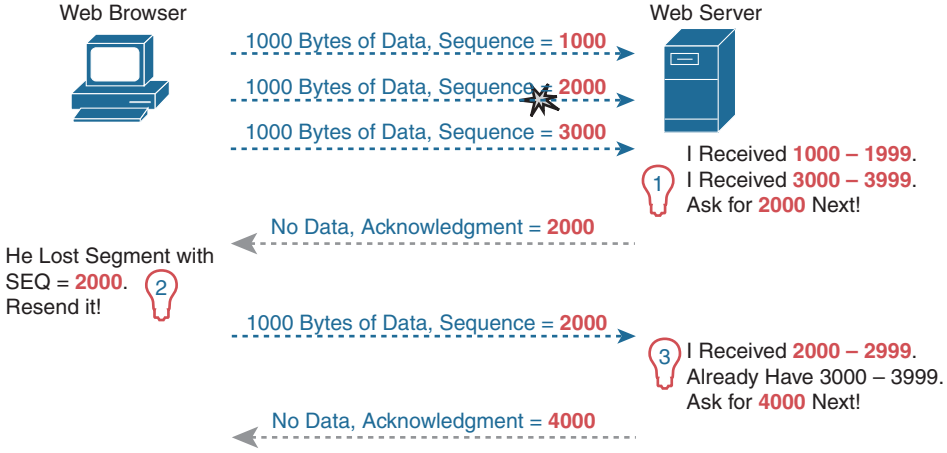


Figure 1-8 *TCP Acknowledgment with Errors*

The figure points out three sets of ideas behind how the two hosts think. First, on the right, the server realizes that it did not receive all the data. The two received TCP segments contain bytes numbered 1000–1999 and 3000–3999. Clearly, the server did not receive the bytes numbered in between. The server then decides to acknowledge all the data up to the lost data—that is, to send back a segment with the Acknowledgment field equal to 2000.

The receipt of an acknowledgment that does not acknowledge all the data sent so far tells the sending host to resend the data. The PC on the left may wait a few moments to make sure no other acknowledgments arrive (using a timer called the retransmission timer), but will soon decide that the server means "I really do need 2000 next—resend it." The PC on the left does so, as shown in the fifth of the six TCP segments in the figure.

Finally, note that the server can acknowledge not only the re-sent data, but any earlier data that had been received correctly. In this case, the server received the re-sent second TCP segment (the data with sequence numbers 2000–2999), but the server had already received the third TCP segment (the data numbered 3000–3999). The server's next Acknowledgment field acknowledges the data in both those segments, with an Acknowledgment field of 4000.

Flow Control Using Windowing

TCP implements flow control by using a window concept that is applied to the amount of data that can be outstanding and awaiting acknowledgment at any one point in time. The window concept lets the receiving host tell the sender how much data it can receive right now, giving the receiving host a way to make the sending host slow down or speed up. The receiver can slide the window size up and down—called a *sliding window* or *dynamic window*—to change how much data the sending host can send.

The sliding window mechanism makes much more sense with an example. The example, shown in Figure 1-9, uses the same basic rules as the examples in the previous few figures. In this case, none of the TCP segments have errors, and the discussion begins one TCP segment earlier than in the previous two figures.

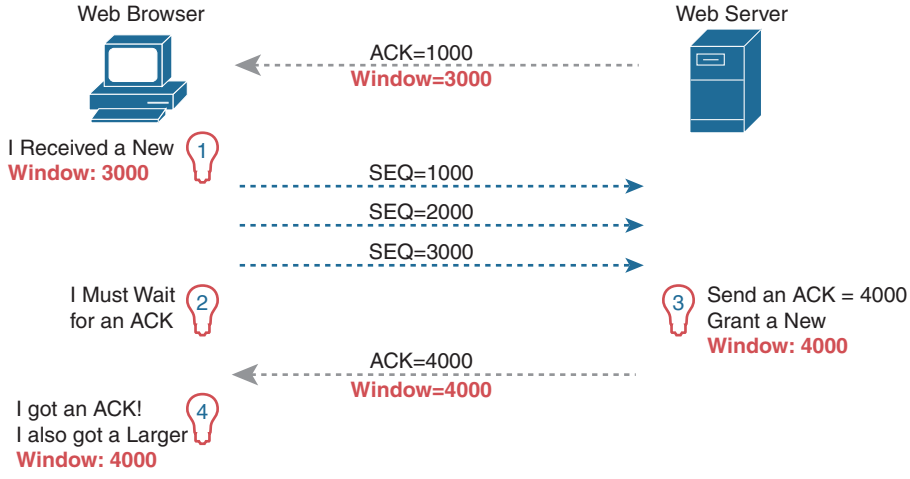


Figure 1-9 *TCP Windowing*

Begin with the first segment, sent by the server to the PC. The Acknowledgment field should be familiar by now: it tells the PC that the server expects a segment with sequence number 1000 next. The new field, the window field, is set to 3000. Because the segment flows to the PC, this value tells the PC that the PC can send no more than 3000 bytes over this connection before receiving an acknowledgment. So, as shown on the left, the PC realizes it can send only 3000 bytes, and it stops sending, waiting on an acknowledgment, after sending three 1000-byte TCP segments.

Continuing the example, the server not only acknowledges receiving the data (without any loss), but the server decides to slide the window size a little higher. Note that second message flowing right to left in the figure, this time with a window of 4000. Once the PC receives this TCP segment, the PC realizes it can send another 4000 bytes (a slightly larger window than the previous value).

Note that while the last few figures show examples for the purpose of explaining how the mechanisms work, the examples might give you the impression that TCP makes the hosts sit there and wait for acknowledgments a lot. TCP does not want to make the sending host have to wait to send data. For instance, if an acknowledgment is received before the window is exhausted, a new window begins, and the sender continues sending data until the

current window is exhausted. Often times, in a network that has few problems, few lost segments, and little congestion, the TCP windows stay relatively large with hosts seldom waiting to send.

User Datagram Protocol

UDP provides a service for applications to exchange messages. Unlike TCP, UDP is connectionless and provides no reliability, no windowing, no reordering of the received data, and no segmentation of large chunks of data into the right size for transmission. However, UDP provides some functions of TCP, such as data transfer and multiplexing using port numbers, and it does so with fewer bytes of overhead and less processing required than TCP.

UDP data transfer differs from TCP data transfer in that no reordering or recovery is accomplished. Applications that use UDP are tolerant of the lost data, or they have some application mechanism to recover lost data. For example, VoIP uses UDP because if a voice packet is lost, by the time the loss could be noticed and the packet retransmitted, too much delay would have occurred, and the voice would be unintelligible. Also, DNS requests use UDP because the user will retry an operation if the DNS resolution fails. As another example, the Network File System (NFS), a remote file system application, performs recovery with application layer code, so UDP features are acceptable to NFS.

Figure 1-10 shows the UDP header format. Most importantly, note that the header includes source and destination port fields, for the same purpose as TCP. However, the UDP has only 8 bytes, in comparison to the 20-byte TCP header shown in Figure 1-1. UDP needs a shorter header than TCP simply because UDP has less work to do.

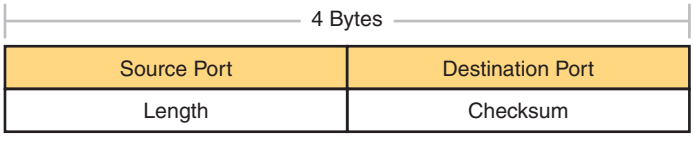


Figure 1-10 *UDP Header*

TCP/IP Applications

The whole goal of building an enterprise network, or connecting a small home or office network to the Internet, is to use applications such as web browsing, text messaging, email, file downloads, voice, and video. This section examines one particular application—web browsing using Hypertext Transfer Protocol (HTTP).

The World Wide Web (WWW) consists of all the Internet-connected web servers in the world, plus all Internet-connected hosts with web browsers. *Web servers*, which consist of web server software running on a computer, store information (in the form of *web pages*) that might be useful to different people. A *web browser*, which is software installed on an end user's computer, provides the means to connect to a web server and display the web pages stored on the web server.

NOTE Although most people use the term *web browser*, or simply *browser*, web browsers are also called *web clients*, because they obtain a service from a web server.

For this process to work, several specific application layer functions must occur. The user must somehow identify the server, the specific web page, and the protocol used to get

the data from the server. The client must find the server's IP address, based on the server's name, typically using DNS. The client must request the web page, which actually consists of multiple separate files, and the server must send the files to the web browser. Finally, for electronic commerce (e-commerce) applications, the transfer of data, particularly sensitive financial data, needs to be secure. The following sections address each of these functions.

Uniform Resource Identifiers

For a browser to display a web page, the browser must identify the server that has the web page, plus other information that identifies the particular web page. Most web servers have many web pages. For example, if you use a web browser to browse www.cisco.com and you click around that web page, you'll see another web page. Click again, and you'll see another web page. In each case, the clicking action identifies the server's IP address as well as the specific web page, with the details mostly hidden from you. (These clickable items on a web page, which in turn bring you to another web page, are called *links*.)

The browser user can identify a web page when you click something on a web page or when you enter a Uniform Resource Identifier (URI) in the browser's address area. Both options—clicking a link and typing a URI—refer to a URI, because when you click a link on a web page, that link actually refers to a URI.

NOTE Most browsers support some way to view the hidden URI referenced by a link. In several browsers, hover the mouse pointer over a link, right-click, and select **Properties**. The pop-up window should display the URI to which the browser would be directed if you clicked that link.

In common speech, many people use the terms *web address* or the similar related terms *Universal Resource Locator* (or Uniform Resource Locator [URL]) instead of URI, but URI is indeed the correct formal term. In fact, URL had been more commonly used than URI for more than a few years. However, the IETF (the group that defines TCP/IP), along with the W3C consortium (W3.org, a consortium that develops web standards) has made a concerted effort to standardize the use of URI as the general term. See RFC 7595 for some commentary to that effect.

From a practical perspective, the URIs used to connect to a web server include three key components, as noted in Figure 1-11. The figure shows the formal names of the URI fields. More importantly to this discussion, note that the text before the :// identifies the protocol used to connect to the server, the text between the // and / identifies the server by name, and the text after the / identifies the web page.

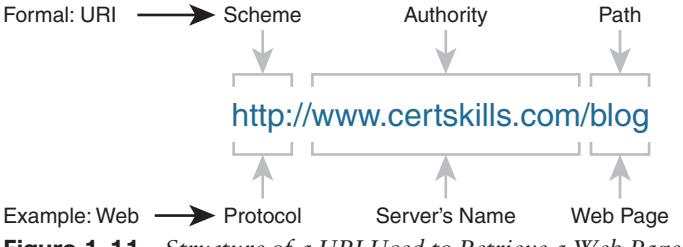


Figure 1-11 *Structure of a URI Used to Retrieve a Web Page*

In this case, the protocol is Hypertext Transfer Protocol (HTTP), the hostname is www.certskills.com, and the name of the web page is blog.

Finding the Web Server Using DNS

A host can use DNS to discover the IP address that corresponds to a particular hostname. URIs typically list the name of the server—a name that can be used to dynamically learn the IP address used by that same server. The web browser cannot send an IP packet to a destination name, but it can send a packet to a destination IP address. So, before the browser can send a packet to the web server, the browser typically needs to resolve the name inside the URI to that name's corresponding IP address.

To pull together several concepts, Figure 1-12 shows the DNS process as initiated by a web browser, as well as some other related information. From a basic perspective, the user enters the URI (in this case, http://www.cisco.com/go/learningnetwork), resolves the www.cisco.com name into the correct IP address, and starts sending packets to the web server.

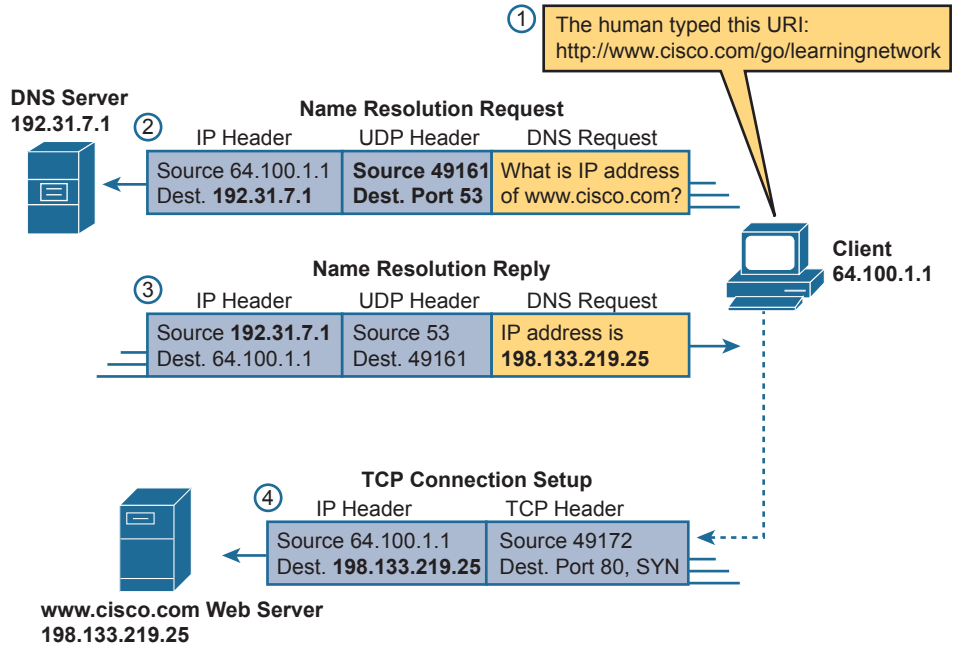


Figure 1-12 *DNS Resolution and Requesting a Web Page*

The steps shown in the figure are as follows:

1. The user enters the URI, http://www.cisco.com/go/learningnetwork, into the browser's address area.
2. The client sends a DNS request to the DNS server. Typically, the client learns the DNS server's IP address through DHCP. Note that the DNS request uses a UDP header, with a destination port of the DNS well-known port of 53. (See Table 1-3, earlier in this chapter, for a list of popular well-known ports.)
3. The DNS server sends a reply, listing IP address 198.133.219.25 as www.cisco.com's IP address. Note also that the reply shows a destination IP address of 64.100.1.1, the

client's IP address. It also shows a UDP header, with source port 53; the source port is 53 because the data is sourced, or sent, by the DNS server.

4. The client begins the process of establishing a new TCP connection to the web server. Note that the destination IP address is the just-learned IP address of the web server. The packet includes a TCP header, because HTTP uses TCP. Also note that the destination TCP port is 80, the well-known port for HTTP. Finally, the SYN bit is shown, as a reminder that the TCP connection establishment process begins with a TCP segment with the SYN bit turned on (binary 1).

The example in Figure 1-12 shows what happens when the client host does not know the IP address associated with the hostname but the enterprise does know the address. However, hosts can cache the results of DNS requests so that for a time the client does not need to ask the DNS to resolve the name. Also, the DNS server can cache the results of previous DNS requests; for instance, the enterprise DNS server in Figure 1-12 would not normally have configured information about hostnames in domains outside that enterprise, so that example relied on the DNS having cached the address associated with hostname www.cisco.com.

When the local DNS does not know the address associated with a hostname, it needs to ask for help. Figure 1-13 shows an example with the same client as in Figure 1-12. In this case, the enterprise DNS acts as a recursive DNS server, sending repeated DNS messages in an effort to identify the authoritative DNS server.

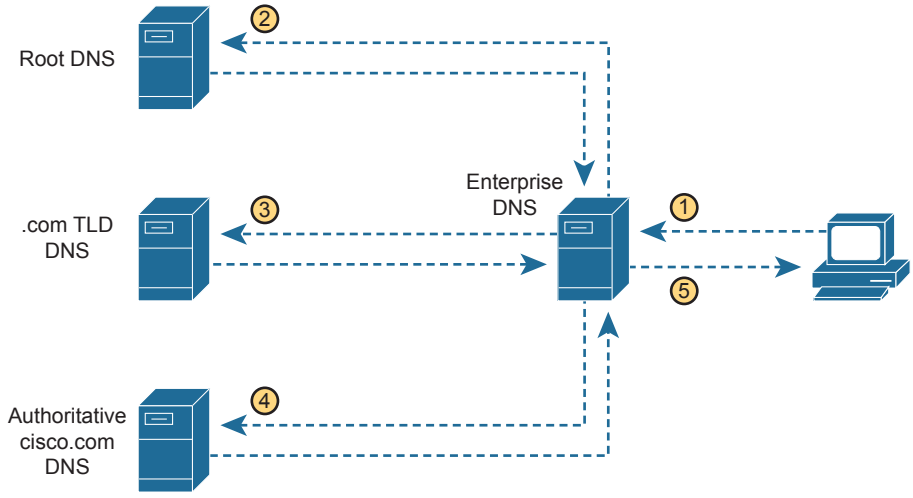


Figure 1-13 *Recursive DNS Lookup*

The steps shown in the figure are as follows:

1. The client sends a DNS request for www.cisco.com to the DNS server it knows, which is the enterprise DNS server.
2. The (recursive) enterprise DNS server does not know the answer yet, but it does not reject the client's DNS request. Instead, it follows a repetitive (recursive) process (shown as steps 2, 3, and 4), beginning with the DNS request sent to a root DNS server. The root does not supply the address either, but it supplies the IP address of another DNS server, one responsible for the .com top-level domain.

3. The recursive enterprise DNS sends the next DNS request to the DNS server learned at the previous step—this time the TLD DNS server for the .com domain. This DNS also does not know the address, but it knows the DNS server that should be the authoritative DNS server for domain cisco.com, so it supplies that DNS server's address.
4. The enterprise DNS sends another DNS request, to the DNS server whose address was learned in the previous step, again asking for resolution of the name www.cisco.com. This DNS server, the authoritative server for cisco.com, supplies the address.
5. The enterprise DNS server returns a DNS reply back to the client, supplying the IP address requested at step 1.

Transferring Files with HTTP

After a web client (browser) has created a TCP connection to a web server, the client can begin requesting the web page from the server. Most often, the protocol used to transfer the web page is HTTP. The HTTP application layer protocol, defined in RFC 7230, defines how files can be transferred between two computers. HTTP was specifically created for the purpose of transferring files between web servers and web clients.

HTTP defines several commands and responses, with the most frequently used being the HTTP GET request. To get a file from a web server, the client sends an HTTP GET request to the server, listing the filename. If the server decides to send the file, the server sends an HTTP GET response, with a return code of 200 (meaning OK), along with the file's contents.

NOTE Many return codes exist for HTTP requests. For example, when the server does not have the requested file, it issues a return code of 404, which means "file not found." Most web browsers do not show the specific numeric HTTP return codes, instead displaying a response such as "page not found" in reaction to receiving a return code of 404.

Web pages typically consist of multiple files, called *objects*. Most web pages contain text as well as several graphical images, animated advertisements, and possibly voice or video. Each of these components is stored as a different object (file) on the web server. To get them all, the web browser gets the first file. This file can (and typically does) include references to other URIs, so the browser then also requests the other objects. Figure 1-14 shows the general idea, with the browser getting the first file and then two others.

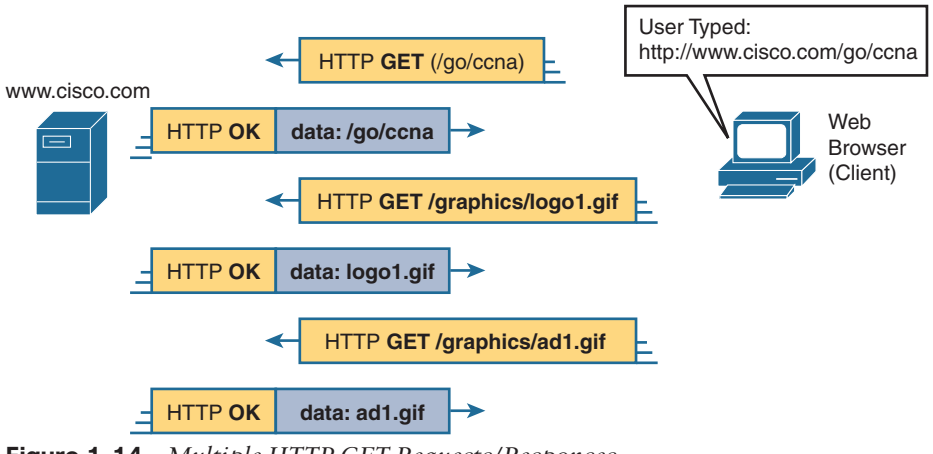


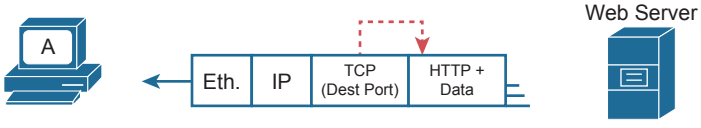
Figure 1-14 *Multiple HTTP GET Requests/Responses*

In this case, after the web browser gets the first file—the one called "/go/ccna" in the URI—the browser reads and interprets that file. Besides containing parts of the web page, the file refers to two other files, so the browser issues two additional HTTP GET requests. Note that, even though it isn't shown in the figure, all these commands flow over one (or possibly more) TCP connection between the client and the server. This means that TCP would provide error recovery, ensuring that the data was delivered.

How the Receiving Host Identifies the Correct Receiving Application

This chapter closes with a discussion of the process by which a host, when receiving any message over any network, can decide which of its many application programs should process the received data.

As an example, consider host A shown on the left side of Figure 1-15. The host happens to have three different web browser windows open, each using a unique TCP port. Host A also has an email client and a chat window open, both of which use TCP. Both the email and chat applications use a unique TCP port number on host A as shown in the figure.



Browser: TCP port 49124
Browser: TCP port 49125
Browser: TCP port 49126
Email: TCP port 49127
Chat: TCP port 49128

Figure 1-15 *Dilemma: How Host A Chooses the App That Should Receive This Data*

This chapter has shown several examples of how transport layer protocols use the destination port number field in the TCP or UDP header to identify the receiving application. For instance, if the destination TCP port value in Figure 1-15 is 49124, host A will know that the data is meant for the first of the three web browser windows.

Before a receiving host can even examine the TCP or UDP header, and find the destination port field, it must first process the outer headers in the message. If the incoming message is an Ethernet frame that encapsulates an IPv4 packet, the headers look like the details in Figure 1-16.

Key Topic

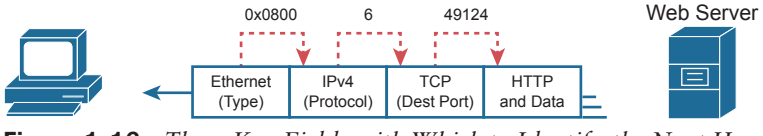


Figure 1-16 *Three Key Fields with Which to Identify the Next Header*

The receiving host needs to look at multiple fields, one per header, to identify the next header or field in the received message. For instance, host A uses an Ethernet NIC to connect to the network, so the received message is an Ethernet frame. The Ethernet Type field identifies the type of header that follows the Ethernet header—in this case, with a value of hex 0800, an IPv4 header.

The IPv4 header has a similar field called the IP Protocol field. The IPv4 Protocol field has a standard list of values that identify the next header, with decimal 6 used for TCP and decimal 17 used for UDP. In this case, the value of 6 identifies the TCP header that follows the IPv4 header. Once the receiving host realizes a TCP header exists, it can process the destination port field to determine which local application process should receive the data.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 1-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 1-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review memory tables		Book, website

Review All the Key Topics

Key Topic

Table 1-5 Key Topics for Chapter 1

Key Topic Element	Description	Page Number
Table 1-2	Functions of TCP and UDP	6
Table 1-3	Well-known TCP and UDP port numbers	11
Figure 1-5	Example of TCP connection establishment	12
List	Definitions of connection-oriented and connectionless	13
Figure 1-12	DNS name resolution	18
Figure 1-16	Header fields that identify the next header	21

Key Terms You Should Know

connection establishment, error detection, error recovery, flow control, forward acknowledgment, HTTP, ordered data transfer, port, segment, sliding windows, URI, web server, DNS server, recursive DNS server

This page intentionally left blank

CHAPTER 2

Basic IPv4 Access Control Lists

This chapter covers the following exam topics:

5.0 Security Fundamentals

5.6 Configure and verify access control lists

IPv4 access control lists (ACL) give network engineers the ability to program a filter into a router. Each router, on each interface, for both the inbound and outbound direction, can enable a different ACL with different rules. Each ACL's rules tell the router which packets to discard and which to allow through.

This chapter discusses the basics of IPv4 ACLs, and in particular, one type of IP ACL: standard numbered IP ACLs. Standard numbered ACLs use simple logic, matching on the source IP address field only, and use a configuration style that references the ACL using a number. This chapter sets out to help you learn this simpler type of ACL first. The next chapter, titled, "Advanced IPv4 Access Control Lists," completes the discussion by describing other types of IP ACLs. The other types of ACLs use features that build on the concepts you learn in this chapter, but with more complexity and additional configuration options.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 2-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
IP Access Control List Basics	1
Standard Numbered IPv4 ACLs	2–5
Practice Applying Standard IP ACLs	6

1. Barney is a host with IP address 10.1.1.1 in subnet 10.1.1.0/24. Which of the following are things that a standard IP ACL could be configured to do? (Choose two answers.)
   a. Match the exact source IP address.
   b. Match IP addresses 10.1.1.1 through 10.1.1.4 with one **access-list** command without matching other IP addresses.
   c. Match all IP addresses in Barney's subnet with one **access-list** command without matching other IP addresses.
   d. Match only the packet's destination IP address.

2. Which of the following answers list a valid number that can be used with standard numbered IP ACLs? (Choose two answers.)
   a. 1987
   b. 2187
   c. 187
   d. 87
3. Which of the following wildcard masks is most useful for matching all IP packets in subnet 10.1.128.0, mask 255.255.255.0?
   a. 0.0.0.0
   b. 0.0.0.31
   c. 0.0.0.240
   d. 0.0.0.255
   e. 0.0.15.0
   f. 0.0.248.255
4. Which of the following wildcard masks is most useful for matching all IP packets in subnet 10.1.128.0, mask 255.255.240.0?
   a. 0.0.0.0
   b. 0.0.0.31
   c. 0.0.0.240
   d. 0.0.0.255
   e. 0.0.15.255
   f. 0.0.248.255
5. ACL 1 has three statements, in the following order, with address and wildcard mask values as follows: 1.0.0.0 0.255.255.255, 1.1.0.0 0.0.255.255, and 1.1.1.0 0.0.0.255. If a router tried to match a packet sourced from IP address 1.1.1.1 using this ACL, which ACL statement does a router consider the packet to have matched?
   a. First
   b. Second
   c. Third
   d. Implied deny at the end of the ACL
6. Which of the following **access-list** commands matches all packets sent from hosts in subnet 172.16.4.0/23?
   a. **access-list 1 permit 172.16.0.5 0.0.255.0**
   b. **access-list 1 permit 172.16.4.0 0.0.1.255**
   c. **access-list 1 permit 172.16.5.0**
   d. **access-list 1 permit 172.16.5.0 0.0.0.127**

Foundation Topics

IPv4 Access Control List Basics

IPv4 access control lists (IP ACL) give network engineers a way to identify different types of packets. To do so, the ACL configuration lists values that the router can see in the IP, TCP, UDP, and other headers. For example, an ACL can match packets whose source IP address is 1.1.1.1, or packets whose destination IP address is some address in subnet 10.1.1.0/24, or packets with a destination port of TCP port 23 (Telnet).

IPv4 ACLs perform many functions in Cisco routers, with the most common use as a packet filter. Engineers can enable ACLs on a router so that the ACL sits in the forwarding path of packets as they pass through the router. After it is enabled, the router considers whether each IP packet will either be discarded or allowed to continue as if the ACL did not exist.

However, ACLs can be used for many other IOS features as well. As an example, ACLs can be used to match packets for applying Quality of Service (QoS) features. QoS allows a router to give some packets better service, and other packets worse service. For example, packets that hold digitized voice need to have very low delay, so ACLs can match voice packets, with QoS logic in turn forwarding voice packets more quickly than data packets.

This first section introduces IP ACLs as used for packet filtering, focusing on these aspects of ACLs: the locations and direction in which to enable ACLs, matching packets by examining headers, and taking action after a packet has been matched.

ACL Location and Direction

Cisco routers can apply ACL logic to packets at the point at which the IP packets enter an interface, or the point at which they exit an interface. In other words, the ACL becomes associated with an interface and for a direction of packet flow (either in or out). That is, the ACL can be applied inbound to the router, before the router makes its forwarding (routing) decision, or outbound, after the router makes its forwarding decision and has determined the exit interface to use.

The arrows in Figure 2-1 show the locations at which you could filter packets flowing left to right in the topology. For example, imagine that you wanted to allow packets sent by host A to server S1, but to discard packets sent by host B to server S1. Each arrowed line represents a location and direction at which a router could apply an ACL, filtering the packets sent by host B.

The four arrowed lines in the figure point out the location and direction for the router interfaces used to forward the packet from host B to server S1. In this particular example, those interfaces and direction are inbound on R1's F0/0 interface, outbound on R1's S0/0/0 interface, inbound on R2's S0/0/1 interface, and outbound on R2's F0/0 interface. If, for example, you enabled an ACL on R2's F0/1 interface, in either direction, that ACL could not possibly filter the packet sent from host B to server S1, because R2's F0/1 interface is not part of the route from B to S1.

Answers to the "Do I Know This Already?" quiz:

1 A, C **2** A, D **3** D **4** E **5** A **6** B

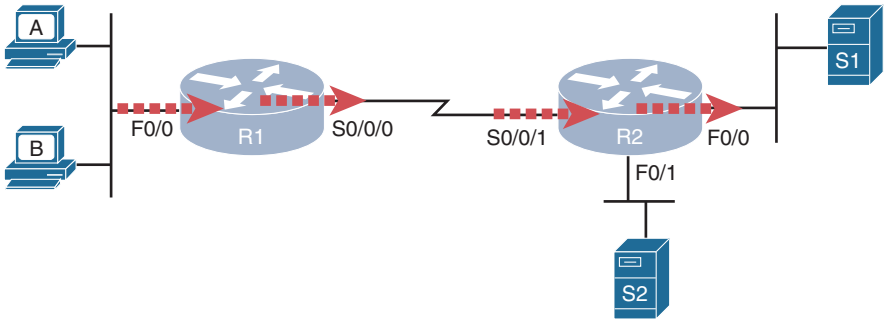


Figure 2-1 *Locations to Filter Packets from Hosts A and B Going Toward Server S1*

Key Topic

In short, to filter a packet, you must enable an ACL on an interface that processes the packet, in the same direction the packet flows through that interface.

When enabled, the router then processes every inbound or outbound IP packet using that ACL. For example, if enabled on R1 for packets inbound on interface F0/0, R1 would compare every inbound IP packet on F0/0 to the ACL to decide that packet's fate: to continue unchanged or to be discarded.

Matching Packets

When you think about the location and direction for an ACL, you must already be thinking about what packets you plan to filter (discard), and which ones you want to allow through. To tell the router those same ideas, you must configure the router with an IP ACL that matches packets. *Matching packets* refers to how to configure the ACL commands to look at each packet, listing how to identify which packets should be discarded and which should be allowed through.

Each IP ACL consists of one or more configuration commands, with each command listing details about values to look for inside a packet's headers. Generally, an ACL command uses logic like "look for these values in the packet header, and if found, discard the packet." (The action could instead be to allow the packet, rather than discard.) Specifically, the ACL looks for header fields you should already know well, including the source and destination IP addresses, plus TCP and UDP port numbers.

For example, consider an example with Figure 2-2, in which you want to allow packets from host A to server S1, but to discard packets from host B going to that same server. The hosts all now have IP addresses, and the figure shows pseudocode for an ACL on R2. Figure 2-2 also shows the chosen location to enable the ACL: inbound on R2's S0/0/1 interface.

Figure 2-2 shows a two-line ACL in a rectangle at the bottom, with simple matching logic: both statements just look to match the source IP address in the packet. When enabled, R2 looks at every inbound IP packet on that interface and compares each packet to those two ACL commands. Packets sent by host A (source IP address 10.1.1.1) are allowed through, and those sourced by host B (source IP address 10.1.1.2) are discarded.

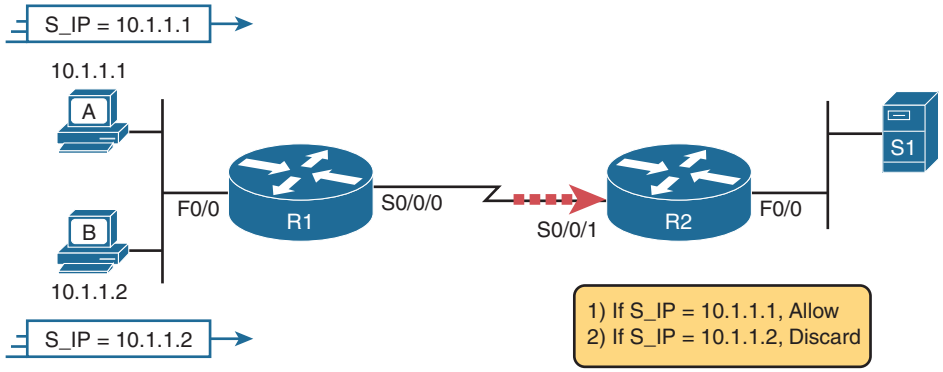


Figure 2-2 *Pseudocode to Demonstrate ACL Command-Matching Logic*

Taking Action When a Match Occurs

When using IP ACLs to filter packets, only one of two actions can be chosen. The configuration commands use the keywords **deny** and **permit**, and they mean (respectively) to discard the packet or to allow it to keep going as if the ACL did not exist.

This book focuses on using ACLs to filter packets, but IOS uses ACLs for many more features. Those features typically use the same matching logic. However, in other cases, the **deny** or **permit** keywords imply some other action.

Types of IP ACLs

Cisco IOS has supported IP ACLs since the early days of Cisco routers. Beginning with the original standard numbered IP ACLs in the early days of IOS, which could enable the logic shown earlier around Figure 2-2, Cisco has added many ACL features, including the following:

- Standard numbered ACLs (1–99)
- Extended numbered ACLs (100–199)
- Additional ACL numbers (1300–1999 standard, 2000–2699 extended)
- Named ACLs
- Improved editing with sequence numbers

This chapter focuses solely on standard numbered IP ACLs, while the next chapter discusses the other three primary categories of IP ACLs. Briefly, IP ACLs will be either numbered or named in that the configuration identifies the ACL either using a number or a name. ACLs will also be either standard or extended, with extended ACLs having much more robust abilities in matching packets. Figure 2-3 summarizes the big ideas related to categories of IP ACLs.

Key Topic

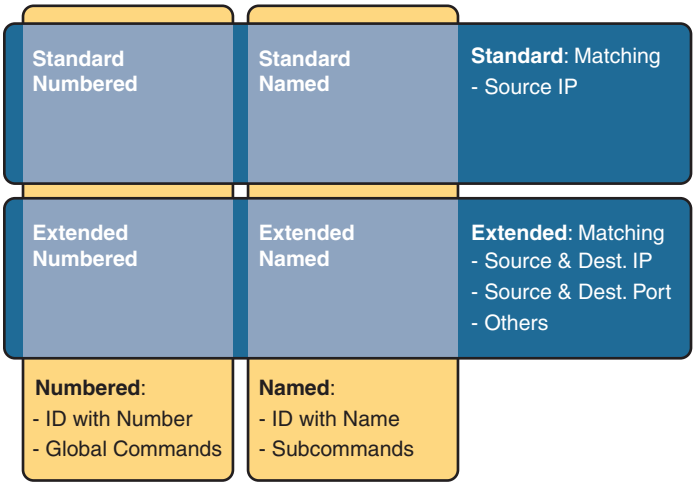


Figure 2-3 *Comparisons of IP ACL Types*

Standard Numbered IPv4 ACLs

The title of this section serves as a great introduction, if you can decode what Cisco means by each specific word. This section is about a type of Cisco filter (*ACL*) that matches only the source IP address of the packet (*standard*), is configured to identify the ACL using numbers rather than names (*numbered*), and looks at IPv4 packets.

This section examines the particulars of standard numbered IP ACLs. First, it examines the idea that one ACL is a list and what logic that list uses. Following that, the text closely looks at how to match the source IP address field in the packet header, including the syntax of the commands. This section ends with a complete look at the configuration and verification commands to implement standard ACLs.

List Logic with IP ACLs

A single ACL is both a single entity and, at the same time, a list of one or more configuration commands. As a single entity, the configuration enables the entire ACL on an interface, in a specific direction, as shown earlier in Figure 2-1. As a list of commands, each command has different matching logic that the router must apply to each packet when filtering using that ACL.

When doing ACL processing, the router processes the packet, compared to the ACL, as follows:

Key Topic

ACLs use first-match logic. Once a packet matches one line in the ACL, the router takes the action listed in that line of the ACL and stops looking further in the ACL.

To see exactly what that means, consider the example built around Figure 2-4. The figure shows an example ACL 1 with three lines of pseudocode. This example applies ACL 1 on R2's S0/0/1 interface, inbound (the same location as in earlier Figure 2-2).

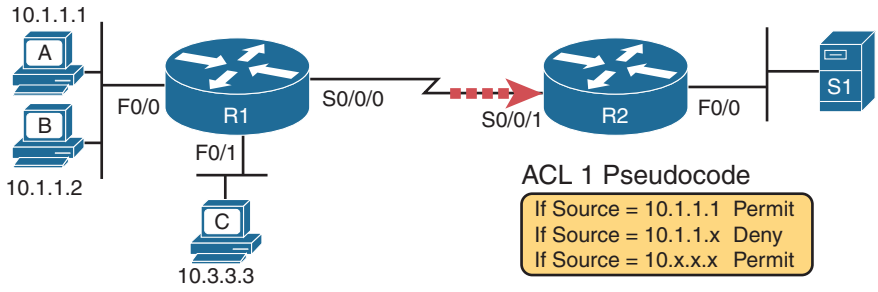


Figure 2-4 *Backdrop for Discussion of List Process with IP ACLs*

Consider the first-match ACL logic for a packet sent by host A to server S1. The source IP address will be 10.1.1.1, and it will be routed so that it enters R2's S0/0/1 interface, driving R2's ACL 1 logic. R2 compares this packet to the ACL, matching the first item in the list with a permit action. So this packet should be allowed through, as shown in Figure 2-5, on the left.

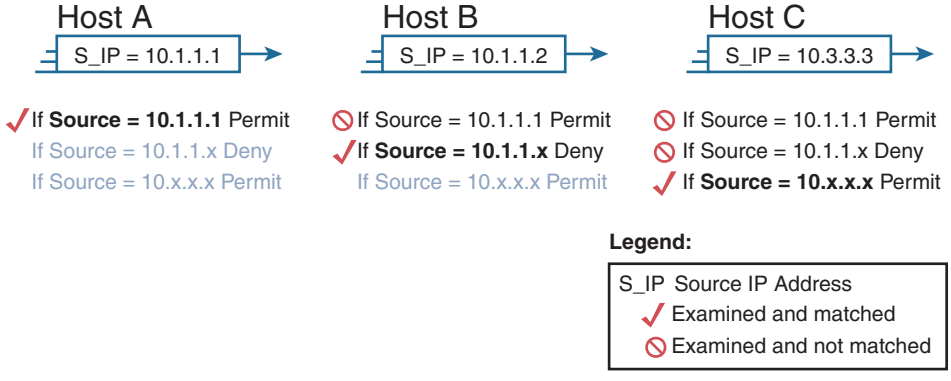


Figure 2-5 *ACL Items Compared for Packets from Hosts A, B, and C in Figure 2-4*

Next, consider a packet sent by host B, source IP address 10.1.1.2. When the packet enters R2's S0/0/1 interface, R2 compares the packet to ACL 1's first statement and does not make a match (10.1.1.1 is not equal to 10.1.1.2). R2 then moves to the second statement, which requires some clarification. The ACL pseudocode, back in Figure 2-4, shows 10.1.1.x, which is meant to be shorthand that any value can exist in the last octet. Comparing only the first three octets, R2 decides that this latest packet does have a source IP address that begins with the first three octets 10.1.1, so R2 considers that to be a match on the second statement. R2 takes the listed action (deny), discarding the packet. R2 also stops ACL processing on the packet, ignoring the third line in the ACL.

Finally, consider a packet sent by host C, again to server S1. The packet has source IP address 10.3.3.3, so when it enters R2's S0/0/1 interface and drives ACL processing on R2, R2 looks at the first command in ACL 1. R2 does not match the first ACL command (10.1.1.1 in the command is not equal to the packet's 10.3.3.3). R2 looks at the second command, compares the first three octets (10.1.1) to the packet source IP address (10.3.3), and still finds no match. R2 then looks at the third command. In this case, the wildcard means ignore the last three octets and just compare the first octet (10), so the packet matches. R2 then takes the listed action (permit), allowing the packet to keep going.

This sequence of processing an ACL as a list happens for any type of IOS ACL: IP, other protocols, standard or extended, named or numbered.

Finally, if a packet does not match any of the items in the ACL, the packet is discarded. The reason is that every IP ACL has a *deny all* statement implied at the end of the ACL. It does not exist in the configuration, but if a router keeps searching the list, and no match is made by the end of the list, IOS considers the packet to have matched an entry that has a **deny** action.

Matching Logic and Command Syntax

Standard numbered IP ACLs use the following global command:

```
access-list {1-99 | 1300-1999} {permit | deny} matching-parameters
```

Each standard numbered ACL has one or more **access-list** commands with the same number, any number from the ranges shown in the preceding line of syntax. (One number is no better than the other.) IOS refers to each line in an ACL as an Access Control Entry (ACE), but many engineers just call them ACL statements.

Besides the ACL number, each **access-list** command also lists the action (**permit** or **deny**), plus the matching logic. The rest of this section examines how to configure the matching parameters, which, for standard ACLs, means that you can only match the source IP address or portions of the source IP address using something called an ACL wildcard mask.

Matching the Exact IP Address

To match a specific source IP address, the entire IP address, all you have to do is type that IP address at the end of the command. For example, the previous example uses pseudocode for "permit if source = 10.1.1.1." The following command configures that logic with correct syntax using ACL number 1:

```
access-list 1 permit 10.1.1.1
```

Matching the exact full IP address is that simple.

In earlier IOS versions, the syntax included a **host** keyword. Instead of simply typing the full IP address, you first typed the **host** keyword and then the IP address. Note that in later IOS versions, if you use the **host** keyword, IOS accepts the command but then removes the keyword.

```
access-list 1 permit host 10.1.1.1
```

Matching a Subset of the Address with Wildcards

Often, the business goals you want to implement with an ACL do not match a single particular IP address, but rather a range of IP addresses. Maybe you want to match all IP addresses in a subnet. Maybe you want to match all IP addresses in a range of subnets. Regardless, you want to check for more than one IP address in a range of addresses.

IOS allows standard ACLs to match a range of addresses using a tool called a *wildcard mask*. Note that this is not a subnet mask. The wildcard mask (which this book abbreviates as *WC mask*) gives the engineer a way to tell IOS to ignore parts of the address when making comparisons, essentially treating those parts as wildcards, as if they already matched.

You can think about WC masks in decimal and in binary, and both have their uses. To begin, think about WC masks in decimal, using these rules:

Key Topic

Decimal 0: The router must compare this octet as normal.

Decimal 255: The router ignores this octet, considering it to already match.

Keeping these two rules in mind, consider Figure 2-6, which demonstrates this logic using three different but popular WC masks: one that tells the router to ignore the last octet, one that tells the router to ignore the last two octets, and one that tells the router to ignore the last three octets.

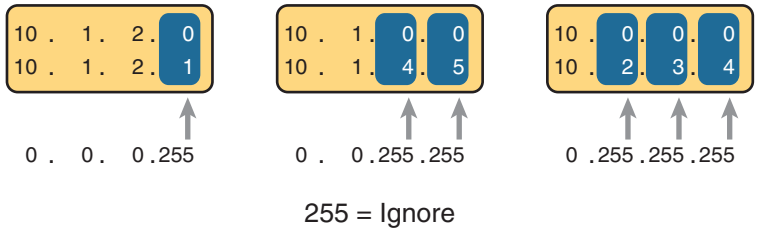


Figure 2-6 *Logic for WC Masks 0.0.0.255, 0.0.255.255, and 0.255.255.255*

All three examples in the boxes of Figure 2-6 show two numbers that are clearly different. The WC mask causes IOS to compare only some of the octets, while ignoring other octets. All three examples result in a match, because each wildcard mask tells IOS to ignore some octets. The example on the left shows WC mask 0.0.0.255, which tells the router to treat the last octet as a wildcard, essentially ignoring that octet for the comparison. Similarly, the middle example shows WC mask 0.0.255.255, which tells the router to ignore the two octets on the right. The rightmost case shows WC mask 0.255.255.255, telling the router to ignore the last three octets when comparing values.

To see the WC mask in action, think back to the earlier example related to Figure 2-4 and Figure 2-5. The pseudocode ACL in those two figures used logic that can be created using a WC mask. As a reminder, the logic in the pseudocode ACL in those two figures included the following:

Line 1: Match and permit all packets with a source address of exactly 10.1.1.1.

Line 2: Match and deny all packets with source addresses with first three octets 10.1.1.

Line 3: Match and permit all addresses with first single octet 10.

Figure 2-7 shows the updated version of Figure 2-4, but with the completed, correct syntax, including the WC masks. In particular, note the use of WC mask 0.0.0.255 in the second command, telling R2 to ignore the last octet of the number 10.1.1.0, and the WC mask 0.255.255.255 in the third command, telling R2 to ignore the last three octets in the value 10.0.0.0.

Finally, note that when using a WC mask, the **access-list** command's loosely defined *source* parameter should be a 0 in any octets where the WC mask is a 255. IOS will specify a source address to be 0 for the parts that will be ignored, even if nonzero values were configured.

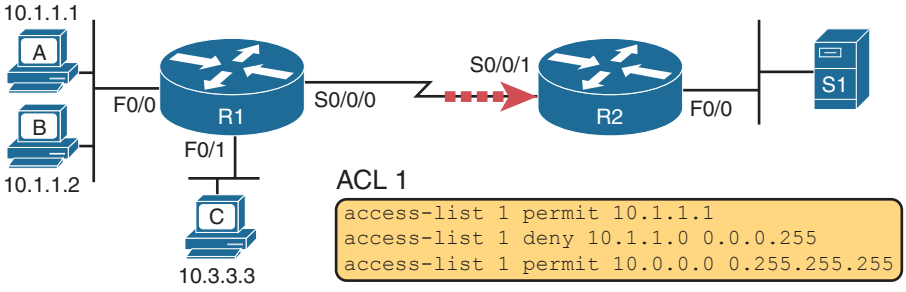


Figure 2-7 *Syntactically Correct ACL Replaces Pseudocode from Figure 2-4*

Binary Wildcard Masks

Wildcard masks, as dotted-decimal number (DDN) values, actually represent a 32-bit binary number. As a 32-bit number, the WC mask actually directs the router's logic bit by bit. In short, a WC mask bit of 0 means the comparison should be done as normal, but a binary 1 means that the bit is a wildcard and can be ignored when comparing the numbers.

Thankfully, for the purposes of CCNA study, and for most real-world applications, you can ignore the binary WC mask. Why? Well, we generally want to match a range of addresses that can be easily identified by a subnet number and mask, whether it be a real subnet, or a summary route that groups subnets together. If you can describe the range of addresses with a subnet number and mask, you can find the numbers to use in your ACL with some simple decimal math, as discussed next.

NOTE If you really want to know the binary mask logic, take the two DDN numbers the ACL will compare (one from the **access-list** command and the other from the packet header) and convert both to binary. Then, also convert the WC mask to binary. Compare the first two binary numbers bit by bit, but also ignore any bits for which the WC mask happens to list a binary 1, because that tells you to ignore the bit. If all the bits you checked are equal, it's a match!

Finding the Right Wildcard Mask to Match a Subnet

In many cases, an ACL needs to match all hosts in a particular subnet. To match a subnet with an ACL, you can use the following shortcut:

Key Topic

- Use the subnet number as the source value in the **access-list** command.
- Use a wildcard mask found by subtracting the subnet mask from 255.255.255.255.

For example, for subnet 172.16.8.0 255.255.252.0, use the subnet number (172.16.8.0) as the address parameter, and then do the following math to find the wildcard mask:

```
  255.255.255.255
– 255.255.252.0
        0. 0. 3.255
```

Continuing this example, a completed command for this same subnet would be as follows:

```
access-list 1 permit 172.16.8.0 0.0.3.255
```

The section "Practice Applying Standard IP ACLs" gives you a chance to practice matching subnets when configuring ACLs.

Matching Any/All Addresses

In some cases, you will want one ACL command to match any and all packets that reach that point in the ACL. First, you have to know the (simple) way to match all packets using the **any** keyword. More importantly, you need to think about when to match any and all packets.

First, to match any and all packets with an ACL command, just use the **any** keyword for the address. For example, to permit all packets:

```
access-list 1 permit any
```

So, when and where should you use such a command? Remember, all Cisco IP ACLs end with an implicit **deny any** concept at the end of each ACL. That is, if a router compares a packet to the ACL, and the packet matches none of the configured statements, the router discards the packet. Want to override that default behavior? Configure a **permit any** at the end of the ACL.

You might also want to explicitly configure a command to deny all traffic (for example, **access-list 1 deny any**) at the end of an ACL. Why, when the same logic already sits at the end of the ACL anyway? Well, the ACL **show** commands list counters for the number of packets matched by each command in the ACL, but there is no counter for that implicit **deny any** concept at the end of the ACL. So, if you want to see counters for how many packets are matched by the **deny any** logic at the end of the ACL, configure an explicit **deny any**.

Implementing Standard IP ACLs

This chapter has already introduced all the configuration steps in bits and pieces. This section summarizes those pieces as a configuration process. The process also refers to the **access-list** command, whose generic syntax is repeated here for reference:

```
access-list access-list-number {deny | permit} source [source-wildcard]
```

Key Topic

Step 1. Plan the location (router and interface) and direction (in or out) on that interface:

- **A.** Standard ACLs should be placed near to the destination of the packets so that they do not unintentionally discard packets that should not be discarded.
- **B.** Because standard ACLs can only match a packet's source IP address, identify the source IP addresses of packets as they go in the direction that the ACL is examining.

Step 2. Configure one or more **access-list** global configuration commands to create the ACL, keeping the following in mind:

- **A.** The list is searched sequentially, using first-match logic.
- **B.** The default action, if a packet does not match any of the **access-list** commands, is to **deny** (discard) the packet.

Step 3. Enable the ACL on the chosen router interface, in the correct direction, using the **ip access-group** *number* {**in | out**} interface subcommand.

The rest of this section shows a couple of examples.

Standard Numbered ACL Example 1

The first example shows the configuration for the same requirements demonstrated with Figure 2-4 and Figure 2-5. Restated, the requirements for this ACL are as follows:

1. Enable the ACL inbound on R2's S0/0/1 interface.
2. Permit packets coming from host A.
3. Deny packets coming from other hosts in host A's subnet.
4. Permit packets coming from any other address in Class A network 10.0.0.0.
5. The original example made no comment about what to do by default, so simply deny all other traffic.

Example 2-1 shows a completed correct configuration, starting with the configuration process, followed by output from the **show running-config** command.

Example 2-1 *Standard Numbered ACL Example 1 Configuration*

```
R2# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)# access-list 1 permit 10.1.1.1
R2(config)# access-list 1 deny 10.1.1.0 0.0.0.255
R2(config)# access-list 1 permit 10.0.0.0 0.255.255.255
R2(config)# interface S0/0/1
R2(config-if)# ip access-group 1 in
R2(config-if)# ^Z
R2# show running-config
! Lines omitted for brevity

access-list 1 permit 10.1.1.1
access-list 1 deny 10.1.1.0 0.0.0.255
access-list 1 permit 10.0.0.0 0.255.255.255
```

First, pay close attention to the configuration process at the top of the example. Note that the **access-list** command does not change the command prompt from the global configuration mode prompt, because the **access-list** command is a global configuration command. Then, compare that to the output of the **show running-config** command: the details are identical compared to the commands that were added in configuration mode. Finally, make sure to note the **ip access-group 1 in** command, under R2's S0/0/1 interface, which enables the ACL logic (both location and direction).

Example 2-2 lists some output from Router R2 that shows information about this ACL. The **show ip access-lists** command lists details about IPv4 ACLs only, while the **show access-lists** command lists details about IPv4 ACLs plus any other types of ACLs that are currently configured; for example, IPv6 ACLs.

Example 2-2 *ACL* **show** *Commands on R2*

```
R2# show ip access-lists
Standard IP access list 1
    10 permit 10.1.1.1 (107 matches)
    20 deny   10.1.1.0, wildcard bits 0.0.0.255 (4 matches)
```

```
    30 permit 10.0.0.0, wildcard bits 0.255.255.255 (10 matches)
R2# show access-lists
Standard IP access list 1
    10 permit 10.1.1.1 (107 matches)
    20 deny   10.1.1.0, wildcard bits 0.0.0.255 (4 matches)
    30 permit 10.0.0.0, wildcard bits 0.255.255.255 (10 matches)
R2# show ip interface s0/0/1
Serial0/0/1 is up, line protocol is up
  Internet address is 10.1.2.2/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Multicast reserved groups joined: 224.0.0.9
  Outgoing access list is not set
  Inbound access list is 1
! Lines omitted for brevity
```

The output of these commands shows two items of note. The first line of output in this case notes the type (standard) and the number. If more than one ACL existed, you would see multiple stanzas of output, one per ACL, each with a heading line like this one. Next, these commands list packet counts for the number of packets that the router has matched with each command. For example, 107 packets so far have matched the first line in the ACL.

Finally, the end of the example lists the **show ip interface** command output. This command lists, among many other items, the number or name of any IP ACL enabled on the interface per the **ip access-group** interface subcommand.

Standard Numbered ACL Example 2

For the second example, use Figure 2-8, and imagine your boss gives you some requirements hurriedly in the hall. At first, he tells you he wants to filter packets going from the servers on the right toward the clients on the left. Then, he says he wants you to allow access for hosts A, B, and other hosts in their same subnet to server S1, but deny access to that server to the hosts in host C's subnet. Then, he tells you that, additionally, hosts in host A's subnet should be denied access to server S2, but hosts in host C's subnet should be allowed access to server S2—all by filtering packets going right to left only. He then tells you to put the ACL inbound on R2's F0/0 interface.

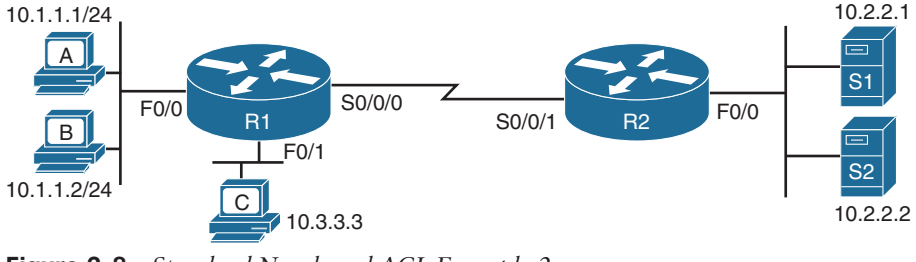


Figure 2-8 *Standard Numbered ACL Example 2*

If you cull through all the boss's comments, the requirements might be reduced to the following:

1. Enable the ACL inbound on R2's F0/0 interface.
2. Permit packets from server S1 going to hosts in A's subnet.
3. Deny packets from server S1 going to hosts in C's subnet.
4. Permit packets from server S2 going to hosts in C's subnet.
5. Deny packets from server S2 going to hosts in A's subnet.
6. (There was no comment about what to do by default; use the implied **deny all** default.)

As it turns out, you cannot do everything your boss asked with a standard ACL. For example, consider the obvious command for requirement number 2: **access-list 2 permit 10.2.2.1**. That permits all traffic whose source IP is 10.2.2.1 (server S1). The very next requirement asks you to filter (deny) packets sourced from that same IP address! Even if you added another command that checked for source IP address 10.2.2.1, the router would never get to it, because routers use first-match logic when searching the ACL. You cannot check both the destination and source IP address, because standard ACLs cannot check the destination IP address.

To solve this problem, you should get a new boss! No, seriously, you have to rethink the problem and change the rules. In real life, you would probably use an extended ACL instead, which lets you check both the source and destination IP address.

For the sake of practicing another standard ACL, imagine your boss lets you change the requirements. First, you will use two outbound ACLs, both on Router R1. Each ACL will permit traffic from a single server to be forwarded onto that connected LAN, with the following modified requirements:

1. Using an outbound ACL on R1's F0/0 interface, permit packets from server S1, and deny all other packets.
2. Using an outbound ACL on R1's F0/1 interface, permit packets from server S2, and deny all other packets.

Example 2-3 shows the configuration that completes these requirements.

Example 2-3 *Alternative Configuration in Router R1*

```
access-list 2 remark This ACL permits server S1 traffic to host A's subnet
access-list 2 permit 10.2.2.1
!
access-list 3 remark This ACL permits server S2 traffic to host C's subnet
access-list 3 permit 10.2.2.2
!
interface F0/0
 ip access-group 2 out
!
interface F0/1
 ip access-group 3 out
```

As highlighted in the example, the solution with ACL number 2 permits all traffic from server S1, with that logic enabled for packets exiting R1's F0/0 interface. All other traffic will be discarded because of the implied **deny all** at the end of the ACL. In addition, ACL 3 permits traffic from server S2, which is then permitted to exit R1's F0/1 interface. Also, note that the solution shows the use of the **access-list remark** parameter, which allows you to leave text documentation that stays with the ACL.

NOTE When routers apply an ACL to filter packets in the outbound direction, as shown in Example 2-3, the router checks packets that it routes against the ACL. However, a router does not filter packets that the router itself creates with an outbound ACL. Examples of those packets include routing protocol messages and packets sent by the **ping** and **traceroute** commands on that router.

Troubleshooting and Verification Tips

Troubleshooting IPv4 ACLs requires some attention to detail. In particular, you have to be ready to look at the address and wildcard mask and confidently predict the addresses matched by those two combined parameters. The upcoming practice problems a little later in this chapter can help prepare you for that part of the work. But a few other tips can help you verify and troubleshoot ACL problems on the exams as well.

First, you can tell if the router is matching packets or not with a couple of tools. Example 2-2 already showed that IOS keeps statistics about the packets matched by each line of an ACL. In addition, if you add the **log** keyword to the end of an **access-list** command, IOS then issues log messages with occasional statistics about matches of that particular line of the ACL. Both the statistics and the log messages can be helpful in deciding which line in the ACL is being matched by a packet.

For example, Example 2-4 shows an updated version of ACL 2 from Example 2-3, this time with the **log** keyword added. The bottom of the example then shows a typical log message, this one showing the resulting match based on a packet with source IP address 10.2.2.1 (as matched with the ACL), to destination address 10.1.1.1.

Example 2-4 *Creating Log Messages for ACL Statistics*

```
R1# show running-config
! lines removed for brevity
access-list 2 remark This ACL permits server S1 traffic to host A's subnet
access-list 2 permit 10.2.2.1 log
!
interface F0/0
 ip access-group 2 out

R1#
Feb 4 18:30:24.082: %SEC-6-IPACCESSLOGNP: list 2 permitted 0 10.2.2.1 -> 10.1.1.1, 1
packet
```

When you troubleshoot an ACL for the first time, before getting into the details of the matching logic, take the time to think about both the interface on which the ACL is enabled and the direction of packet flow. Sometimes, the matching logic is perfect—but the ACL

has been enabled on the wrong interface, or for the wrong direction, to match the packets as configured for the ACL.

For example, Figure 2-9 repeats the same ACL shown earlier in Figure 2-7. The first line of that ACL matches the specific host address 10.1.1.1. If that ACL exists on Router R2, placing that ACL as an inbound ACL on R2's S0/0/1 interface can work, because packets sent by host 10.1.1.1—on the left side of the figure—can enter R2's S0/0/1 interface. However, if R2 enables ACL 1 on its F0/0 interface, for inbound packets, the ACL will never match a packet with source IP address 10.1.1.1, because packets sent by host 10.1.1.1 will never enter that interface. Packets sent by 10.1.1.1 will exit R2's F0/0 interface, but never enter it, just because of the network topology.

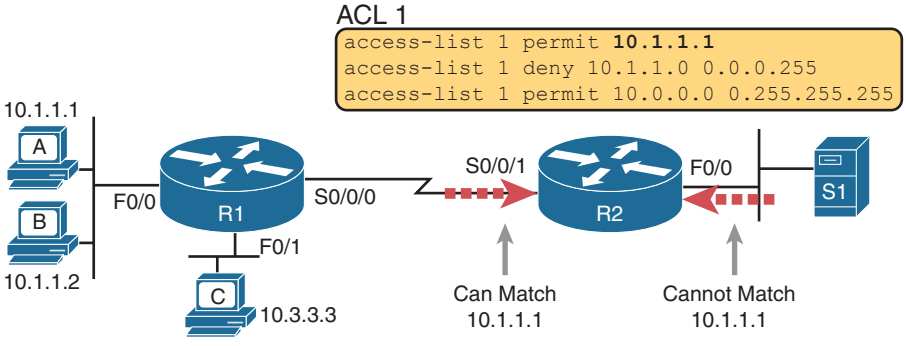


Figure 2-9 *Example of Checking the Interface and Direction for an ACL*

Practice Applying Standard IP ACLs

Some CCNA topics, like ACLs, simply require more drills and practice than others. ACLs require you to think of parameters to match ranges of numbers, and that of course requires some use of math and some use of processes.

This section provides some practice problems and tips, from two perspectives. First, this section asks you to build one-line standard ACLs to match some packets. Second, this section asks you to interpret existing ACL commands to describe what packets the ACL will match. Both skills are useful for the exams.

Practice Building access-list Commands

In this section, practice getting comfortable with the syntax of the **access-list** command, particularly with choosing the correct matching logic. These skills will be helpful when reading about extended and named ACLs in the next chapter.

First, the following list summarizes some important tips to consider when choosing matching parameters to any **access-list** command:

Key Topic

- To match a specific address, just list the address.
- To match any and all addresses, use the **any** keyword.
- To match based only on the first one, two, or three octets of an address, use the 0.255.255.255, 0.0.255.255, and 0.0.0.255 WC masks, respectively. Also, make the source (address) parameter have 0s in the wildcard octets (those octets with 255 in the wildcard mask).

- To match a subnet, use the subnet ID as the source, and find the WC mask by subtracting the DDN subnet mask from 255.255.255.255.

Table 2-2 lists the criteria for several practice problems. Your job: Create a one-line standard ACL that matches the packets. The answers are listed in the section "Answers to Earlier Practice Problems," later in this chapter.

Table 2-2 Building One-Line Standard ACLs: Practice

Problem	Criteria
1	Packets from 172.16.5.4
2	Packets from hosts with 192.168.6 as the first three octets
3	Packets from hosts with 192.168 as the first two octets
4	Packets from any host
5	Packets from subnet 10.1.200.0/21
6	Packets from subnet 10.1.200.0/27
7	Packets from subnet 172.20.112.0/23
8	Packets from subnet 172.20.112.0/26
9	Packets from subnet 192.168.9.64/28
10	Packets from subnet 192.168.9.64/30

Reverse Engineering from ACL to Address Range

In some cases, you may not be creating your own ACL. Instead, you may need to interpret some existing **access-list** commands. To answer these types of questions on the exams, you need to determine the range of IP addresses matched by a particular address/wildcard mask combination in each ACL statement.

Under certain assumptions that are reasonable for CCNA certifications, calculating the range of addresses matched by an ACL can be relatively simple. Basically, the range of addresses begins with the address configured in the ACL command. The range of addresses ends with the sum of the address field and the wildcard mask. That's it.

For example, with the command **access-list 1 permit 172.16.200.0 0.0.7.255**, the low end of the range is simply 172.16.200.0, taken directly from the command itself. Then, to find the high end of the range, just add this number to the WC mask, as follows:

```
172.16.200.0
+ 0.  0.   7.255
172.16.207.255
```

For this last bit of practice, look at the existing **access-list** commands in Table 2-3. In each case, make a notation about the exact IP address, or range of IP addresses, matched by the command.

Table 2-3 Finding IP Addresses/Ranges Matching by Existing ACLs

Problem	Commands for Which to Predict the Source Address Range
1	**access-list 1 permit 10.7.6.5**
2	**access-list 2 permit 192.168.4.0 0.0.0.127**
3	**access-list 3 permit 192.168.6.0 0.0.0.31**

Problem	Commands for Which to Predict the Source Address Range
4	access-list 4 permit 172.30.96.0 0.0.3.255
5	access-list 5 permit 172.30.96.0 0.0.0.63
6	access-list 6 permit 10.1.192.0 0.0.0.31
7	access-list 7 permit 10.1.192.0 0.0.1.255
8	access-list 8 permit 10.1.192.0 0.0.63.255

Interestingly, IOS lets the CLI user type an **access-list** command in configuration mode, and IOS will potentially change the address parameter before placing the command into the running-config file. This process of just finding the range of addresses matched by the **access-list** command expects that the **access-list** command came from the router, so that any such changes were complete.

The change IOS can make with an **access-list** command is to convert to 0 any octet of an address for which the wildcard mask's octet is 255. For example, with a wildcard mask of 0.0.255.255, IOS ignores the last two octets. IOS expects the address field to end with two 0s. If not, IOS still accepts the **access-list** command, but IOS changes the last two octets of the address to 0s. Example 2-5 shows an example, where the configuration shows address 10.1.1.1, but wildcard mask 0.0.255.255.

Example 2-5 *IOS Changing the Address Field in an* **access-list** *Command*

```
R2# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)# access-list 21 permit 10.1.1.1 0.0.255.255
R2(config)# ^Z
R2#
R2# show ip access-lists
Standard IP access list 21
    10 permit 10.1.0.0, wildcard bits 0.0.255.255
```

The math to find the range of addresses relies on the fact that either the command is fully correct or that IOS has already set these address octets to 0, as shown in the example.

NOTE The most useful WC masks, in binary, do not interleave 0s and 1s. This book assumes the use of only these types of WC masks. However, Cisco IOS allows WC masks that interleave 0s and 1s, but using these WC masks breaks the simple method of calculating the range of addresses. As you progress through to CCIE studies, be ready to dig deeper to learn how to determine what an ACL matches.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 2-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 2-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review command tables		Book

Review All the Key Topics

Key Topic

Table 2-5 Key Topics for Chapter 2

Key Topic Element	Description	Page Number
Paragraph	Summary of the general rule of the location and direction for an ACL	27
Figure 2-3	Summary of four main categories of IPv4 ACLs in Cisco IOS	29
Paragraph	Summary of first-match logic used by all ACLs	29
List	Wildcard mask logic for decimal 0 and 255	32
List	Wildcard mask logic to match a subnet	33
List	Steps to plan and implement a standard IP ACL	34
List	Tips for creating matching logic for the source address field in the **access-list** command	39

Key Terms You Should Know

standard access list, wildcard mask

Additional Practice for This Chapter's Processes

For additional practice with analyzing subnets, you may do the same set of practice problems using your choice of tools:

Application: Use the two ACL practice exercise applications listed on the companion website.

PDF: Alternatively, practice the same problems found in these apps using online Appendix E, "Practice for Chapter 2: Basic IPv4 Access Control Lists."

Command References

Tables 2-6 and 2-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 2-6 Chapter 2 Configuration Command Reference

Command	Description
access-list *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]	Global command for standard numbered access lists. Use a number between 1 and 99 or 1300 and 1999, inclusive.

Command	Description
access-list *access-list-number* **remark** *text*	Command that defines a remark to help you remember what the ACL is supposed to do.
ip access-group *number* {**in** | **out**}	Interface subcommand to enable access lists.

Table 2-7 Chapter 2 EXEC Command Reference

Command	Description
show ip interface [*type number*]	Includes a reference to the access lists enabled on the interface
show access-lists [*access-list-number* | *access-list-name*]	Shows details of configured access lists for all protocols
show ip access-lists [*access-list-number* | *access-list-name*]	Shows IP access lists

Answers to Earlier Practice Problems

Table 2-8 lists the answers to the problems listed earlier in Table 2-2.

Table 2-8 Building One-Line Standard ACLs: Answers

Problem	Answers
1	access-list 1 permit 172.16.5.4
2	access-list 2 permit 192.168.6.0 0.0.0.255
3	access-list 3 permit 192.168.0.0 0.0.255.255
4	access-list 4 permit any
5	access-list 5 permit 10.1.200.0 0.0.7.255
6	access-list 6 permit 10.1.200.0 0.0.0.31
7	access-list 7 permit 172.20.112.0 0.0.1.255
8	access-list 8 permit 172.20.112.0 0.0.0.63
9	access-list 9 permit 192.168.9.64 0.0.0.15
10	access-list 10 permit 192.168.9.64 0.0.0.3

Table 2-9 lists the answers to the problems listed earlier in Table 2-3.

Table 2-9 Address Ranges for Problems in Table 2-3: Answers

Problem	Address Range
1	One address: 10.7.6.5
2	192.168.4.0 – 192.168.4.127
3	192.168.6.0 – 192.168.6.31
4	172.30.96.0 – 172.30.99.255
5	172.30.96.0 – 172.30.96.63
6	10.1.192.0 – 10.1.192.31
7	10.1.192.0 – 10.1.193.255
8	10.1.192.0 – 10.1.255.255

CHAPTER 3

Advanced IPv4 Access Control Lists

This chapter covers the following exam topics:

5.0 Security Fundamentals

5.6 Configure and verify access control lists

IPv4 ACLs are either standard or extended ACLs, with standard ACLs matching only the source IP address, and extended matching a variety of packet header fields. At the same time, IP ACLs are either numbered or named. Figure 3-1 shows the categories and the main features of each as introduced in the previous chapter.

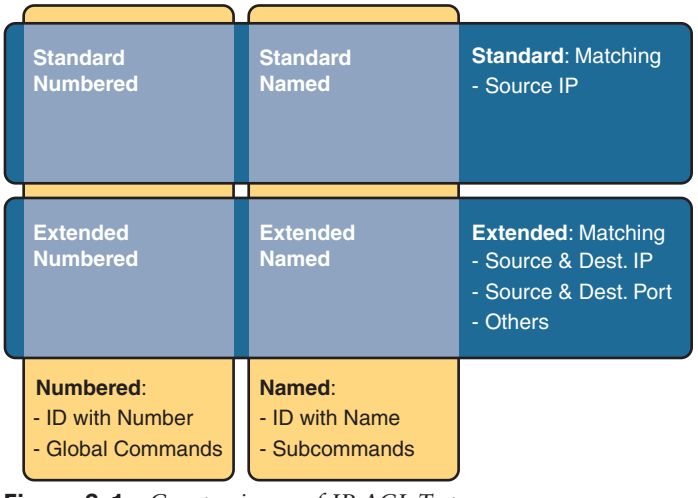


Figure 3-1 *Comparisons of IP ACL Types*

This chapter discusses the other three categories of ACLs beyond standard numbered IP ACLs and ends with a few miscellaneous features to secure Cisco routers and switches.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 3-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Extended IP Access Control Lists	1–3
Named ACLs and ACL Editing	4–6

1. Which of the following fields cannot be compared based on an extended IP ACL? (Choose two answers.)
   a. Protocol
   b. Source IP address
   c. Destination IP address
   d. TOS byte
   e. URL
   f. Filename for FTP transfers
2. Which of the following **access-list** commands permit packets going from host 10.1.1.1 to all web servers whose IP addresses begin with 172.16.5? (Choose two answers.)
   a. **access-list 101 permit tcp host 10.1.1.1 172.16.5.0 0.0.0.255 eq www**
   b. **access-list 1951 permit ip host 10.1.1.1 172.16.5.0 0.0.0.255 eq www**
   c. **access-list 2523 permit ip host 10.1.1.1 eq www 172.16.5.0 0.0.0.255**
   d. **access-list 2523 permit tcp host 10.1.1.1 eq www 172.16.5.0 0.0.0.255**
   e. **access-list 2523 permit tcp host 10.1.1.1 172.16.5.0 0.0.0.255 eq www**
3. Which of the following **access-list** commands permits packets going to any web client from all web servers whose IP addresses begin with 172.16.5?
   a. **access-list 101 permit tcp host 10.1.1.1 172.16.5.0 0.0.0.255 eq www**
   b. **access-list 1951 permit ip host 10.1.1.1 172.16.5.0 0.0.0.255 eq www**
   c. **access-list 2523 permit tcp any eq www 172.16.5.0 0.0.0.255**
   d. **access-list 2523 permit tcp 172.16.5.0 0.0.0.255 eq www 172.16.5.0 0.0.0.255**
   e. **access-list 2523 permit tcp 172.16.5.0 0.0.0.255 eq www any**
4. In a router running a recent IOS version (at least version 15.0), an engineer needs to delete the second line in ACL 101, which currently has four commands configured. Which of the following options could be used? (Choose two answers.)
   a. Delete the entire ACL and reconfigure the three ACL statements that should remain in the ACL.
   b. Delete one line from the ACL using the **no access-list…** global command.
   c. Delete one line from the ACL by entering ACL configuration mode for the ACL and then deleting only the second line based on its sequence number.
   d. Delete the last three lines from the ACL from global configuration mode, and then add the last two statements back into the ACL.

5. Refer to the following command output, which details an ACL enabled on port G0/0 for the inbound direction. Which answers list a configuration mode and command that would result in the deletion of the line that matches subnet 172.16.1.0/24? (Choose two answers.)

```
show ip access-lists dikta-list
Standard IP access list dikta-list
    10 permit 172.16.1.0, wildcard bits 0.0.0.255
    20 permit 172.16.2.0, wildcard bits 0.0.0.255
    30 permit 172.16.3.0, wildcard bits 0.0.0.255
```

   a. In global config mode: **no 10**
   b. In interface G0/0 config mode: **no 10**
   c. In ACL dikta-list config mode: **no 10**
   d. In ACL dikta-list config mode: **no permit 172.16.1.0 0.0.0.255**
   e. In global config mode: **no permit 172.16.1.0 0.0.0.255**

6. An engineer configures an ACL but forgets to save the configuration. At that point, which of the following commands display the configuration of an IPv4 ACL, including line numbers? (Choose two answers.)

   a. **show running-config**
   b. **show startup-config**
   c. **show ip access-lists**
   d. **show access-lists**

Foundation Topics

Extended Numbered IP Access Control Lists

Extended IP access lists have many similarities compared to the standard numbered IP ACLs discussed in the previous chapter. Just like standard IP ACLs, you enable extended access lists on interfaces for packets either entering or exiting the interface. IOS searches the list sequentially. Extended ACLs also use first-match logic, because the router stops the search through the list as soon as the first statement is matched, taking the action defined in the first-matched statement. All these features are also true of standard numbered access lists (and named ACLs).

Extended ACLs differ from standard ACLs mostly because of the larger variety of packet header fields that can be used to match a packet. One extended ACE (ACL statement) can examine multiple parts of the packet headers, requiring that all the parameters be matched correctly to match that one ACE. That powerful matching logic makes extended access lists both more useful and more complex than standard IP ACLs.

Matching the Protocol, Source IP, and Destination IP

Like standard numbered IP ACLs, extended numbered IP ACLs also use the **access-list** global command. The syntax is identical, at least up through the **permit** or **deny** keyword. At that point, the command lists matching parameters, and those differ, of course. In particular, the extended ACL **access-list** command requires three matching parameters: the IP protocol type, the source IP address, and the destination IP address.

The IP header's Protocol field identifies the header that follows the IP header. Figure 3-2 shows the location of the IP Protocol field, the concept of it pointing to the type of header that follows, along with some details of the IP header for reference.

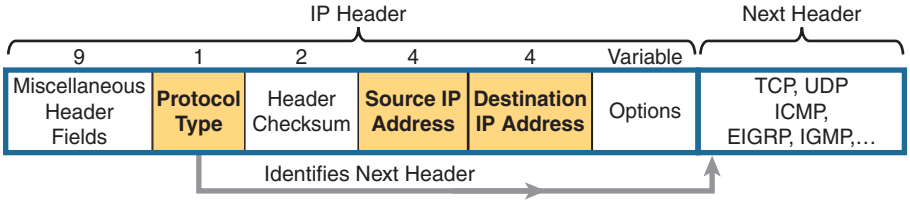


Figure 3-2 *IP Header, with Focus on Required Fields in Extended IP ACLs*

IOS requires that you configure parameters for the three highlighted parts of Figure 3-2. For the protocol type, you simply use a keyword, such as **tcp**, **udp**, or **icmp**, matching IP packets that happen to have a TCP, UDP, or ICMP header, respectively, following the IP header. Or you can use the keyword **ip**, which means "all IPv4 packets." You also must configure some values for the source and destination IP address fields that follow; these fields use the same syntax and options for matching the IP addresses as discussed in Chapter 2, "Basic IPv4 Access Control Lists." Figure 3-3 shows the syntax.

Key Topic

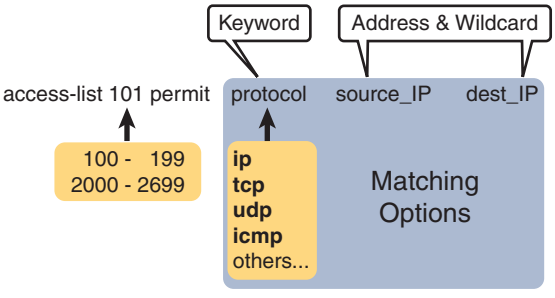


Figure 3-3 *Extended ACL Syntax, with Required Fields*

NOTE When matching IP addresses in the source and destination fields, there is one difference with standard ACLs: When matching a specific IP address, the extended ACL requires the use of the **host** keyword. You cannot simply list the IP address alone.

Table 3-2 lists several sample **access-list** commands that use only the required matching parameters. Feel free to cover the right side and use the table for an exercise, or just review the explanations to get an idea for the logic in some sample commands.

Table 3-2 Extended **access-list** Commands and Logic Explanations

access-list Statement	What It Matches
access-list 101 deny tcp any any	Any IP packet that has a TCP header
access-list 101 deny udp any any	Any IP packet that has a UDP header
access-list 101 deny icmp any any	Any IP packet that has an ICMP header
access-list 101 deny ip host 1.1.1.1 host 2.2.2.2	All IP packets from host 1.1.1.1 going to host 2.2.2.2, regardless of the header after the IP header
access-list 101 deny udp 1.1.1.0 0.0.0.255 any	All IP packets that have a UDP header following the IP header, from subnet 1.1.1.0/24, and going to any destination

The last entry in Table 3-2 helps make an important point about how IOS processes extended ACLs:

Key Topic

> In an extended ACL **access-list** command, all the matching parameters must match the packet for the packet to match the command.

For example, in that last example from Table 3-2, the command checks for UDP, a source IP address from subnet 1.1.1.0/24, and any destination IP address. If a packet with source IP address 1.1.1.1 were examined, it would match the source IP address check, but if it had a TCP header instead of UDP, it would not match this **access-list** command. All parameters must match.

Matching TCP and UDP Port Numbers

Extended ACLs can also examine parts of the TCP and UDP headers, particularly the source and destination port number fields. The port numbers identify the application that sends or receives the data.

The most useful ports to check are the well-known ports used by servers. For example, web servers use well-known port 80 by default. Figure 3-4 shows the location of the port numbers in the TCP header, following the IP header.

Key Topic

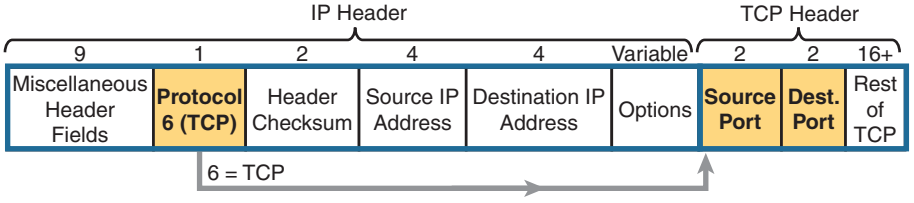


Figure 3-4 *IP Header, Followed by a TCP Header and Port Number Fields*

When an extended ACL command includes either the **tcp** or **udp** keyword, that command can optionally reference the source and/or destination port. To make these comparisons, the syntax uses keywords for equal, not equal, less than, greater than, and for a range of port numbers. In addition, the command can use either the literal decimal port numbers or more convenient keywords for some well-known application ports. Figure 3-5 shows the positions of the source and destination port fields in the **access-list** command and these port number keywords.

Key Topic

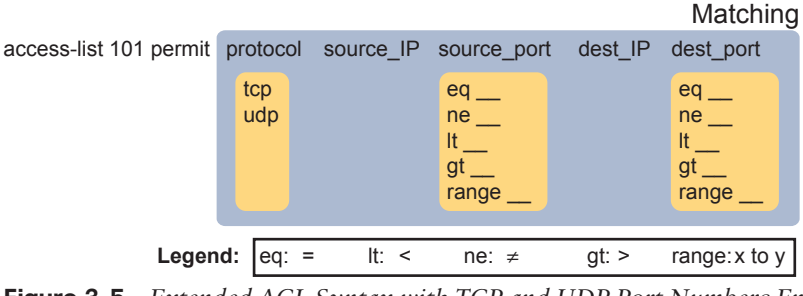


Figure 3-5 *Extended ACL Syntax with TCP and UDP Port Numbers Enabled*

Answers to the "Do I Know This Already?" quiz:

1 E, F **2** A, E **3** E **4** A, C **5** C, D **6** C, D

For example, consider the simple network shown in Figure 3-6. The FTP server sits on the right, with the client on the left. The figure shows the syntax of an ACL that matches the following:

- Packets that include a TCP header
- Packets sent from the client subnet
- Packets sent to the server subnet
- Packets with TCP destination port 21 (FTP server control port)

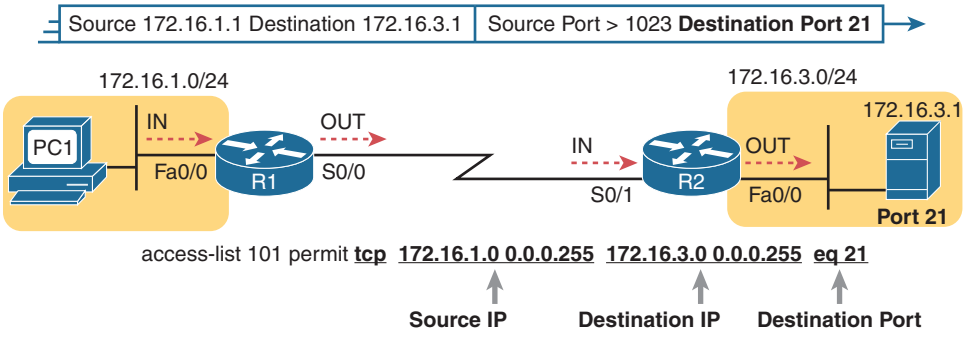


Figure 3-6 *Filtering Packets Based on Destination Port*

To fully appreciate the matching of the destination port with the **eq 21** parameters, consider packets moving from left to right, from PC1 to the server. Assuming the server uses well-known port 21 (FTP control port), the packet's TCP header has a destination port value of 21. The ACL syntax includes the **eq 21** parameters after the destination IP address. The position after the destination address parameters is important: that position identifies the fact that the **eq 21** parameters should be compared to the packet's destination port. As a result, the ACL statement shown in Figure 3-6 would match this packet and the destination port of 21 if used in any of the four locations implied by the four dashed arrowed lines in the figure.

Conversely, Figure 3-7 shows the reverse flow, with a packet sent by the server back toward PC1. In this case, the packet's TCP header has a source port of 21, so the ACL must check the source port value of 21, and the ACL must be located on different interfaces. In this case, the **eq 21** parameters follow the source address field but come before the destination address field.

Key Topic

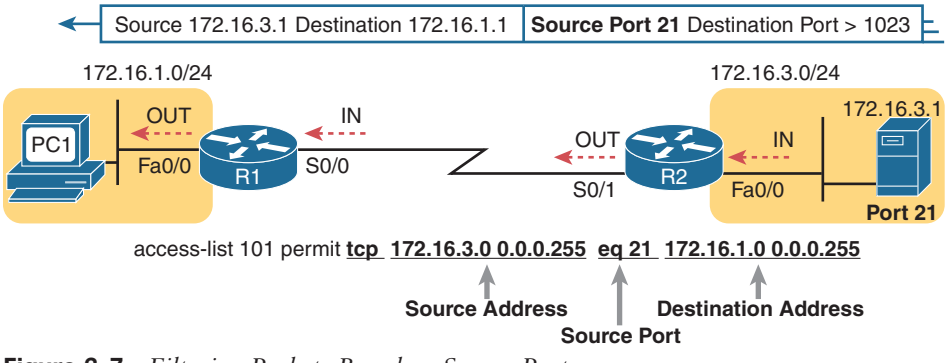


Figure 3-7 *Filtering Packets Based on Source Port*

When examining ACLs that match port numbers, first consider the location and direction in which the ACL will be applied. That direction determines whether the packet is being sent to the server or from the server. At that point, you can decide whether you need to check the source or destination port in the packet. For reference, Table 3-3 lists many of the popular port numbers and their transport layer protocols and applications. Note that the syntax of the **access-list** commands accepts both the port numbers and a shorthand version of the application name.

Table 3-3 Popular Applications and Their Well-Known Port Numbers

Port Number(s)	Protocol	Application	access-list Command Keyword
20	TCP	FTP data	**ftp-data**
21	TCP	FTP control	**ftp**
22	TCP	SSH	**—**
23	TCP	Telnet	**telnet**
25	TCP	SMTP	**smtp**
53	UDP, TCP	DNS	**domain**
67	UDP	DHCP Server	**bootps**
68	UDP	DHCP Client	**bootpc**
69	UDP	TFTP	**tftp**
80	TCP	HTTP (WWW)	**www**
110	TCP	POP3	**pop3**
161	UDP	SNMP	**snmp**
443	TCP	SSL	**—**
514	UDP	Syslog	**—**
16,384–32,767	UDP	RTP (voice, video)	**—**

Table 3-4 lists several sample **access-list** commands that match based on port numbers. Cover the right side of the table, and try to characterize the packets matched by each command. Then check the right side of the table to see if you agree with the assessment.

Table 3-4 Extended **access-list** Command Examples and Logic Explanations

access-list Statement	What It Matches
access-list 101 deny tcp any gt 49151 host 10.1.1.1 eq 23	Packets with a TCP header, any source IP address, with a source port greater than (gt) 49151, a destination IP address of exactly 10.1.1.1, and a destination port equal to (eq) 23.
access-list 101 deny tcp any host 10.1.1.1 eq 23	The same as the preceding example, but any source port matches, because that parameter is omitted in this case.
access-list 101 deny tcp any host 10.1.1.1 eq telnet	The same as the preceding example. The **telnet** keyword is used instead of port 23.
access-list 101 deny udp 1.0.0.0 0.255.255.255 lt 1023 any	A packet with a source in network 1.0.0.0/8, using UDP with a source port less than (lt) 1023, with any destination IP address.

Extended IP ACL Configuration

Because extended ACLs can match so many different fields in the various headers in an IP packet, the command syntax cannot be easily summarized in a single generic command. However, the two commands in Table 3-5 summarize the syntax options as covered in this book.

Table 3-5 Extended IP Access List Configuration Commands

Command	Configuration Mode and Description
access-list *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log** | **log-input**]	Global command for extended numbered access lists. Use a number between 100 and 199 or 2000 and 2699, inclusive.
access-list *access-list-number* {**deny** | **permit**} {**tcp** | **udp**} *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established**] [**log**]	A version of the **access-list** command with parameters specific to TCP and/or UDP.

The configuration process for extended ACLs mostly matches the same process used for standard ACLs. You must choose the location and direction in which to enable the ACL, particularly the direction, so that you can characterize whether certain addresses and ports will be either the source or destination. Configure the ACL using **access-list** commands, and when complete, then enable the ACL using the same **ip access-group** command used with standard ACLs. All these steps mirror what you do with standard ACLs; however, when configuring, keep the following differences in mind:

Key Topic

- Place extended ACLs as close as possible to the source of the packets that will be filtered. Filtering close to the source of the packets saves some bandwidth.
- Remember that all fields in one **access-list** command must match a packet for the packet to be considered to match that **access-list** statement.
- Use numbers of 100–199 and 2000–2699 on the **access-list** commands; no one number is inherently better than another.

Extended IP Access Lists: Example 1

This example focuses on understanding basic syntax. In this case, the ACL denies Bob access to all FTP servers on R1's Ethernet, and it denies Larry access to Server1's web server. Figure 3-8 shows the network topology; Example 3-1 shows the configuration on R1.

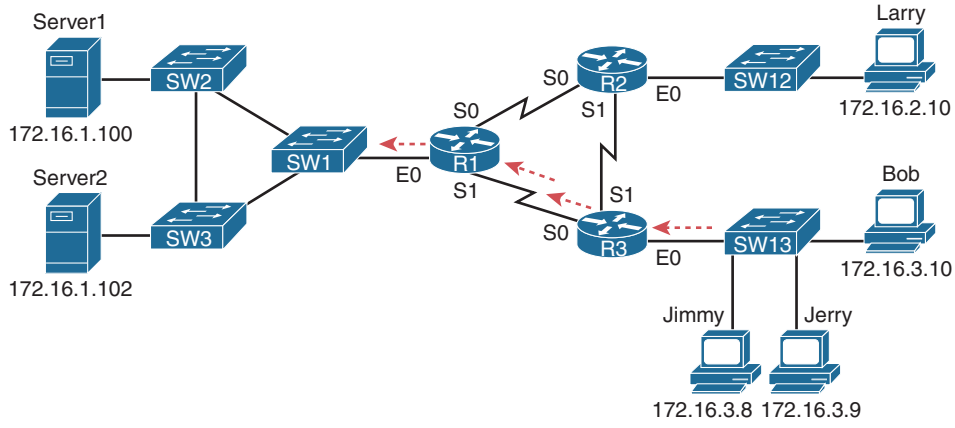


Figure 3-8 *Network Diagram for Extended Access List Example 1*

Example 3-1 *R1's Extended Access List: Example 1*

```
interface Serial0
 ip address 172.16.12.1 255.255.255.0
 ip access-group 101 in
!
interface Serial1
 ip address 172.16.13.1 255.255.255.0
 ip access-group 101 in
!
access-list 101 remark Stop Bob to FTP servers, and Larry to Server1 web
access-list 101 deny tcp host 172.16.3.10 172.16.1.0 0.0.0.255 eq ftp
access-list 101 deny tcp host 172.16.2.10 host 172.16.1.100 eq www
access-list 101 permit ip any any
```

The first ACL statement prevents Bob's access to FTP servers in subnet 172.16.1.0. The second statement prevents Larry's access to web services on Server1. The final statement permits all other traffic.

If we focus on the syntax for a moment, we can see several new items to review. First, the access-list number for extended access lists falls in the range of 100 to 199 or 2000 to 2699. Following the **permit** or **deny** action, the *protocol* parameter defines whether you want to check for all IP packets or specific headers, such as TCP or UDP headers. When you check for TCP or UDP port numbers, you must specify the TCP or UDP protocol. Both FTP and the web use TCP.

This example uses the **eq** parameter, meaning "equals," to check the destination port numbers for FTP control (keyword **ftp**) and HTTP traffic (keyword **www**). You can use the numeric values—or, for the more popular options, a more obvious text version is valid. (If you were to type **eq 80**, the config would show **eq www**.)

This example enables the ACL in two places on R1: inbound on each serial interface. These locations achieve the goal of the ACL. However, that initial placement was made to make the point that Cisco suggests that you locate them as close as possible to the source of the packet. Therefore, Example 3-2 achieves the same goal as Example 3-1 of stopping Bob's access to FTP servers at the main site, and it does so with an ACL on R3.

Example 3-2 *R3's Extended Access List Stopping Bob from Reaching FTP Servers Near R1*

```
interface Ethernet0
 ip address 172.16.3.1 255.255.255.0
 ip access-group 103 in

access-list 103 remark deny Bob to FTP servers in subnet 172.16.1.0/24
access-list 103 deny tcp host 172.16.3.10 172.16.1.0 0.0.0.255 eq ftp
access-list 103 permit ip any any
```

The new configuration on R3 meets the goals to filter Bob's traffic, while also meeting the overarching design goal of keeping the ACL close to the source of the packets. ACL 103 on R3 looks a lot like ACL 101 on R1 from Example 3-1, but this time, the ACL does not

bother to check for the criteria to match Larry's traffic, because Larry's traffic will never enter R3's Ethernet 0 interface. ACL 103 filters Bob's FTP traffic to destinations in subnet 172.16.1.0/24, with all other traffic entering R3's E0 interface making it into the network.

Extended IP Access Lists: Example 2

Example 3-3, based on the network shown in Figure 3-9, shows another example of how to use extended IP access lists. This example uses the following criteria:

- Sam is not allowed access to the subnet of Bugs or Daffy.
- Hosts on the Seville Ethernet are not allowed access to hosts on the Yosemite Ethernet.
- All other combinations are allowed.

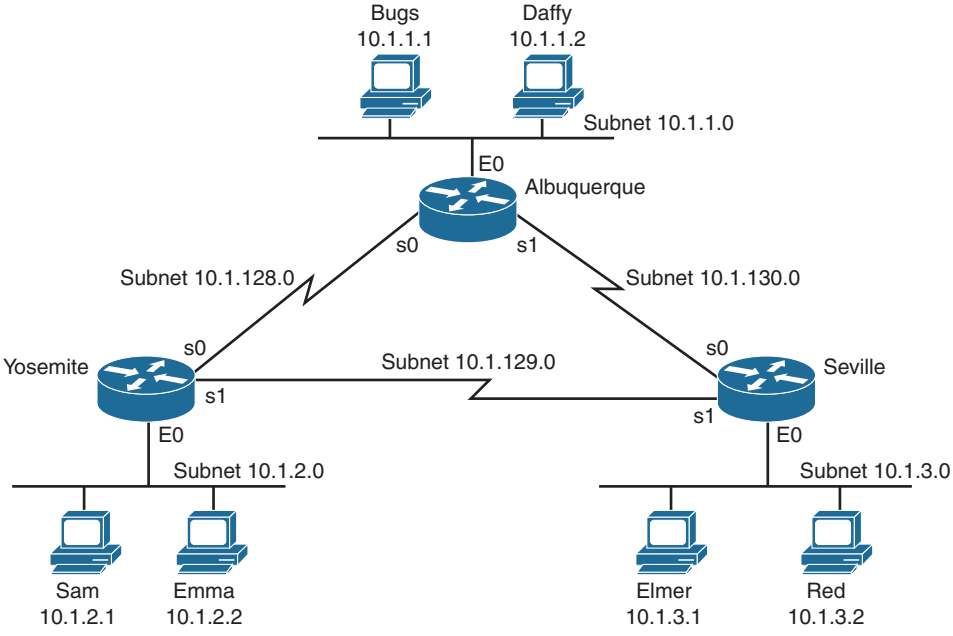


Figure 3-9 *Network Diagram for Extended Access List Example 2*

Example 3-3 *Yosemite Configuration for Extended Access List Example 2*

```
interface ethernet 0
 ip access-group 110 in
!
access-list 110 deny ip host 10.1.2.1 10.1.1.0 0.0.0.255
access-list 110 deny ip 10.1.2.0 0.0.0.255 10.1.3.0 0.0.0.255
access-list 110 permit ip any any
```

This configuration solves the problem with few statements while keeping to the Cisco design guideline of placing extended ACLs as close as possible to the source of the traffic. The ACL filters packets that enter Yosemite's E0 interface, which is the first router interface that packets sent by Sam enter. If the route between Yosemite and the other subnets changes over time, the ACL still applies. Also, the filtering mandated by the second requirement

(to disallow Seville's LAN hosts from accessing Yosemite's) is met by the second **access-list** statement. Stopping packet flow from Yosemite's LAN subnet to Seville's LAN subnet stops effective communication between the two subnets. Alternatively, the opposite logic could have been configured at Seville.

Practice Building access-list Commands

Table 3-6 supplies a practice exercise to help you get comfortable with the syntax of the extended **access-list** command, particularly with choosing the correct matching logic. Your job: create a one-line extended ACL that matches the packets. The answers are in the section "Answers to Earlier Practice Problems," later in this chapter. Note that if the criteria mention a particular application protocol, for example, "web client," that means to specifically match for that application protocol.

Table 3-6 Building One-Line Extended ACLs: Practice

Problem	Criteria
1	From web client 10.1.1.1, sent to a web server in subnet 10.1.2.0/24.
2	From Telnet client 172.16.4.3/25, sent to a Telnet server in subnet 172.16.3.0/25. Match all hosts in the client's subnet as well.
3	ICMP messages from the subnet in which 192.168.7.200/26 resides to all hosts in the subnet where 192.168.7.14/29 resides.
4	From web server 10.2.3.4/23's subnet to clients in the same subnet as host 10.4.5.6/22.
5	From Telnet server 172.20.1.0/24's subnet, sent to any host in the same subnet as host 172.20.44.1/23.
6	From web client 192.168.99.99/28, sent to a web server in subnet 192.168.176.0/28. Match all hosts in the client's subnet as well.
7	ICMP messages from the subnet in which 10.55.66.77/25 resides to all hosts in the subnet where 10.66.55.44/26 resides.
8	Any and every IPv4 packet.

Named ACLs and ACL Editing

Now that you have a good understanding of the core concepts in IOS IP ACLs, this section examines a few enhancements to IOS support for ACLs: named ACLs and ACL editing with sequence numbers. Although both features are useful and important, neither adds any function as to what a router can and cannot filter. Instead, named ACLs and ACL sequence numbers make it easier to remember ACL names and edit existing ACLs when an ACL needs to change.

Named IP Access Lists

Named IP ACLs have many similarities with numbered IP ACLs. They can be used for filtering packets, plus for many other purposes. They can match the same fields as well: standard numbered ACLs can match the same fields as a standard named ACL, and extended numbered ACLs can match the same fields as an extended named ACL.

Of course, there are differences between named and numbered ACLs. Named ACLs originally had three big differences compared to numbered ACLs:

Key Topic

- Using names instead of numbers to identify the ACL, making it easier to remember the reason for the ACL
- Using ACL subcommands, not global commands, to define the action and matching parameters
- Using ACL editing features that allow the CLI user to delete individual lines from the ACL and insert new lines

You can easily learn named ACL configuration by just converting numbered ACLs to use the equivalent named ACL configuration. Figure 3-10 shows just such a conversion, using a simple three-line standard ACL number 1. To create the three **permit** subcommands for the named ACL, you literally copy parts of the three numbered ACL commands, beginning with the **permit** keyword.

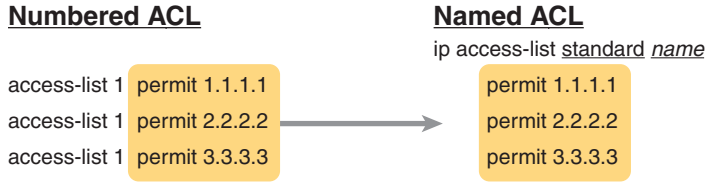


Figure 3-10 *Named ACL Versus Numbered ACL Configuration*

The only truly new part of the named ACL configuration is the **ip access-list** global configuration command. This command defines whether an ACL is a standard or extended ACL and defines the name. It also moves the user to ACL configuration mode, as shown in upcoming Example 3-4. Once in ACL configuration mode, you configure **permit**, **deny**, and **remark** commands that mirror the syntax of numbered ACL **access-list** commands. If you're configuring a standard named ACL, these commands match the syntax of standard numbered ACLs; if you're configuring extended named ACLs, they match the syntax of extended numbered ACLs.

Example 3-4 shows the configuration of a named extended ACL. Pay particular attention to the configuration mode prompts, which show ACL configuration mode.

Example 3-4 *Named Access List Configuration*

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# ip access-list extended barney
Router(config-ext-nacl)# permit tcp host 10.1.1.2 eq www any
Router(config-ext-nacl)# deny udp host 10.1.1.1 10.1.2.0 0.0.0.255
Router(config-ext-nacl)# deny ip 10.1.3.0 0.0.0.255 10.1.2.0 0.0.0.255
Router(config-ext-nacl)# deny ip 10.1.2.0 0.0.0.255 10.2.3.0 0.0.0.255
Router(config-ext-nacl)# permit ip any any
Router(config-ext-nacl)# interface serial1
Router(config-if)# ip access-group barney out
Router(config-if)# ^Z
Router# show running-config
Building configuration...
```

```
Current configuration:

! lines omitted for brevity

interface serial 1
 ip access-group barney out
!
ip access-list extended barney
 permit tcp host 10.1.1.2 eq www any
 deny   udp host 10.1.1.1 10.1.2.0 0.0.0.255
 deny   ip 10.1.3.0 0.0.0.255 10.1.2.0 0.0.0.255
 deny   ip 10.1.2.0 0.0.0.255 10.2.3.0 0.0.0.255
 permit ip any any
```

Example 3-4 begins with the creation of an ACL named barney. The **ip access-list extended barney** command creates the ACL, naming it barney and placing the user in ACL configuration mode. This command also tells the IOS that barney is an extended ACL. Next, five different **permit** and **deny** statements define the matching logic and action to be taken upon a match. The **show running-config** command output lists the named ACL configuration before the single entry is deleted.

Named ACLs allow the user to delete and add new lines to the ACL from within ACL configuration mode. Example 3-5 shows how, with the **no deny ip…** command deleting a single entry from the ACL. Notice that the output of the **show access-list** command at the end of the example still lists the ACL, with four **permit** and **deny** commands instead of five.

Example 3-5 *Removing One Command from a Named ACL*

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# ip access-list extended barney
Router(config-ext-nacl)# no deny ip 10.1.2.0 0.0.0.255 10.2.3.0 0.0.0.255
Router(config-ext-nacl)# ^Z
Router# show access-list

Extended IP access list barney
    10 permit tcp host 10.1.1.2 eq www any
    20 deny   udp host 10.1.1.1 10.1.2.0 0.0.0.255
    30 deny   ip 10.1.3.0 0.0.0.255 10.1.2.0 0.0.0.255
    50 permit ip any any
```

Editing ACLs Using Sequence Numbers

Numbered ACLs have existed in IOS since the early days of Cisco routers and IOS; however, for many years, through many IOS versions, the ability to edit a numbered IP ACL was poor. For example, to simply delete a line from the ACL, the user had to delete the entire ACL and then reconfigure it.

The ACL editing feature uses an ACL sequence number that is added to each ACL **permit** or **deny** statement, with the numbers representing the sequence of statements in the ACL.

ACL sequence numbers provide the following features for both numbered and named ACLs:

Key Topic

New configuration style for numbered: Numbered ACLs use a configuration style like named ACLs, as well as the traditional style, for the same ACL; the new style is required to perform advanced ACL editing.

Deleting single lines: An individual ACL **permit** or **deny** statement can be deleted with a **no** *sequence-number* subcommand.

Inserting new lines: Newly added **permit** and **deny** commands can be configured with a sequence number before the **deny** or **permit** command, dictating the location of the statement within the ACL.

Automatic sequence numbering: IOS adds sequence numbers to commands as you configure them, even if you do not include the sequence numbers.

To take advantage of the ability to delete and insert lines in an ACL, both numbered and named ACLs must use the same overall configuration style and commands used for named ACLs. The only difference in syntax is whether a name or number is used. Example 3-6 shows the configuration of a standard numbered IP ACL, using this alternative configuration style. The example shows the power of the ACL sequence number for editing. In this example, the following occurs:

Step 1. Numbered ACL 24 is configured using this new-style configuration, with three **permit** commands.

Step 2. The **show ip access-lists** command shows the three **permit** commands with sequence numbers 10, 20, and 30.

Step 3. The engineer deletes only the second **permit** command using the **no 20** ACL subcommand, which simply refers to sequence number 20.

Step 4. The **show ip access-lists** command confirms that the ACL now has only two lines (sequence numbers 10 and 30).

Step 5. The engineer adds a new **deny** command to the beginning of the ACL, using the **5 deny 10.1.1.1** ACL subcommand.

Step 6. The **show ip access-lists** command again confirms the changes, this time listing three commands, sequence numbers 5, 10, and 30.

NOTE For this example, note that the user does not leave configuration mode, instead using the **do** command to tell IOS to issue the **show ip access-lists** EXEC command from configuration mode.

Example 3-6 *Editing ACLs Using Sequence Numbers*

```
! Step 1: The 3-line Standard Numbered IP ACL is configured.
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ip access-list standard 24
R1(config-std-nacl)# permit 10.1.1.0 0.0.0.255
R1(config-std-nacl)# permit 10.1.2.0 0.0.0.255
```

```
R1(config-std-nacl)# permit 10.1.3.0 0.0.0.255

! Step 2: Displaying the ACL's contents, without leaving configuration mode.
R1(config-std-nacl)# do show ip access-lists 24
Standard IP access list 24
    10 permit 10.1.1.0, wildcard bits 0.0.0.255
    20 permit 10.1.2.0, wildcard bits 0.0.0.255
    30 permit 10.1.3.0, wildcard bits 0.0.0.255

! Step 3: Still in ACL 24 configuration mode, the line with sequence number 20 is
deleted.
R1(config-std-nacl)# no 20

! Step 4: Displaying the ACL's contents again, without leaving configuration mode.
! Note that line number 20 is no longer listed.
R1(config-std-nacl)#do show ip access-lists 24
Standard IP access list 24
    10 permit 10.1.1.0, wildcard bits 0.0.0.255
    30 permit 10.1.3.0, wildcard bits 0.0.0.255

! Step 5: Inserting a new first line in the ACL.
R1(config-std-nacl)# 5 deny 10.1.1.1

! Step 6: Displaying the ACL's contents one last time, with the new statement
!(sequence number 5) listed first.
R1(config-std-nacl)# do show ip access-lists 24
Standard IP access list 24
     5 deny 10.1.1.1
    10 permit 10.1.1.0, wildcard bits 0.0.0.255
    30 permit 10.1.3.0, wildcard bits 0.0.0.255
```

Note that although Example 3-6 uses a numbered ACL, named ACLs use the same process to edit (add and remove) entries.

Numbered ACL Configuration Versus Named ACL Configuration

As a brief aside about numbered ACLs, note that IOS actually allows two ways to configure numbered ACLs in the more recent versions of IOS. First, IOS supports the traditional method, using the **access-list** global commands shown earlier in Examples 3-1, 3-2, and 3-3. IOS also supports the numbered ACL configuration with commands just like named ACLs, as shown in Example 3-6.

Oddly, IOS always stores numbered ACLs with the original style of configuration, as global **access-list** commands, no matter which method is used to configure the ACL. Example 3-7 demonstrates these facts, picking up where Example 3-6 ended, with the following additional steps:

Step 7. The engineer lists the configuration (**show running-config**), which lists the old-style configuration commands—even though the ACL was created with the new-style commands.

Step 8. The engineer adds a new statement to the end of the ACL using the old-style **access-list 24 permit 10.1.4.0 0.0.0.255** global configuration command.

Step 9. The **show ip access-lists** command confirms that the old-style **access-list** command from the previous step followed the rule of being added only to the end of the ACL.

Step 10. The engineer displays the configuration to confirm that the parts of ACL 24 configured with both new-style commands and old-style commands are all listed in the same old-style ACL (**show running-config**).

Example 3-7 *Adding to and Displaying a Numbered ACL Configuration*

```
! Step 7: A configuration snippet for ACL 24.
R1# show running-config
! The only lines shown are the lines from ACL 24
access-list 24 deny 10.1.1.1
access-list 24 permit 10.1.1.0 0.0.0.255
access-list 24 permit 10.1.3.0 0.0.0.255

! Step 8: Adding a new access-list 24 global command
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# access-list 24 permit 10.1.4.0 0.0.0.255
R1(config)# ^Z

! Step 9: Displaying the ACL's contents again, with sequence numbers. Note that even
! the new statement has been automatically assigned a sequence number.
R1# show ip access-lists 24
Standard IP access list 24
     5 deny 10.1.1.1
    10 permit 10.1.1.0, wildcard bits 0.0.0.255
    30 permit 10.1.3.0, wildcard bits 0.0.0.255
    40 permit 10.1.4.0, wildcard bits 0.0.0.255

! Step 10: The numbered ACL config remains in old-style configuration commands.
R1# show running-config
! The only lines shown are the lines from ACL 24
access-list 24 deny   10.1.1.1
access-list 24 permit 10.1.1.0 0.0.0.255
access-list 24 permit 10.1.3.0 0.0.0.255
access-list 24 permit 10.1.4.0 0.0.0.255
```

ACL Implementation Considerations

ACLs can be a great tool to enhance the security of a network, but engineers should think about some broader issues before simply configuring an ACL to fix a problem. To help, Cisco makes the following general recommendations in the courses on which the CCNA exam is based:

- Place extended ACLs as close as possible to the source of the packet. This strategy allows ACLs to discard the packets early.
- Place standard ACLs as close as possible to the destination of the packet. This strategy avoids the mistake with standard ACLs (which match the source IPv4 address only) of unintentionally discarding packets that did not need to be discarded.
- Place more specific statements early in the ACL.
- Disable an ACL from its interface (using the **no ip access-group** interface subcommand) before making changes to the ACL.

The first point deals with the concept of where to locate your ACLs. If you intend to filter a packet, filtering closer to the packet's source means that the packet takes up less bandwidth in the network, which seems to be more efficient—and it is. Therefore, Cisco suggests locating extended ACLs as close to the source as possible.

However, the second point seems to contradict the first point, at least for standard ACLs, to locate them close to the destination. Why? Well, because standard ACLs look only at the source IP address, they tend to filter more than you want filtered when placed close to the source. For example, imagine that Fred and Barney are separated by four routers. If you filter Barney's traffic sent to Fred on the first router, Barney can't reach any hosts near the other three routers. So, the Cisco courses make a blanket recommendation to locate standard ACLs closer to the destination to avoid filtering traffic you do not mean to filter.

For the third item in the list, by placing more specific matching parameters early in each list, you are less likely to make mistakes in the ACL. For example, imagine that the ACL first listed a command that permitted traffic going to 10.1.1.0/24, and the second command denied traffic going to host 10.1.1.1. Packets sent to host 10.1.1.1 would match the first command, and never match the more specific second command. Note that later IOS versions prevent this mistake during configuration in some cases.

Finally, Cisco recommends that you disable the ACLs on the interfaces before you change the statements in the list. By doing so, you avoid issues with the ACL during an interim state. First, if you delete an entire ACL and leave the IP ACL enabled on an interface with the **ip access-group** command, IOS does not filter any packets (that was not always the case in far earlier IOS versions)! As soon as you add one ACL command to that enabled ACL, however, IOS starts filtering packets based on that ACL. Those interim ACL configurations could cause problems.

For example, suppose you have ACL 101 enabled on S0/0/0 for output packets. You delete list 101 so that all packets are allowed through. Then you enter a single **access-list 101** command. As soon as you press Enter, the list exists, and the router filters all packets exiting S0/0/0 based on the one-line list. If you want to enter a long ACL, you might temporarily filter packets you don't want to filter! Therefore, the better way is to disable the list from the interface, make the changes to the list, and then reenable it on the interface.

Additional Reading on ACLs

Cisco has long included IP ACLs in the CCNA exam. Preceding the current CCNA 200-301 exam, the CCNA R&S 200-125 exam included IP ACL troubleshooting. If you would like to learn more about ACLs, particularly about troubleshooting ACLs, as well as some unexpected behavior with ACLs and router-generated packets, refer to the section titled "Troubleshooting with IPv4 ACLs," in Appendix D, "Topics from Previous Editions."

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 3-7 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 3-7 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review memory tables		Book, website
Review command tables		Book

Review All the Key Topics

Key Topic

Table 3-8 Key Topics for Chapter 3

Key Topic Element	Description	Page Number
Figure 3-3	Syntax and notes about the three required matching fields in the extended ACL **access-list** command	47
Paragraph	Summary of extended ACL logic that all parameters must match in a single **access-list** statement for a match to occur	48
Figure 3-4	Drawing of the IP header followed by a TCP header	48
Figure 3-5	Syntax and notes about matching TCP and UDP ports with extended ACL **access-list** commands	48
Figure 3-7	Logic and syntax to match TCP source ports	49
List	Guidelines for using extended numbered IP ACLs	51
List	Differences between named and numbered ACLs when named ACLs introduced	55
List	Features enabled by ACL sequence numbers	57
List	ACL implementation recommendations	60

Key Terms You Should Know

extended access list, named access list

Command References

Tables 3-9 and 3-10 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 3-9 Chapter 3 ACL Configuration Command Reference

Command	Description
access-list *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]	Global command for extended numbered **access lists**. Use a number between 100 and 199 or 2000 and 2699, inclusive.
access-list *access-list-number* {**deny** | **permit**} **tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**log**]	A version of the **access-list** command with TCP-specific parameters.
access-list *access-list-number* **remark** *text*	Command that defines a remark to help you remember what the ACL is supposed to do.
ip access-group {*number* | *name* [**in** | **out**]}	Interface subcommand to enable access lists.
access-class *number* | *name* [**in** | **out**]	Line subcommand to enable either standard or extended access lists on vty lines.
ip access-list {**standard** | **extended**} *name*	Global command to configure a named standard or extended ACL and enter ACL configuration mode.
{**deny** | **permit**} *source* [*source wildcard*] [**log**]	ACL mode subcommand to configure the matching details and action for a standard named ACL.
{**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]	ACL mode subcommand to configure the matching details and action for an extended named ACL.
{**deny** | **permit**} **tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**log**]	ACL mode subcommand to configure the matching details and action for a named ACL that matches TCP segments.
remark *text*	ACL mode subcommand to configure a description of a named ACL.

Table 3-10 Chapter 3 EXEC Command Reference

Command	Description
show ip *interface* [*type number*]	Includes a reference to the access lists enabled on the interface
show access-lists [*access-list-number* | *access-list-name*]	Shows details of configured access lists for all protocols
show ip access-lists [*access-list-number* | *access-list-name*]	Shows IP access lists

Answers to Earlier Practice Problems

Table 3-11 lists the answers to the practice problems listed in Table 3-6. Note that for any question that references a client, you might have chosen to match port numbers greater than 49151, matching all dynamic ports. The answers in this table mostly ignore that option, but just to show one sample, the answer to the first problem lists one with a reference to client ports greater than 49151 and one without. The remaining answers simply omit this part of the logic.

Table 3-11 Building One-Line Extended ACLs: Answers

	Criteria
1	access-list 101 permit tcp host 10.1.1.1 10.1.2.0 0.0.0.255 eq www or access-list 101 permit tcp host 10.1.1.1 gt 49151 10.1.2.0 0.0.0.255 eq www
2	access-list 102 permit tcp 172.16.4.0 0.0.0.127 172.16.3.0 0.0.0.127 eq telnet
3	access-list 103 permit icmp 192.168.7.192 0.0.0.63 192.168.7.8 0.0.0.7
4	access-list 104 permit tcp 10.2.2.0 0.0.1.255 eqwww 10.4.4.0 0.0.3.255
5	access-list 105 permit tcp 172.20.1.0 0.0.0.255 eq 23 172.20.44.0 0.0.1.255
6	access-list 106 permit tcp 192.168.99.96 0.0.0.15 192.168.176.0 0.0.0.15 eq www
7	access-list 107 permit icmp 10.55.66.0 0.0.0.127 10.66.55.0 0.0.0.63
8	access-list 108 permit ip any any

Part I Review

Keep track of your part review progress with the checklist in Table P1-1. Details about each task follow the table.

Table P1-1 Part I Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		

Repeat All DIKTA Questions

For this task, use the PTP software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Do Labs

Depending on your chosen lab tool, here are some suggestions for what to do in the lab:

Pearson Network Simulator: If you use the full Pearson CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Config Labs: In your idle moments, review and repeat any of the Config Labs for this book part in the author's blog; navigate to blog.certskills.com/config-labs for instructions on how to navigate to the labs.

Other: If you are using other lab tools, here are a few suggestions: when building ACL labs, you can test with Telnet (port 23), SSH (port 22), ping (ICMP), and traceroute (UDP) traffic as generated from an extra router. So, do not just configure the ACL; make an ACL that can match these types of traffic, denying some and permitting others, and then test.

This page intentionally left blank

With the introduction of the new CCNA certification in early 2020, Cisco expanded the number of security topics in comparison to the old CCNA Routing and Switching certification. Part II includes the majority of the new security topics added to the new CCNA 200-301 certification as well as a few of the classic topics found in previous CCNA R&S exams.

Chapter 4 kicks off Part II with a wide description of security threats, vulnerabilities, and exploits. This introductory chapter sets the stage to help you think more like a security engineer.

Chapters 5, 6, and 8 then focus on a wide range of short security topics. Those topics include Chapter 5's discussion of how to protect router and switch logins and passwords, along with an introduction to the functions and roles of firewalls or intrusion protection systems (IPSs). Chapters 6 and 8 then get into three separate security features built into Cisco switches: port security (Chapter 6), DHCP Snooping (Chapter 8), and Dynamic ARP Inspection (DAI). All three security features require a switch to examine frames as they enter the switch interface. This information enables port security, DHCP Snooping, and DAI to decide whether to allow the message to continue on its way.

Chapter 7 discusses the Dynamic Host Configuration Protocol (DHCP) as an end to itself. While this topic is actually an IP Service and would be a great fit for Part III (IP Services), the topics in Chapter 8 require that you know DHCP, so Chapter 7 sets that stage.

Part II

Security Services

Chapter 4: Security Architectures

Chapter 5: Securing Network Devices

Chapter 6: Implementing Switch Port Security

Chapter 7: Implementing DHCP

Chapter 8: DHCP Snooping and ARP Inspection

Part II Review

CHAPTER 4

Security Architectures

This chapter covers the following exam topics:

5.0 Security Fundamentals

5.1 Define key security concepts (threats, vulnerabilities, exploits, and mitigation techniques)

5.2 Describe security program elements (user awareness, training, and physical access control)

5.4 Describe security password policies elements, such as management, complexity, and password alternatives (multifactor authentication, certificates, and biometrics)

5.8 Differentiate authentication, authorization, and accounting concepts

As you have learned about various networking technologies, your attention has probably been focused on using network devices to build functional networks. After all, networks should let data flow freely so that all connected users have a good experience, right? The unfortunate fact is that not all connected users can be trusted to obey the rules and be good network citizens. In this chapter, you will learn about many aspects of an enterprise network that can be exploited, as well as some ways you can protect them.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 4-1 "Do I Know This Already?" Section-to-Question Mapping

Foundation Topics Section	Questions
Security Terminology	1–2
Common Security Threats	3–7
Controlling and Monitoring User Access	8
Developing a Security Program to Educate Users	9

1. Which one of the following terms means anything that can be considered to be a weakness that can compromise security?
   - **a.** Exploit
   - **b.** Vulnerability
   - **c.** Attack
   - **d.** Threat

2. An actual potential to exploit a vulnerability is known as which one of the following terms?
   a. Vulnerability
   b. Attack
   c. Exploit
   d. Threat
3. In a spoofing attack, which of the following parameters are commonly spoofed? (Choose two answers.)
   a. MAC address
   b. Source IP address
   c. Destination IP address
   d. ARP address
4. Suppose an attacker sends a series of packets toward a destination IP address with the TCP SYN flag set but sends no other packet types. Which of the following attacks is likely taking place?
   a. Spoofing attack
   b. Reflection attack
   c. Reconnaissance attack
   d. Denial-of-service attack
   e. None of the choices are correct.
5. In a reflection attack, the source IP address in the attack packets is spoofed so that it contains which one of the following entities?
   a. The address of the attacker
   b. The address of the reflector
   c. The address of the victim
   d. The address of the router
6. During a successful man-in-the-middle attack, which two of the following actions is an attacker most likely to perform?
   a. Eavesdrop on traffic passing between hosts
   b. Induce a buffer overflow on multiple hosts
   c. Modify data passing between hosts
   d. Use ping sweeps and port scans to discover the network

7. Which one of the following is the goal of a brute-force attack?
   a. Try every possible TCP port until a service answers
   b. Try every possible combination of keyboard characters to guess a user's password
   c Initiate a denial-of-service operation on every possible host in a subnet
   d. Spoof every possible IP address in an organization
8. Which one of the following is an example of a AAA server?
   a. DHCP
   b. DNS
   c. SNMP
   d. ISE
9. Physical access control is important for which one of the following reasons?
   a. It prevents unauthorized people from sitting at a corporate user's desk and using their computer.
   b. It prevents users from getting angry and damaging computer equipment.
   c. It prevents unauthorized access to network closets.
   d. It prevents fires from destroying data centers.

Foundation Topics

Security Terminology

In a perfect world, you might build a network that supports every user in an enterprise, with the assumption that every user is known, every user is approved to access everything on the network, and every user will use the available resources exactly according to some corporate guidelines. The network shown in Figure 4-1 might represent such a scenario. Even this ideal, closed system is not completely secure because a user might decide to misbehave in order to pester a coworker or to view information on the corporate server that should be restricted or confidential.

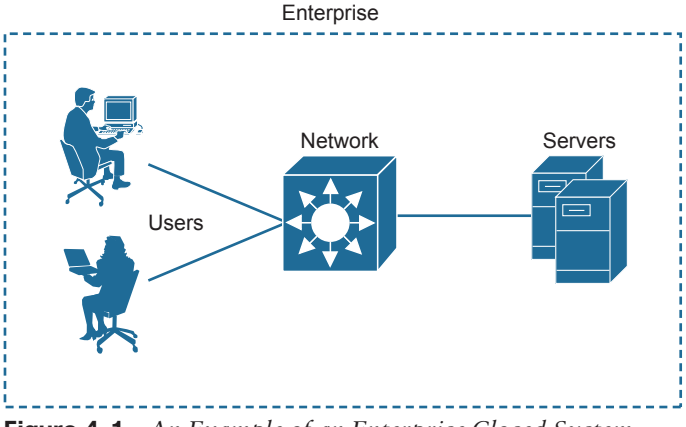


Figure 4-1 *An Example of an Enterprise Closed System*

Now consider that almost no enterprise uses such a limited, closed environment. After all, the enterprise will probably want to somehow connect itself to the public Internet and perhaps to some corporate partners. It will also probably want to allow its workers to be mobile and carry laptops, tablets, and smartphones in and out of the corporate boundaries for convenience. The enterprise might want to provide network access to guests who visit. If the enterprise offers wireless connectivity to its employees (and guests), it might also unknowingly offer its wireless access to people who are within range of the signals. And the list goes on and on. As the network and its connectivity expand, as Figure 4-2 shows, the enterprise will have more difficulty maintaining the safe, closed boundary around itself.

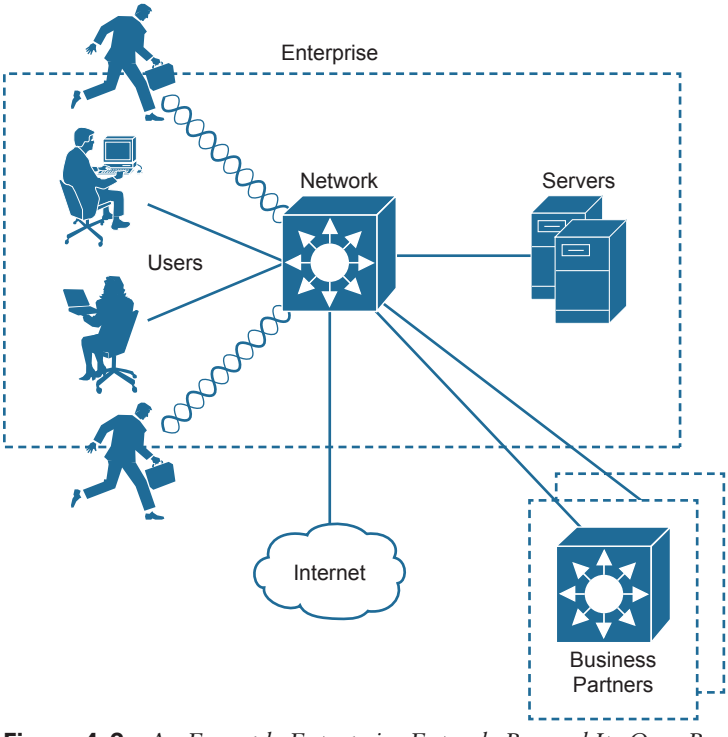


Figure 4-2 *An Example Enterprise Extends Beyond Its Own Boundary*

To begin securing a network, you first need to understand what might go wrong with it. Think of an enterprise network as a simple box-shaped facility, as shown in part A of Figure 4-3. When all of the walls, floor, and ceiling are made of a very strong material and are very thick, the contents inside the box will likely remain safe from harm or theft. The owner, however, might have a hard time getting in and out of the box.

Key Topic

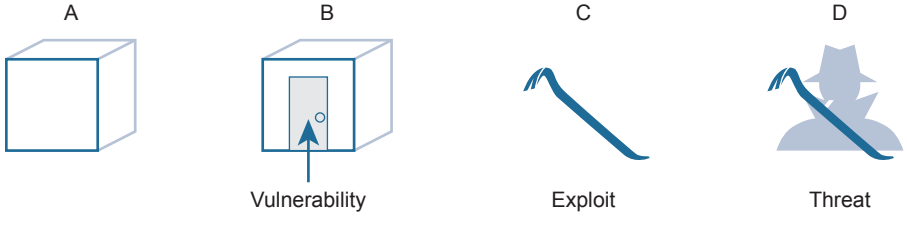


Figure 4-3 *Security Terminology Illustrated*

Suppose a door is introduced for convenience, as shown in part B of Figure 4-3. The owner can now come and go, but so might anyone else. Even if the door is locked, someone might find a way to get the door open and access the treasures inside. Because no door is impenetrable, the door becomes a *vulnerability*. In terms of security, a vulnerability is anything that can be considered to be a weakness that can compromise the security of something else, such as the integrity of data or how a system performs.

Just because a vulnerability exists, nothing is necessarily in jeopardy. In the locked door example, nobody but the trusted owner can open the door unless some sort of tool other than the key is used. Such a tool can be used to exploit a vulnerability. In fact, the tool itself is called an *exploit*, as shown by the pry bar in part C of Figure 4-3. An exploit is not very effective if it is used against anything other than the targeted weakness or vulnerability.

Technically, an exploit such as the pry bar is not very effective at all by itself. Someone must pick it up and use it against the vulnerability. In part D of Figure 4-3, a malicious user possesses the pry bar and intends to use it to open the locked door. Now there is an actual potential to break in, destroy, steal, or otherwise modify something without permission. This is known as a *threat*.

In the IT world of networks, systems, workstations, and applications, there are many, many different vulnerabilities and exploits that can be leveraged by malicious users to become threats to an organization and its data. The remainder of this chapter provides an overview of many of them, along with some techniques you can leverage to counteract or prevent the malicious activity. Such measures are known as *mitigation techniques*. You might be thinking of some ways the Figure 4-3 building owner could mitigate the threats he faces. Perhaps he could add stronger, more secure locks to the door, a more robust door frame to withstand prying forces, or an alarm system to detect an intrusion and alert the authorities.

Common Security Threats

Because modern enterprise networks are usually made up of many parts that all work together, securing them can become a very complex task. As with the simple box analogy, you cannot effectively try to secure it until you have identified many of the vulnerabilities, assessed the many exploits that exist, and realized where the threats might come from. Only then can the appropriate countermeasures and mitigations be put in place.

You should also consider some important attributes of enterprise resources that should be protected and preserved. As you work through the many threats that are discussed in this chapter, think about the vulnerability and exploit that makes the threat possible. Notice how many different parts of the enterprise network exhibit vulnerabilities and how the threats are crafted to take advantage of the weaknesses.

Attacks That Spoof Addresses

When systems behave normally, parameters and services can be trusted and used effectively. For example, when a machine sends an IP packet, everyone expects the source IP address to be the machine's own IP address. The source MAC address in the Ethernet frame

---

Answers to the "Do I Know This Already?" quiz:

1 B **2** D **3** A, B **4** D **5** C **6** A, C **7** B **8** D **9** C

is expected to be the sender's own MAC address. Even services like DHCP and DNS should follow suit; if a machine sends a DHCP or DNS request, it expects any DHCP or DNS reply to come from a legitimate, trusted server.

Spoofing attacks focus on one vulnerability; addresses and services tend to be implicitly trusted. Attacks usually take place by replacing expected values with spoofed or fake values. Address spoofing attacks can be simple and straightforward, where one address value is substituted for another.

For example, an attacker can send packets with a spoofed source IP address instead of its own, as shown in Figure 4-4. When the target receives the packets, it will send return traffic to the spoofed address, rather than the attacker's actual address. If the spoofed address exists, then an unsuspecting host with that address will receive the packet. If the address does not exist, the packet will be forwarded and then dropped further out in the network.

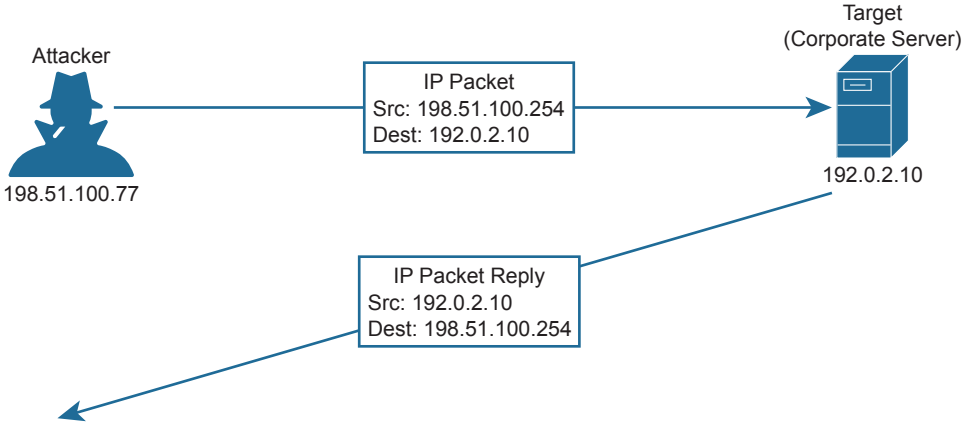


Figure 4-4 *A Sample Spoofing Attack*

An attacker can send spoofed MAC addresses too, to add false information to the forwarding tables used by Layer 2 switches or ARP tables used by other hosts and routers. DHCP requests with spoofed MAC addresses can also be sent to a legitimate DHCP server, filling its address lease table and leaving no free IP addresses for normal use.

Note that Chapter 6, "Implementing Switch Port Security," discusses a tool that can be used to help mitigate MAC address spoofing. In Chapter 8, "DHCP Snooping and ARP Inspection," you can learn more about Dynamic ARP Inspection (DAI) and how to use it to mitigate IP address spoofing using ARP.

Denial-of-Service Attacks

In the normal operation of a business application, clients open connections to corporate servers to exchange information. This might occur in the form of web-based sessions that are open to internal users as well as external users on the public Internet. The process is simple: users open a web browser to the corporate site, which then opens a TCP connection with the corporate web server; then some transaction can take place. If all the users are well behaved and conduct legitimate transactions, the corporate servers are (hopefully) not stressed and many clients can do business normally.

Now suppose a malicious user finds a way to open an abnormal connection to the same corporate server. The TCP connection begins with the malicious user sending a SYN flag to the server, but the source IP address is replaced with a fake address. The server adds the TCP connection to its table of client connections and replies to the fake address with a SYN-ACK. Because the fake address is not involved in the TCP connection, there is no ACK reply to complete the TCP three-way handshake. The incomplete connection stays in the server's table until it eventually times out and is removed. During this time, the attacker can try to open many, many more abnormal connections at such a rate that the server's connection table fills. At that point, the server is no longer able to maintain TCP connections with legitimate users, so their business transactions all halt. Figure 4-5 illustrates this process.

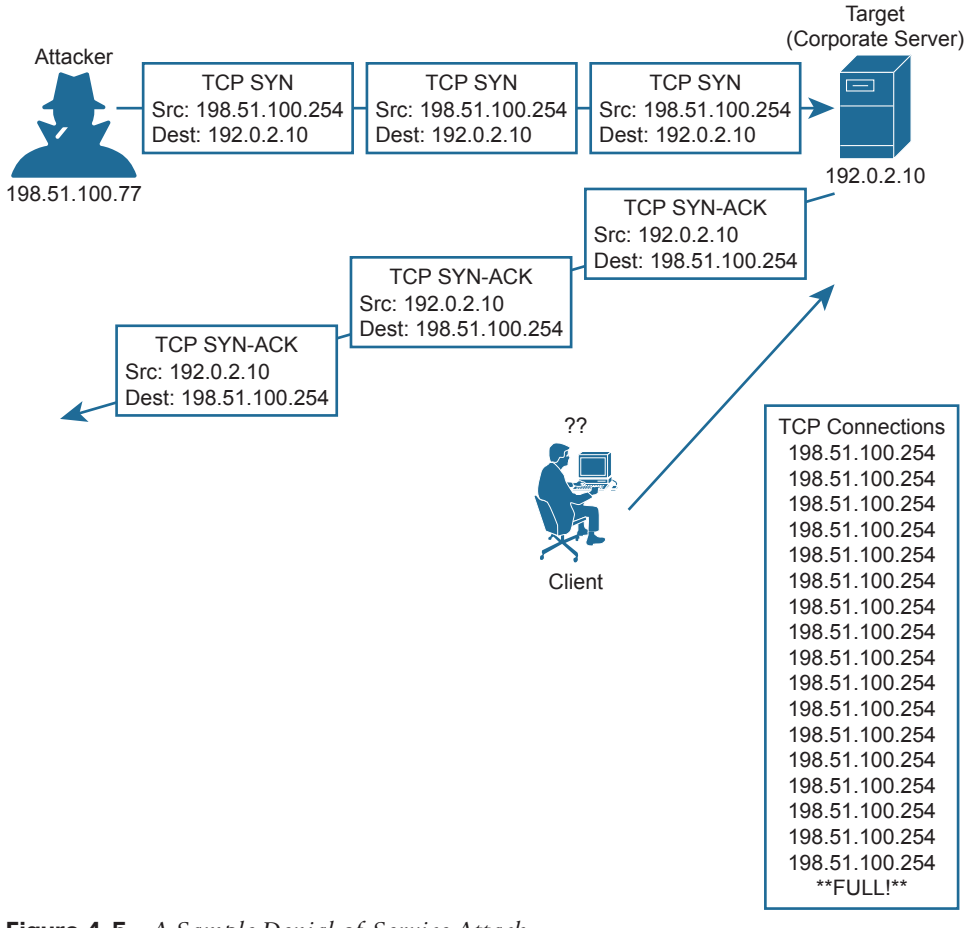


Figure 4-5 *A Sample Denial-of-Service Attack*

When an attacker is able to deplete a system resource, services and systems become unavailable or crash. This is called a *denial-of-service (DoS)* attack because it denies service to legitimate users or operations. DoS attacks can involve something as simple as ICMP echo (ping) packets, a flood of UDP packets, and TCP connections, such as the TCP SYN flood attack previously described. Such attacks can be successful provided a system has a vulnerability with the protocol or type of traffic that is exploited.

Attackers can carry the DoS idea even further by enlisting many other systems to participate. To do this, the attacker sets up a master control computer somewhere on the Internet. Next, many computers must first be infected with malicious code or malware by leveraging vulnerabilities present in those machines. Each machine then silently becomes a "bot," appearing to operate normally, while awaiting commands from the master control. When the time comes for an attack to begin, the master control sends a command to every bot and tells it to initiate a denial-of-service attack against a single target host. This is called a *distributed denial-of-service* (*DDoS*) attack because the attack is distributed across a large number of bots, all flooding or attacking the same target.

Reflection and Amplification Attacks

Recall that in a spoofing attack, the attacker sends packets with a spoofed source address to a target. The goal is to force the target to deal with the spoofed traffic and send return traffic toward a nonexistent source. The attacker does not care where the return traffic goes or that it cannot be delivered successfully.

In a somewhat related attack, the attacker again sends packets with a spoofed source address toward a live host. However, the host is not the intended target; the goal is to get the host to reflect the exchange toward the spoofed address that is the target. This is known as a *reflection attack* as illustrated in Figure 4-6, and the host reflecting the traffic toward the target is called the reflector. The attacker might also send the spoofed packets to multiple reflectors, causing the target to receive multiple copies of the unexpected traffic.

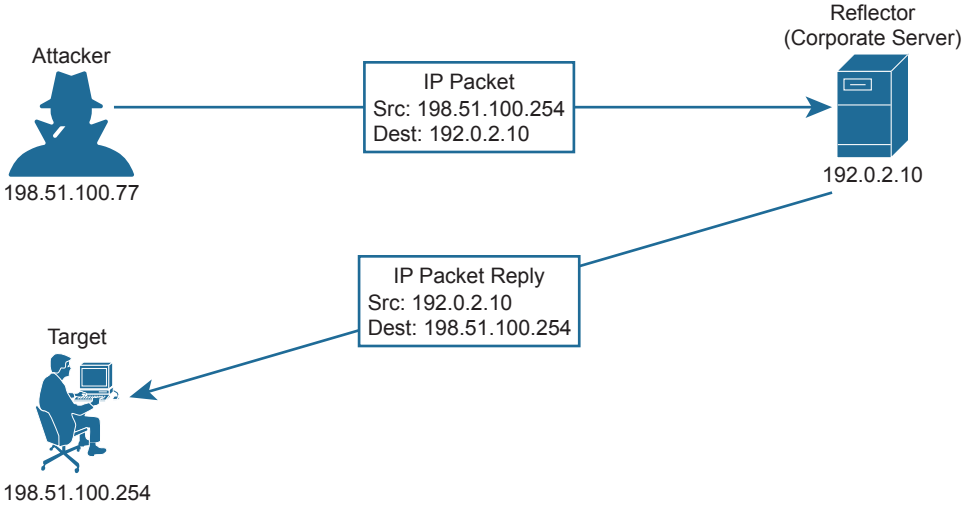


Figure 4-6 *A Sample Reflection Attack*

The impact of a reflection attack might seem limited because a single target host is the victim, and the amount of traffic being reflected to the target is in proportion to the packets sent by the attacker. If an attacker is able to send a small amount of traffic to a reflector and leverage a protocol or service to generate a large volume of traffic toward a target, then an *amplification attack* has occurred. In effect, such an attack amplifies the attacker's efforts to disrupt the target. Another result is that large amounts of network bandwidth can be consumed forwarding the amplified traffic toward the target, especially if many reflectors are involved. Some mechanisms of DNS and NTP have been exploited in the past to set new records for enormous bandwidth consumption during an amplification attack.

Man-in-the-Middle Attacks

Many types of attacks are meant to disrupt or directly compromise targeted systems, often with noticeable results. Sometimes an attacker might want to eavesdrop on data that passes from one machine to another, avoiding detection. A *man-in-the-middle attack* does just that, by allowing the attacker to quietly wedge itself into the communication path as an intermediary between two target systems.

One type of man-in-the-middle attack exploits the ARP table that each host maintains to communicate with other hosts on its local network segment. Normally, if one host needs to send data to another, it looks for the destination host in its ARP table. If an entry is found, the Ethernet frame can be sent directly to the destination MAC address; otherwise, the sender must broadcast an ARP request containing the destination's IP address and wait for the destination to answer with an ARP reply and its own MAC address.

Figure 4-7 illustrates a successful man-in-the-middle attack.

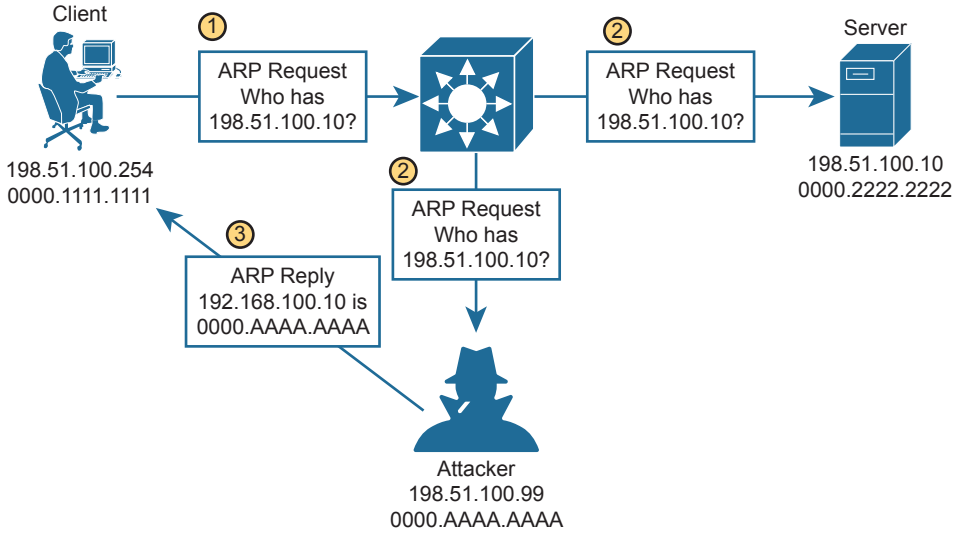


Figure 4-7 *A Man-in-the-Middle Attack Begins*

In step 1, a client broadcasts an ARP request to find out what MAC address is used by the host with IP address 198.51.100.10. In step 2, the ARP request is flooded to all hosts in the broadcast domain. This allows the attacker to overhear the ARP request and prepare to exploit the information learned. The legitimate owner of 198.51.100.10 may indeed respond with its own ARP reply and real MAC address, as expected. However, in step 3, the attacker simply waits a brief time and then sends a spoofed ARP reply containing its own MAC address, rather than that of the actual destination. The goal is for the attacker to send the last ARP reply so that any listening host will update its ARP table with the most recent information.

This process effectively poisons the ARP table entry in any system receiving the spoofed ARP reply. From that point on, a poisoned system will blindly forward traffic to the attacker's MAC address, which now masquerades as the destination. The attacker is able to know the real destination's MAC address because he received an earlier ARP reply from the

destination host. Figure 4-8 depicts the end result. The attacker can repeat this process by poisoning the ARP entries on multiple hosts and then relaying traffic between them without easy detection.

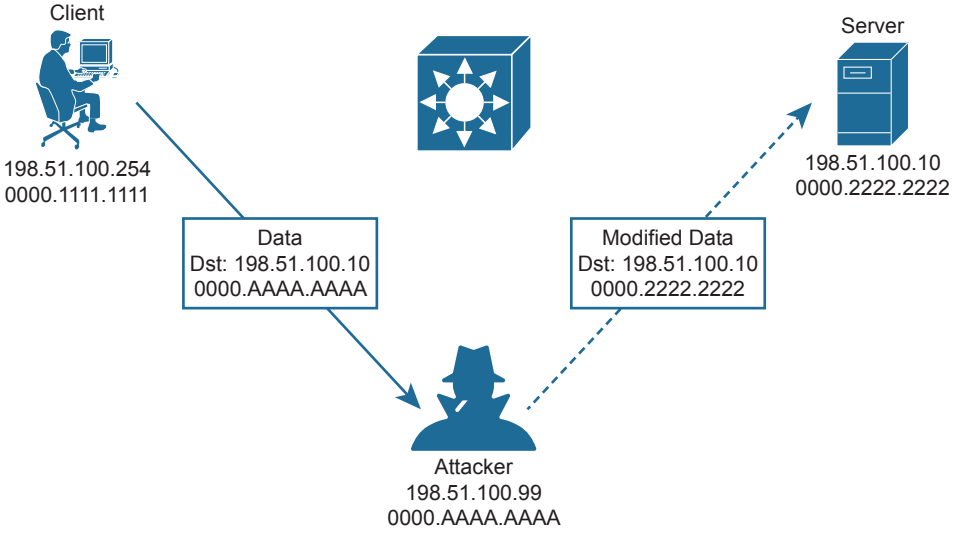


Figure 4-8 *A Man-in-the-Middle Attack Succeeds*

Once an attacker has inserted herself between two hosts, she can passively eavesdrop on and inspect all traffic passing between them. The attacker might also take an active role and modify the data passing through.

Address Spoofing Attack Summary

As you work through the various types of address spoofing attacks, remember that the attacker's goal is to disguise his identity and fool other systems in a malicious way. Use Table 4-2 to review the concepts and characteristics of each attack type.

Table 4-2 Summary of Address Spoofing Attacks

Goal	DoS/DDoS	Reflection	Amplification	Man-in-the-Middle
Exhaust a system service or resource; crash the target system	Yes	No	No	No
Trick an unwitting accomplice host to send traffic to target	No	Yes	Yes	No
Eavesdrop on traffic	No	No	No	Yes
Modify traffic passing through	No	No	No	Yes

Reconnaissance Attacks

When an attacker intends to launch an attack on a target, that attacker might want to identify some vulnerabilities so the attack can be focused and more effective. A *reconnaissance attack* can be used to discover more details about the target and its systems prior to an actual attack.

During a reconnaissance attack, the attacker can use some common tools to uncover public details like who owns a domain and what IP address ranges are used there. For example, the **nslookup** command exists in many operating systems and can perform a DNS lookup to resolve an IP address from a fully qualified domain name. If an attacker knows the domain name of a business, **nslookup** can reveal the owner of the domain and the IP address space registered to it. The **whois** and **dig** commands are complementary tools that can query DNS information to reveal detailed information about domain owners, contact information, mail servers, authoritative name servers, and so on.

Then the attacker can progress to using ping sweeps to send pings to each IP address in the target range. Hosts that answer the ping sweep then become live targets. Port scanning tools can then sweep through a range of UDP and TCP ports to see if a target host answers on any port numbers. Any replies indicate that a corresponding service is running on the target host.

Keep in mind that a reconnaissance attack is not a true attack because nothing is exploited as a result. It is used for gathering information about target systems and services so that vulnerabilities can be discovered and exploited using other types of attacks.

Buffer Overflow Attacks

Operating systems and applications normally read and write data using buffers and temporary memory space. Buffers are also important when one system communicates with another, as IP packets and Ethernet frames come and go. As long as the memory space is maintained properly and data is placed within the correct buffer boundaries, everything should work as expected.

However, some systems and applications have vulnerabilities that can allow buffers to overflow. This means some incoming data might be stored in unexpected memory locations if a buffer is allowed to fill beyond its limit. An attacker can exploit this condition by sending data that is larger than expected. If a vulnerability exists, the target system might store that data, overflowing its buffer into another area of memory, eventually crashing a service or the entire system. The attacker might also be able to specially craft the large message by inserting malicious code in it. If the target system stores that data as a result of a buffer overflow, then it can potentially run the malicious code without realizing.

Malware

Some types of security threats can come in the form of malicious software or *malware*. For example, a *trojan horse* is malicious software that is hidden and packaged inside other software that looks normal and legitimate. If a well-meaning user decides to install it, the trojan horse software is silently installed too. Then the malware can run attacks of its own on the local system or against other systems. Trojan horse malware can spread from one computer to another only through user interaction such as opening email attachments, downloading software from the Internet, and inserting a USB drive into a computer.

In contrast, *viruses* are malware that can propagate between systems more readily. To spread, virus software must inject itself into another application, then rely on users to transport the infected application software to other victims.

One other type of malware is able to propagate to and infect other systems on its own. An attacker develops *worm* software and deposits it on a system. From that point on, the worm replicates itself and spreads to other systems through their vulnerabilities, then replicates and spreads again and again.

To summarize, Table 4-3 lists the key ideas behind each type of malware described in this section.

Key Topic

Table 4-3 Summary of Malware Types

Characteristic	Trojan Horse	Virus	Worm
Packaged inside other software	Yes	No	No
Self-injected into other software	No	Yes	No
Propagates automatically	No	No	Yes

Human Vulnerabilities

Many types of attack must take advantage of a vulnerability in an operating system, service, or other types of application software. In other words, an attacker or the malware involved must find a weakness in the target computer system. There are still many other attacks that can succeed by exploiting weaknesses in the humans that use computer systems.

One rather straightforward attack is called *social engineering*, where human trust and social behaviors can become security vulnerabilities. For example, an attacker might pose as an IT staff member and attempt to contact actual end users through phone calls, emails, and social media. The end goal might be to convince the users to reveal their credentials or set their passwords to a "temporary" value due to some fictitious IT maintenance that will take place, allowing the attacker to gain easy access to secure systems. Attackers might also be physically present and secretly observe users as they enter their credentials.

Phishing is a technique that attackers use to lure victims into visiting malicious websites. The idea is to either disguise the invitation as something legitimate, frighten victims into following a link, or otherwise deceive users into browsing content that convinces them to enter their confidential information.

Phishing comes in many forms. *Spear phishing* targets a group of similar users who might work for the same company, shop at the same stores, and so on, who all receive the same convincing email with a link to a malicious site. *Whaling* is similar but targets high-profile individuals in corporations, governments, and organizations. Phishing can also occur over traditional communications, such as voice calls (*vishing*) and SMS text messages (*smishing*).

Pharming also attempts to send victims to a malicious website, but it takes a more drastic approach. Rather than enticing victims to follow a disguised link, pharming involves compromising the services that direct users toward a well-known or trusted website. For instance, an attacker can compromise a DNS service or edit local hosts files to change the entry for a legitimate site. When a victim tries to visit the site using its actual link, the altered name resolution returns the address of a malicious site instead.

In a *watering hole* attack, an attacker determines which users frequently visit a site; then that site is compromised and malware is deposited there. The malware infects only the target users who visit the site, while leaving other users unscathed.

You can refer to Table 4-4 to review the key ideas behind each type of human vulnerability that is commonly exploited.

Key Topic

Table 4-4 Summary of Human Security Vulnerabilities

Attack Type	Goal
Social engineering	Exploits human trust and social behavior
Phishing	Disguises a malicious invitation as something legitimate
Spear phishing	Targets group of similar users
Whaling	Targets high-profile individuals
Vishing	Uses voice calls
Smishing	Uses SMS text messages
Pharming	Uses legitimate services to send users to a compromised site
Watering hole	Targets specific victims who visit a compromised site

Password Vulnerabilities

Key Topic

Most systems in an enterprise network use some form of authentication to grant or deny user access. When users access a system, a username and password are usually involved. It might be fairly easy to guess someone's username based on that person's real name. If the user's password is set to some default value or to a word or text string that is easy to guess, an attacker might easily gain access to the system too.

Think like an attacker for a moment and see if you can make some guesses about passwords you might try if you wanted to log in to a random system. Perhaps you thought of passwords like *password*, *password123*, *123456*, and so on. Perhaps you could try username *admin* and password *admin*.

An attacker can launch an online attack by actually entering each password guess as the system prompts for user credentials. In contrast, an offline attack occurs when the attacker is able to retrieve the encrypted or hashed passwords ahead of time, then goes offline to an external computer and uses software there to repeatedly attempt to recover the actual password.

Attackers can also use software to perform dictionary attacks to discover a user's password. The software will automatically attempt to log in with passwords taken from a dictionary or word list. It might have to go through thousands or millions of attempts before discovering the real password. In addition, the software can perform a brute-force attack by trying every possible combination of letter, number, and symbol strings. Brute-force attacks require very powerful computing resources and a large amount of time.

To mitigate password attacks, an enterprise should implement password policies for all users. Such a policy might include guidelines that require a long password string made up of a combination of upper- and lowercase characters along with numbers and some special characters. The goal is to require all passwords to be complex strings that are difficult to guess or reveal by a password attack. As well, password management should require all passwords to be changed periodically so that even lengthy brute-force attacks would not be able to recover a password before it is changed again.

Password Alternatives

A simple password string is the single factor that a user must enter to be authenticated. Because a password should be remembered and not written down anywhere, you might

think of your password as "something you know." Hopefully nobody else knows it too; otherwise, they could use it to impersonate you when authenticating.

An enterprise might also consider using alternative credentials that bring more complexity and more security. Multifactor credentials require users to provide values or factors that come from different sources, reducing the chance that an attacker might possess all of the factors. An old saying describes two-factor credentials as "something you have" (a dynamic changing cryptographic key or a text message containing a time-limited code) and "something you know" (a password).

A digital certificate can serve as one alternative factor because it serves as a trusted form of identification, adheres to a standardized format, and contains encrypted information. If an enterprise supports certificate use, then a user must request and be granted a unique certificate to use for specific purposes. For example, certificates used for authenticating users must be approved for authentication. In order to be trusted, certificates must be granted and digitally signed by a trusted certificate authority (CA). As long as the services used by the enterprise know and trust the CA, then individual certificates signed by that CA can be trusted as well.

Digital certificates are also time sensitive, as each is approved for a specific time range. Once a certificate expires, any attempts to authenticate with it will be rejected. The user who possesses the certificate can request a new one prior to the expiration date or at any time afterward. Certificates can also be revoked, if the business decides to revoke privileges from a user, if the user separates from the business, and so on. Even if the user still possesses a revoked certificate, he will be refused access when he tries to authenticate with it.

Because digital certificates exist as files on a computer or device, you might think they can be freely copied and used to identify people other than the original owners. Each digital certificate must also carry proof of possession to show that it was truly granted to the user who presents it during authentication. This proof is built into the encrypted certificate content, as a result of combining public keys that the user's machine and the authentication server can publicly share, along with private keys that each party keeps private and secret. As long as the authentication server can verify that the certificate was created using the correct public and private keys, then the certificate must be possessed by the expected owner. If not, then authentication will be rejected to keep an imposter out.

Biometric credentials carry the scheme even further by providing a factor that represents "something you are." The idea is to use some physical attribute from a user's body to uniquely identify that person. Physical attributes are usually unique to each individual's body structure and cannot be easily stolen or duplicated. For example, a user's fingerprint can be scanned and used as an authentication factor. Other examples include face recognition, palm prints, voice recognition, iris recognition, and retinal scans. As you might expect, some methods can be trusted more than others. Sometimes facial recognition systems can be fooled when presented with photographs or masks of trusted individuals. Injuries and the aging process can also alter biometric patterns such as fingerprints, facial shapes, and iris patterns. To help mitigate potential weaknesses, multiple biometric credentials can be collected and used to authenticate users as well.

To summarize, Table 4-5 lists the key ideas used in each alternative to password authentication.

Table 4-5 Summary of Password Authentication and Alternatives

Characteristic	Password Only	Two-Factor	Digital Certificates	Biometric
Something you know	Yes	Yes		
Something you have		Yes	Yes	
Something you are				Yes

Controlling and Monitoring User Access

You can manage user activity to and through systems with authentication, authorization, and accounting (AAA, also pronounced "triple-A") mechanisms. AAA uses standardized methods to challenge users for their credentials before access is allowed or authorized. Accounting protocols also can record user activity on enterprise systems. AAA is commonly used to control and monitor access to network devices like routers, switches, firewalls, and so on.

Key Topic

In a nutshell, you can think of AAA in the following manner:

- **Authentication:** Who is the user?
- **Authorization:** What is the user allowed to do?
- **Accounting:** What did the user do?

As an example, a network administrator can have several methods to manage users who might try to log in to a switch to perform some operation. At the most basic level, you could authenticate users with simple passwords that are configured on the switch console and VTY lines. Authorization could be equally simple: when users successfully log in, they are authorized for EXEC level privileges. By entering the correct enable secret password, users could be authorized for a higher privilege level.

Under the simple scenario, if a user knows the correct password, he can connect to the switch. But who is that user? You might never know who actually logged in and changed the configuration or rebooted the switch! Instead, you could configure individual usernames and passwords on the switch. That would solve the user anonymity problem, but your network might consist of many administrative users and many switches, requiring quite a bit of username configuration and maintenance.

A more scalable solution is to leverage AAA functions that are centralized, standardized, resilient, and flexible. For example, a centralized authentication server can contain a database of all possible users and their passwords, as well as policies to authorize user activities. As users come and go, their accounts can be easily updated in one place. All switches and routers would query the AAA server to get up-to-date information about a user. For greater security, AAA servers can also support multifactor user credentials and more. Cisco implements AAA services in its Identity Services Engine (ISE) platform.

AAA servers usually support the following two protocols to communicate with enterprise resources:

- **TACACS+:** A Cisco proprietary protocol that separates each of the AAA functions. Communication is secure and encrypted over TCP port 49.
- **RADIUS:** A standards-based protocol that combines authentication and authorization into a single resource. Communication uses UDP ports 1812 and 1813 (accounting) but is not completely encrypted.

Both TACACS+ and RADIUS are arranged as a client/server model, where an authenticating device acts as a client talking to a AAA server. Figure 4-9 shows a simplified view of the process, where a user is attempting to connect to a switch for management purposes. In the AAA client role, the switch is often called Network Access Device (NAD) or Network Access Server (NAS). When a user tries to connect to the switch, the switch challenges the user for credentials, then passes the credentials along to the AAA server. In simple terms, if the user passes authentication, the AAA server returns an "accept" message to the switch. If the AAA server requires additional credentials, as in multifactor authentication, it returns a "challenge" message to the switch. Otherwise, a "reject" message is returned, denying access to the user.

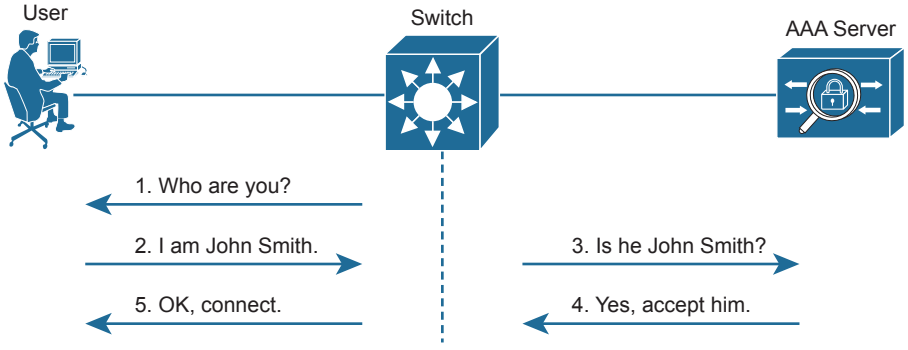


Figure 4-9 *A Simplified View of AAA*

Developing a Security Program to Educate Users

One effective approach an enterprise can take to improve information security is to educate its user community through a corporate security program. Most users may not have an IT background, so they might not recognize vulnerabilities or realize the consequences of their own actions. For example, if corporate users receive an email message that contains a message concerning a legal warrant for their arrest or a threat to expose some supposed illegal behavior, they might be tempted to follow a link to a malicious site. Such an action might infect a user's computer and then open a back door or introduce malware or a worm that could then impact the business operations.

Key Topic

An effective security program should have the following basic elements:

- **User awareness:** All users should be made aware of the need for data confidentiality to protect corporate information, as well as their own credentials and personal information. They should also be made aware of potential threats, schemes to mislead, and proper procedures to report security incidents. Users should also be instructed to follow strict guidelines regarding data loss. For example, users should not include sensitive information in emails or attachments, should not keep or transmit that information from a smartphone, or store it on cloud services or removable storage drives.
- **User training:** All users should be required to participate in periodic formal training so that they become familiar with all corporate security policies. (This also implies that the enterprise should develop and publish formal security policies for its employees, users, and business partners to follow.)

- **Physical access control:** Infrastructure locations, such as network closets and data centers, should remain securely locked. Badge access to sensitive locations is a scalable solution, offering an audit trail of identities and timestamps when access is granted. Administrators can control access on a granular basis and quickly remove access when an employee is dismissed.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 4-6 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 4-6 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website

Review All the Key Topics

Key Topic

Table 4-7 Key Topics for Chapter 4

Key Topic Element	Description	Page Number
Figure 4-3	Security terminology	71
Section	Common Security Threats	72
Table 4-3	Types of malware	79
Table 4-4	Human security vulnerabilities	80
Paragraph	Password vulnerabilities	80
List	AAA functions	82
List	User education	83

Key Terms You Should Know

AAA, amplification attack, brute-force attack, buffer overflow attack, denial-of-service (DoS) attack, dictionary attack, distributed denial-of-service (DDoS) attack, exploit, malware, man-in-the-middle attack, mitigation technique, multifactor authentication, password guessing, pharming, phishing, reconnaissance attack, reflection attack, social engineering, spear phishing, spoofing attack, threat, trojan horse, virus, vulnerability, watering hole attack, whaling, worm

This page intentionally left blank

CHAPTER 5

Securing Network Devices

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the Role of Network Components

1.1.c Next-generation Firewalls and IPS

4.0 IP Services

4.8 Configure network devices for remote access using SSH

5.0 Security Fundamentals

5.3 Configure device access control using local passwords

All devices in the network—endpoints, servers, and infrastructure devices like routers and switches—include some methods for the devices to legitimately communicate using the network. To protect those devices, the security plan will include a wide variety of tools and mitigation techniques, with the chapters in Part II of this book discussing a large variety of those tools and techniques.

This chapter focuses on two particular security needs in an enterprise network. First, access to the CLI of the network devices needs to be protected. The network engineering team needs to be able to access the devices remotely, so the devices need to allow remote SSH (and possibly Telnet) access. The first half of this chapter discusses how to configure passwords to keep them safe and how to filter login attempts at the devices themselves.

The second half of the chapter turns to two different security functions most often implemented with purpose-built appliances: firewalls and IPSs. These devices together monitor traffic in transit to determine if the traffic is legitimate or if it might be part of some exploit. If considered to be part of an exploit, or if contrary to the rules defined by the devices, they can discard the messages, stopping any attack before it gets started.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 5-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Securing IOS Passwords	1–4
Firewalls and Intrusion Prevention Systems	5, 6

1. Imagine that you have configured the **enable secret** command, followed by the **enable password** command, from the console. You log out of the switch and log back in at the console. Which command defines the password that you had to enter to access privileged mode?
   a. **enable password**
   b. **enable secret**
   c. Neither
   d. The **password** command, if it's configured

2. Some IOS commands store passwords as clear text, but you can then encrypt the passwords with the **service password-encryption** global command. By comparison, other commands store a computed hash of the password instead of storing the password. Comparing the two options, which one answer is the *most accurate* about why one method is better than the other?
   a. Using hashes is preferred because encrypted IOS passwords can be easily decrypted.
   b. Using hashes is preferred because of the large CPU effort required for encryption.
   c. Using encryption is preferred because it provides stronger password protection.
   d. Using encryption is preferred because of the large CPU effort required for hashes.

3. A network engineer issues a **show running-config** command and sees only one line of output that mentions the **enable secret** command, as follows:

   ```
   enable secret 5 $1$ZGMA$e8cmvkz4UjiJhVp7.maLE1
   ```

   Which of the following is true about users of this router?
   a. A user must type **1ZGMA$e8cmvkz4UjiJhVp7.maLE1** to reach enable mode.
   b. The router will hash the clear-text password that the user types to compare to the hashed password.
   c. A **no service password-encryption** configuration command would decrypt this password.
   d. The router will decrypt the password in the configuration to compare to the clear-text password typed by the user.

4. A single-line ACL has been added to a router configuration using the command **ip access-list 1 permit 172.16.4.0 0.0.1.255**. The configuration also includes the **access-class 1 in** command in VTY configuration mode. Which answer accurately describes how the router uses ACL 1?
   a. Hosts in subnet 172.16.4.0/23 alone can telnet into the router.
   b. CLI users cannot telnet from the router to hosts in subnet 172.16.4.0/23 alone.
   c. Hosts in subnet 172.16.4.0/23 alone can log in but cannot reach enable mode of the router.
   d. The router will only forward packets with source addresses in subnet 172.16.4.0/23.

5. A next-generation firewall sits at the edge of a company's connection to the Internet. It has been configured to prevent Telnet clients residing in the Internet from accessing Telnet servers inside the company. Which of the following might a next-generation firewall use that a traditional firewall would not?
   a. Match message destination well-known port 23
   b. Match message application data
   c. Match message IP protocol 23
   d. Match message source TCP ports greater than 49152
6. Which actions show a behavior typically supported by a Cisco next-generation IPS (NGIPS) beyond the capabilities of a traditional IPS? (Choose two answers)
   a. Gather and use host-based information for context
   b. Comparisons between messages and a database of exploit signatures
   c. Logging events for later review by the security team
   d. Filter URIs using reputation scores

Foundation Topics

Securing IOS Passwords

The ultimate way to protect passwords in Cisco IOS devices is to not store passwords in IOS devices. That is, for any functions that can use an external authentication, authorization, and accounting (AAA) server, use it. However, it is common to store some passwords in a router or switch configuration, and this first section of the chapter discusses some of the ways to protect those passwords.

As a brief review, Figure 5-1 summarizes some typical login security configuration on a router or switch. On the lower left, you see Telnet support configured, with the use of a password only (no username required). On the right, the configuration adds support for login with both username and password, supporting both Telnet and SSH users. The upper left shows the one command required to define an enable password in a secure manner.

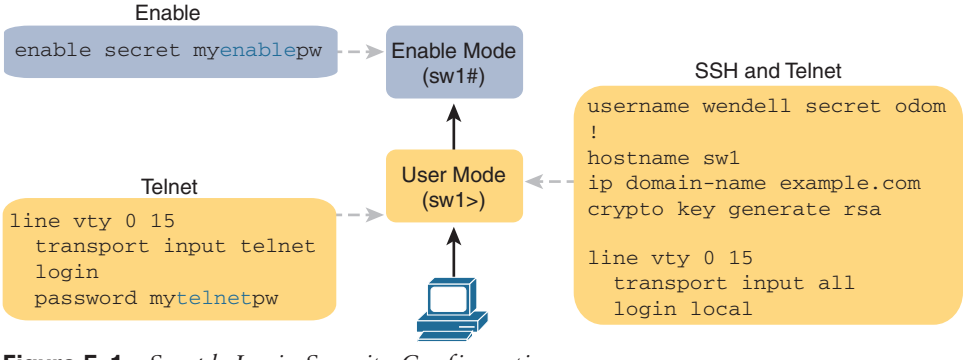


Figure 5-1 *Sample Login Security Configuration*

NOTE The configuration on the far right of the figure supports both SSH and Telnet, but consider allowing SSH only by instead using the **transport input ssh** command. The Telnet protocol sends all data unencrypted, so any attacker who copies the message with a Telnet login will have a copy of the password.

The rest of this first section discusses how to make these passwords secure. In particular, this section looks at ways to avoid keeping clear-text passwords in the configuration and storing the passwords in ways that make it difficult for attackers to learn the password.

Encrypting Older IOS Passwords with service password-encryption

Some older-style IOS passwords create a security exposure because the passwords exist in the configuration file as clear text. These clear-text passwords might be seen in printed versions of the configuration files, in a backup copy of the configuration file stored on a server, or as displayed on a network engineer's display.

Cisco attempted to solve this clear-text problem by adding a command to encrypt those passwords: the **service password-encryption** global configuration command. This command encrypts passwords that are normally held as clear text, specifically the passwords for these commands:

Key Topic

password *password* (console or vty mode)

username *name* **password** *password* (global)

enable password *password* (global)

To see how it works, Example 5-1 shows how the **service password-encryption** command encrypts the clear-text console password. The example uses the **show running-config | section line con 0** command both before and after the encryption; this command lists only the section of the configuration about the console.

Example 5-1 *Encryption and the* **service password-encryption** *Command*

```
Switch3# show running-config | section line con 0
line con 0
 password cisco
 login

Switch3# configure terminal
Enter configuration commands, one per line.   End with CNTL/Z.
Switch3(config)# service password-encryption
Switch3(config)# ^Z

Switch3# show running-config | section line con 0
line con 0
 password 7 070C285F4D06
 login
```

A close examination of the before and after **show running-config** command output reveals both the obvious effect and a new concept. The encryption process now hides the original

clear-text password. Also, IOS needs a way to signal that the value in the **password** command lists an encrypted password rather than the clear text. IOS adds the encryption or encoding type of "7" to the command, which specifically refers to passwords encrypted with the **service password-encryption** command. (IOS considers the clear-text passwords to be type 0; some commands list the 0, and some do not.)

While the **service password-encryption** global command encrypts passwords, the **no service password-encryption** global command does not immediately decrypt the passwords back to their clear-text state. Instead, the process works as shown in Figure 5-2. Basically, after you enter the **no service password-encryption** command, the passwords remain encrypted until you change a password.

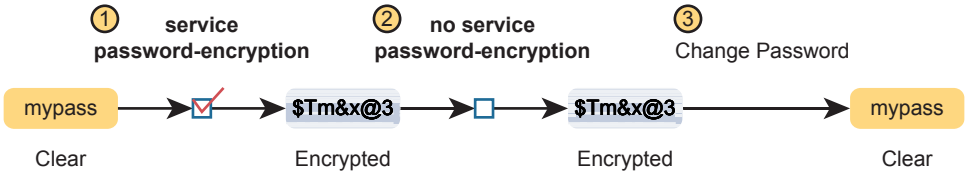


Figure 5-2 *Encryption Is Immediate; Decryption Awaits Next Password Change*

Unfortunately, the **service password-encryption** command does not protect the passwords very well. Armed with the encrypted value, you can search the Internet and find sites with tools to decrypt these passwords. In fact, you can take the encrypted password from this example, plug it into one of these sites, and it decrypts to "cisco." So, the **service password-encryption** command will slow down the curious, but it will not stop a knowledgeable attacker.

Encoding the Enable Passwords with Hashes

In the earliest days of IOS, Cisco used the **enable password** *password* global command to define the password that users had to use to reach enable mode (after using the **enable** EXEC command). However, as just noted, the **enable password** *password* command stored the password as clear text, and the **service password-encryption** command encrypted the password in a way that was easily decrypted.

Cisco solved the problem of only weak ways to store the password of the **enable password** *password* global command by making a more secure replacement: the **enable secret** *password* global command. However, both these commands exist in IOS even today. The next few pages look at these two commands from a couple of angles, including interactions between these two commands, why the **enable secret** command is more secure, along with a note about some advancements in how IOS secures the **enable secret** password.

Interactions Between Enable Password and Enable Secret

First, for real life: use the **enable secret** *password* global command, and ignore the **enable password** *password* global command. That has been true for around 20 years.

However, to be complete, Cisco has never removed the much weaker **enable password** command from IOS. So, on a single switch (or router), you can configure one or the other,

---

Answers to the "Do I Know This Already?" quiz:

1 B **2** A **3** B **4** A **5** B **6** A, D

both, or neither. What, then, does the switch expect us to type as the password to reach enable mode? It boils down to these rules:

Key Topic

Both commands configured: Users must use the password in the **enable secret** *password* command (and ignore the **enable password** *password* command).

Only one command configured: Use the password in that one command.

Neither command configured (default): Console users move directly to enable mode without a password prompt; Telnet and SSH users are rejected with no option to supply an enable password.

Making the Enable Secret Truly Secret with a Hash

The Cisco **enable secret** command protects the password value by never even storing the clear-text password in the configuration. However, that one sentence may cause you a bit of confusion: If the router or switch does not remember the clear-text password, how can the switch know that the user typed the right password after using the **enable** command? This section works through a few basics to show you how and appreciate why the password's value is secret.

First, by default, IOS uses a hash function called Message Digest 5 (MD5) to store an alternative value in the configuration, rather than the clear-text password. Think of MD5 as a rather complex mathematical formula. In addition, this formula is chosen so that even if you know the exact result of the formula—that is, the result after feeding the clear-text password through the formula as input—it is computationally difficult to compute the original clear-text password. Figure 5-3 shows the main ideas:

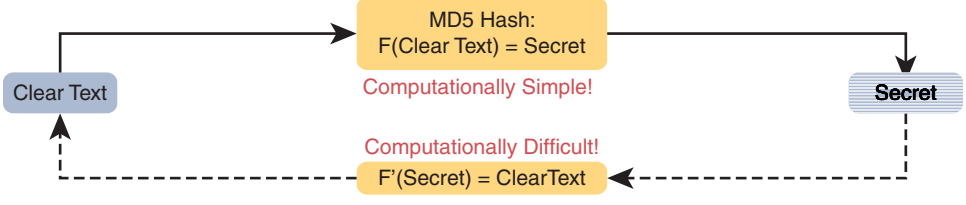


Figure 5-3 *One-Way Nature of MD5 Hash to Create Secret*

NOTE "Computationally difficult" is almost a code phrase, meaning that the designers of the function hope that no one is willing to take the time to compute the original clear text.

So, if the original clear-text password cannot be re-created, how can a switch or router use it to compare to the clear-text password typed by the user? The answer depends on another fact about these security hashes like MD5: each clear-text input results in a unique result from the math formula.

The **enable secret fred** command generates an MD5 hash. If a user types **fred** when trying to enter enable mode, IOS will run MD5 against that value and get the same MD5 hash as is listed in the **enable secret** command, so IOS allows the user to access enable mode. If the user typed any other value besides **fred**, IOS would compute a different MD5 hash than the value stored with the **enable secret** command, and IOS would reject that user's attempt to reach enable mode.

Knowing that fact, the switch can make a comparison when a user types a password after using the **enable** EXEC command as follows:

Key Topic

Step 1. IOS computes the MD5 hash of the password in the **enable secret** command and stores the hash of the password in the configuration.

Step 2. When the user types the **enable** command to reach enable mode, a password that needs to be checked against that configuration command, IOS hashes the clear-text password as typed by the user.

Step 3. IOS compares the two hashed values: if they are the same, the user-typed password must be the same as the configured password.

As a result, IOS can store the hash of the password but never store the clear-text password; however, it can still determine whether the user typed the same password.

Switches and routers already use the logic described here, but you can see the evidence by looking at the switch configuration. Example 5-2 shows the creation of the **enable secret** command, with a few related details. This example shows the stored (hashed) value as revealed in the **show running-configuration** command output. That output also shows that IOS changed the **enable secret fred** command to list the encryption type 5 (which means the listed password is actually an MD5 hash of the clear-text password). The gobbledygook long text string is the hash, preventing others from reading the password.

Example 5-2 *Cisco IOS Encoding Password "cisco" as Type 5 (MD5)*

```
Switch3(config)# enable secret fred
Switch3(config)# ^Z
Switch3# show running-config | include enable secret

enable secret 5 $1$ZGMA$e8cmvkz4UjiJhVp7.maLE1

Switch3# configure terminal
Enter configuration commands, one per line.   End with CNTL/Z.
Switch3(config)# no enable secret
Switch3(config)# ^Z
```

The end of the example also shows an important side point about deleting the **enable secret** password: after you are in enable mode, you can delete the enable secret password using the **no enable secret** command, without even having to enter the password value. You can also overwrite the old password by just repeating the **enable secret** command. But you cannot view the original clear-text password.

NOTE Example 5-2 shows another shortcut illustrating how to work through long **show** command output, this time using the pipe to the **include** command. The **| include enable secret** part of the command processes the output from **show running-config** to include only the lines with the case-sensitive text "enable secret."

Improved Hashes for Cisco's Enable Secret

The use of any hash function to encode passwords relies on several key features of the particular hash function. In particular, every possible input value must result in a single hashed

value, so that when users type a password, only one password value matches each hashed value. Also, the hash algorithm must result in computationally difficult math (in other words, a pain in the neck) to compute the clear-text password based on the hashed value to discourage attackers.

The MD5 hash algorithm has been around 30 years. Over those years, computers have gotten much faster, and researchers have found creative ways to attack the MD5 algorithm, making MD5 less challenging to crack. That is, someone who saw your running configuration would have an easier time re-creating your clear-text secret passwords than in the early years of MD5.

These facts are not meant to say that MD5 is bad, but like many cryptographic functions before MD5, progress has been made, and new functions were needed. To provide more recent options that would create a much greater challenge to attackers, Cisco added two additional hashes in the 2010s, as noted in Figure 5-4.

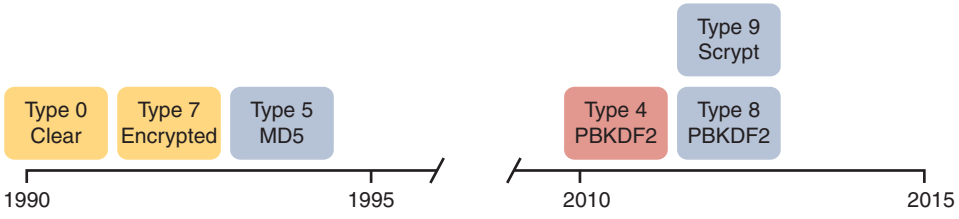


Figure 5-4 *Timeline of Encryptions/Hashes of Cisco IOS Passwords*

IOS now supports two alternative algorithm types in the more recent router and switch IOS images. Both use an SHA-256 hash instead of MD5, but with two newer options, each of which has some differences in the particulars of how each algorithm uses SHA-256. Table 5-2 shows the configuration of all three algorithm types on the **enable secret** command.

Table 5-2 Commands and Encoding Types for the **enable secret** Command

Command	Type	Algorithm
enable [algorithm-type md5] secret *password*	5	MD5
enable algorithm-type sha256 secret *password*	8	SHA-256
enable algorithm-type scrypt secret *password*	9	SHA-256

Example 5-3 shows the **enable secret** command being changed from MD5 to the scrypt algorithm. Of note, the example shows that only one **enable secret** command should exist between those three commands in Table 5-2. Basically, if you configure another **enable secret** command with a different algorithm type, that command replaces any existing **enable secret** command.

Example 5-3 *Cisco IOS Encoding Password "mypass1" as Type 9 (SHA-256)*

```
R1# show running-config | include enable
enable secret 5 $1$ZSYj$725dBZmLUJ0nx8gFPTtTv0
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# enable algorithm-type scrypt secret mypass1
R1(config)# ^Z
```

```
R1#
R1# show running-config | include enable
enable secret 9 $9$II/EeKiRW91uxE$fwYuOE5EHoii16AWv2wSywkLJ/KNeGj8uK/24B0TVU6
R1#
```

Following the process shown in the example, the first command confirms that the current **enable secret** command uses encoding type 5, meaning it uses MD5. Second, the user configures the password using algorithm type scrypt. The last command confirms that only one **enable secret** command exists in the configuration, now with encoding type 9.

Encoding the Passwords for Local Usernames

Cisco added the **enable secret** command back in the 1990s to overcome the problems with the **enable password** command. The **username password** and **username secret** commands have a similar history. Originally, IOS supported the **username** *user* **password** *password* command—a command that had those same issues of being a clear-text password or a poorly encrypted value (with the **service password-encryption** feature). Many years later, Cisco added the **username** *user* **secret** *password* global command, which encoded the password as an MD5 hash, with Cisco adding support for the newer SHA-256 hashes later.

Today, the **username secret** command is preferred over the **username password** command; however, IOS does not use the same logic for the **username** command as it does for allowing both the **enable secret** plus **enable password** commands to exist in the same configuration. IOS allows

- Only one **username** command for a given username—either a **username** *name* **password** *password* command or a **username** *name* **secret** *password* command
- A mix of commands (**username password** and **username secret**) in the same router or switch (for different usernames)

You should use the **username secret** command instead of the **username password** command when possible. However, note that some IOS features require that the router knows a clear-text password via the **username** command (for instance, when performing some common authentication methods for serial links called PAP and CHAP). In those cases, you still need to use the **username password** command.

As mentioned, the more recent IOS versions on both switches and routers use the additional encoding options beyond MD5, just as supported with the **enable secret** command. Table 5-3 shows the syntax of those three options in the **username** command, with the MD5 option shown as an option because it is the default used with the **username secret** command.

Table 5-3 Commands and Encoding Types for the **username secret** Command

Command	Type	Algorithm
username *name* [**algorithm-type md5**] **secret** *password*	5	MD5
username *name* **algorithm-type sha256 secret** *password*	8	SHA-256
username *name* **algorithm-type scrypt secret** *password*	9	SHA-256

Controlling Password Attacks with ACLs

Attackers can repeatedly try to log in to your network devices to gain access, but IOS has a feature that uses ACLs to prevent the attacker from even seeing a password prompt. When an external user connects to a router or switch using Telnet or SSH, IOS uses a vty line to represent that user connection. IOS can apply an ACL to the vty lines, filtering the addresses that can telnet or SSH into the router or switch. If filtered, the user never sees a login prompt.

For example, imagine that all the network engineering staff's devices connect into subnet 10.1.1.0/24. The security policy states that only the network engineering staff should be allowed to telnet or SSH into any of the Cisco routers in a network. In such a case, the configuration shown in Example 5-4 could be used on each router to deny access from IP addresses not in that subnet.

Example 5-4 *vty Access Control Using the* **access-class** *Command*

```
line vty 0 4
 login
 password cisco
 access-class 3 in
!
! Next command is a global command that matches IPv4 packets with
! a source address that begins with 10.1.1.
 access-list 3 permit 10.1.1.0 0.0.0.255
```

The **access-class** command refers to the matching logic in **access-list 3**. The keyword **in** refers to Telnet and SSH connections into this router—in other words, people telnetting into this router. As configured, ACL 3 checks the source IP address of packets for incoming Telnet connections.

IOS also supports using ACLs to filter outbound Telnet and SSH connections. For example, consider a user who first uses Telnet or SSH to connect to the CLI and now sits in user or enable mode. With an outbound vty filter, IOS will apply ACL logic if the user tries the **telnet** or **ssh** commands to connect *out of the local device* to another device.

To configure an outbound VTY ACL, use the **access-class** *acl* **out** command in VTY configuration mode. Once configured, the router filters any attempts made by current vty users to use the **telnet** and **ssh** commands to initiate new connections to other devices.

Of the two options—to protect inbound and outbound connections—protecting inbound connections is by far the more important and more common. However, to be complete, outbound VTY ACLs have a surprisingly odd feature in how they use the ACL. When the **out** keyword is used, the standard IP ACL listed in the **access-class** command actually looks at the *destination IP address*, and not the source. That is, it filters based on the device to which the **telnet** or **ssh** command is trying to connect.

Firewalls and Intrusion Prevention Systems

The next topic examines the roles of a couple of different kinds of networking devices: firewalls and intrusion prevention systems (IPSs). Both devices work to secure networks but with slightly different goals and approaches.

This second major section of the chapter takes a look at each. This section first discusses the core traditional features of both firewalls and IPSs. The section closes with a description of the newer features in the current generation of these products, called next-generation products, which improves the functions of each.

Traditional Firewalls

Traditionally, a firewall sits in the forwarding path of all packets so that the firewall can then choose which packets to discard and which to allow through. By doing so, the firewall protects the network from different kinds of issues by allowing only the intended types of traffic to flow in and out of the network. In fact, in its most basic form, firewalls do the same kinds of work that routers do with ACLs, but firewalls can perform that packet-filtering function with many more options, as well as perform other security tasks.

Figure 5-5 shows a typical network design for a site that uses a physical firewall. The figure shows a firewall, like the Cisco Adaptive Security Appliance (ASA) firewall, connected to a Cisco router, which in turn connects to the Internet. All enterprise traffic going to or from the Internet would be sent through the firewall. The firewall would consider its rules and make a choice for each packet, whether the packet should be allowed through.

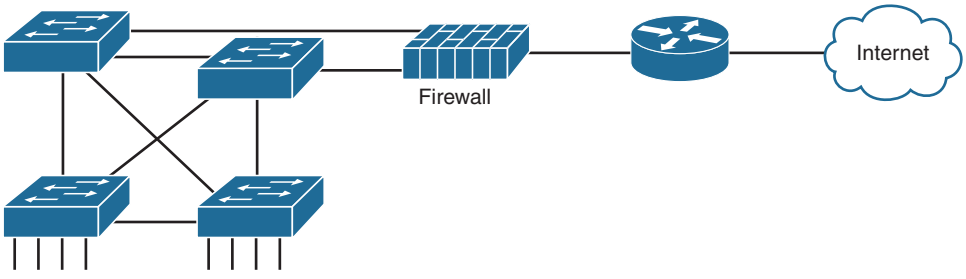


Figure 5-5 *Firewall as Positioned in the Packet Forwarding Path*

Although firewalls have some router-like features (such as packet forwarding and packet filtering), they provide much more advanced security features than a traditional router. For example, most firewalls can use the following kinds of logic to make the choice of whether to discard or allow a packet:

- Like router IP ACLs, match the source and destination IP addresses
- Like router IP ACLs, identify applications by matching their static well-known TCP and UDP ports
- Watch application-layer flows to know what additional TCP and UDP ports are used by a particular flow, and filter based on those ports
- Match the text in the URI of an HTTP request—that is, look at and compare the contents of what is often called the web address—and match patterns to decide whether to allow or deny the download of the web page identified by that URI
- Keep state information by storing information about each packet, and make decisions about filtering future packets based on the historical state information (called *stateful inspection*, or being a stateful firewall)

The stateful firewall feature provides the means to prevent a variety of attacks and is one of the more obvious differences between the ACL processing of a router versus security

filtering by a firewall. Routers must spend as little time as possible processing each packet so that the packets experience little delay passing through the router. The router cannot take the time to gather information about a packet, and then for future packets, consider some saved state information about earlier packets when making a filtering decision. Because they focus on network security, firewalls do save some information about packets and can consider that information for future filtering decisions.

As an example of the benefits of using a stateful firewall, consider a simple denial of service (DoS) attack. An attacker can make this type of attack against a web server by using tools that create (or start to create) a large volume of TCP connections to the server. The firewall might allow TCP connections to that server normally, but imagine that the server might typically receive 10 new TCP connections per second under normal conditions and 100 per second at the busiest times. A DoS attack might attempt thousands or more TCP connections per second, driving up CPU and RAM use on the server and eventually overloading the server to the point that it cannot serve legitimate users.

A stateful firewall could be tracking the number of TCP connections per second—that is, recording state information based on earlier packets—including the number of TCP connection requests from each client IP address to each server address. The stateful firewall could notice a large number of TCP connections, check its state information, and then notice that the number of requests is very large from a small number of clients to that particular server, which is typical of some kinds of DoS attacks. The stateful firewall could then start filtering those packets, helping the web server survive the attack, whereas a stateless firewall or a router ACL would not have had the historical state information to realize that a DoS attack was occurring.

Security Zones

Firewalls not only filter packets, they also pay close attention to which host initiates communications. That concept is most obvious with TCP as the transport layer protocol, where the client initiates the TCP connection by sending a TCP segment that sets the SYN bit only (as seen in Figure 1-5 in Chapter 1, "Introduction to TCP/IP Transport and Applications").

Firewalls use logic that considers which host initiated a TCP connection by watching these initial TCP segments. To see the importance of who initiates the connections, think about a typical enterprise network with a connection to the Internet, as shown in Figure 5-6. The company has users inside the company who open web browsers, initiating connections to web servers across the Internet. However, by having a working Internet connection, that same company opens up the possibility that an attacker might try to create a TCP connection to the company's internal web servers used for payroll processing. Of course, the company does not want random Internet users or attackers to be able to connect to their payroll server.

Firewalls use the concept of *security zones* (also called a *zone* for short) when defining which hosts can initiate new connections. The firewall has rules, and those rules define which host can initiate connections from one zone to another zone. Also, by using zones, a firewall can place multiple interfaces into the same zone, in cases for which multiple interfaces should have the same security rules applied. Figure 5-7 depicts the idea with the inside part of the enterprise considered to be in a separate zone compared to the interfaces connected toward the Internet.

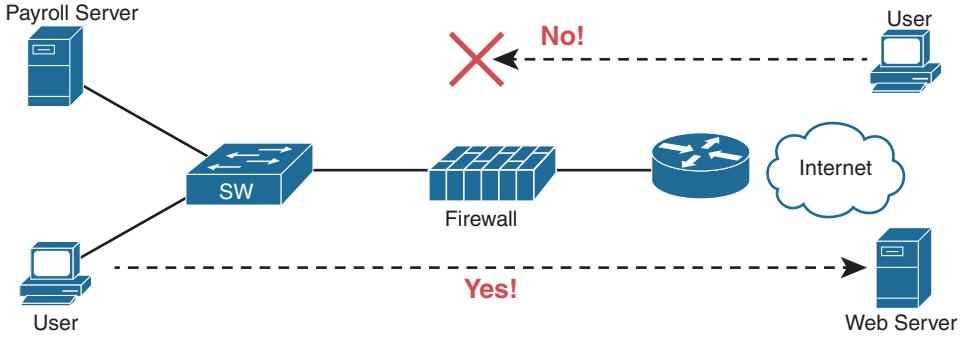


Figure 5-6 *Allowing Outbound Connections and Preventing Inbound Connections*

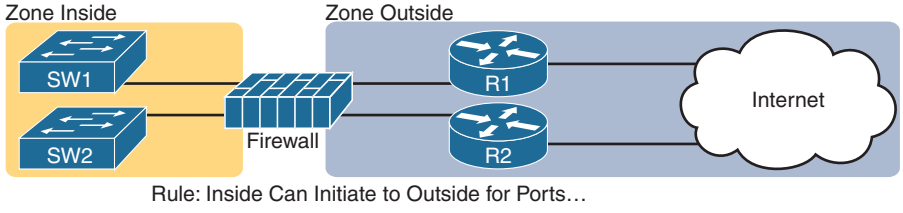


Figure 5-7 *Using Security Zones with Firewalls*

The most basic firewall rule when using two zones like Figure 5-7 reduces to this logic:

> Allow hosts from zone inside to initiate connections to hosts in zone outside, for a predefined set of safe well-known ports (like HTTP port 80, for instance).

Note that with this one simple rule, the correct traffic is allowed while filtering the unwanted traffic by default. Firewalls typically disallow all traffic unless a rule specifically allows the packet. So, with this simple rule to allow inside users to initiate connections to the outside zone, and that alone, the firewall also prevents outside users from initiating connections to inside hosts.

Most companies have an inside and outside zone, as well as a special zone called the *demilitarized zone (DMZ)*. Although the DMZ name comes from the real world, it has been used in IT for decades to refer to a firewall security zone used to place servers that need to be available for use by users in the public Internet. For example, Figure 5-8 shows a typical Internet edge design, with the addition of a couple of web servers in its DMZ connected through the firewall. The firewall then needs another rule that enables users in the zone outside—that is, users in the Internet—to initiate connections to those web servers in the DMZ. By separating those web servers into the DMZ, away from the rest of the enterprise, the enterprise can prevent Internet users from attempting to connect to the internal devices in the inside zone, preventing many types of attacks.

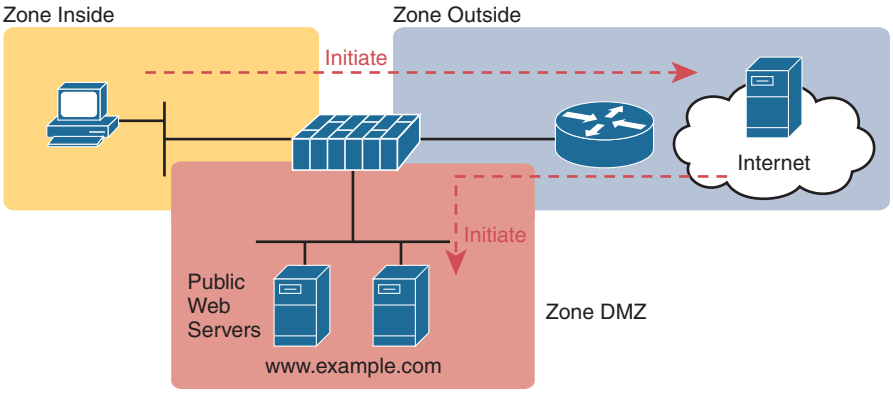


Figure 5-8 *Using a DMZ for Enterprise Servers That Need to Be Accessible from the Internet*

Intrusion Prevention Systems (IPS)

Traditionally, a firewall works with a set of user-configured rules about where packets should be allowed to flow in a network. The firewall needs to sit in the path of the packets so it can filter the packets, redirect them for collection and later analysis, or let them continue toward their destination.

A traditional intrusion prevention system (IPS) can sit in the path packets take through the network, and it can filter packets, but it makes its decisions with different logic. The IPS first downloads a database of exploit signatures. Each signature defines different header field values found in sequences of packets used by different exploits. Then the IPS can examine packets, compare them to the known exploit signatures, and notice when packets may be part of a known exploit. Once identified, the IPS can log the event, discard packets, or even redirect the packets to another security application for further examination.

A traditional IPS differs from firewalls in that instead of an engineer at the company defining rules for that company based on applications (by port number) and zones, the IPS applies the logic based on signatures supplied mostly by the IPS vendor. Those signatures look for these kinds of attacks:

- DoS
- DDoS
- Worms
- Viruses

To accomplish its mission, the IPS needs to download and keep updating its signature database. Security experts work to create the signatures. The IPS must then download the exploit signature database and keep downloading updates over time, as shown in Figure 5-9.

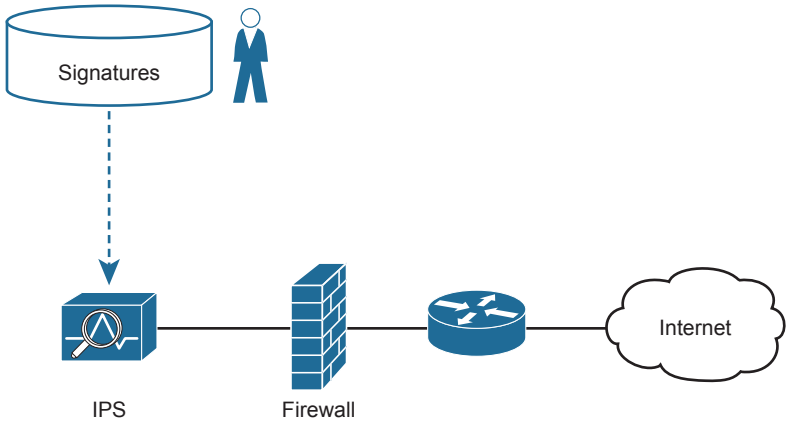


Figure 5-9 *IPS and Signature Database*

For example, think about what happens when an entirely new computer virus has been created. Host-based security products, like antivirus software, should be installed on the computers inside the company. These tools use a similar model as the IPS, keeping an updated database of virus signatures. The signatures might look for patterns in how a computer virus could be stored inside files on the computer, or in files sent to the computer via email or web browsers. But there will be some time lag between the day when the virus has been discovered (called zero-day attacks) and when researchers have developed a virus signature, changed their database, and allowed time for all the hosts to update their antivirus software. The hosts are at risk during this time lag.

The IPS provides a complimentary service to prevent viruses. Researchers will look for ways an IPS could recognize the same virus while in flight through the network with new IPS signatures—for instance, looking for packets with a particular port and a particular hex string in the application payload. Once developed, the IPS devices in the network need to be updated with the new signature database, protecting against that virus. Both the host-based and IPS-based protections play an important role, but the fact that one IPS protects sections of a network means that the IPS can sometimes more quickly react to new threats to protect hosts.

Cisco Next-Generation Firewalls

The CCNA 200-301 exam topics mention the terms *firewall* and *IPS* but prefaced with the term *next-generation*. Around the mid 2010s, Cisco and some of their competitors started using the term *next generation* when discussing their security products to emphasize some of the newer features. In short, a next-generation firewall (NGFW) and a next-generation IPS (NGIPS) are the now-current firewall and IPS products from Cisco.

However, the use of the term *next generation* goes far beyond just a marketing label: the term emphasizes some major shifts and improvements over the years. The security industry sees endless cycles of new attacks followed by new solutions, with some solutions requiring new product features or even new products. Some of the changes that have required new security features include the proliferation of mobile devices—devices that leave the enterprise, connect to the Internet, and return to the Enterprise—creating a whole new level of risk. Also, no single security function or appliance (firewall, IPS, antimalware) can hope to stop some threats, so the next-generation tools must be able to work better together to

provide solutions. In short, the next-generation products have real useful features not found in their predecessor products.

As for Cisco products, for many years Cisco branded its firewalls as the Cisco Adaptive Security Appliance (ASA). Around 2013, Cisco acquired Sourcefire, a security product company. Many of the next-generation firewall (and IPS) features come from software acquired through that acquisition. As of 2019 (when this chapter was written), all of Cisco's currently sold firewalls have names that evoke memories of the Sourcefire acquisition, with most of the firewall product line being called Cisco Firepower firewalls (www.cisco.com/go/firewalls).

An NGFW still does the traditional functions of a firewall, of course, like stateful filtering by comparing fields in the IP, TCP, and UDP headers, and using security zones when defining firewall rules. To provide some insight into some of the newer next-generation features, consider the challenge of matching packets with ports:

1. Each IP-based application should use a well-known port.
2. Attackers know that firewalls will filter most well-known ports from sessions initiated from the outside zone to the inside zone (see Figure 5-8).
3. Attackers use port scanning to find any port that a company's firewall will allow through right now.
4. Attackers attempt to use a protocol of their choosing (for example, HTTP) but with the nonstandard port found through port scanning as a way to attempt to connect to hosts inside the enterprise.

The sequence lists a summary of some of the steps attackers need to take but does not list every single task. However, even to this depth, you can see how attackers can find a way to send packets past the corporate firewall.

The solution? A next-generation firewall that looks at the application layer data to identify the application instead of relying on the TCP and UDP port numbers used. Cisco performs their deep packet inspection using a feature called Application Visibility and Control (AVC). Cisco AVC can identify many applications based on the data sent (application layer headers plus application data structures far past the TCP and UDP headers). When used with a Cisco NGFW, instead of matching port numbers, the firewall matches the application, defeating attacks like the one just described.

The following list mentions a few of the features of an NGFW. Note that while *NGFW* is a useful term, the line between a traditional firewall and a next-generation firewall can be a bit blurry, as the terms describe products that have gone through repeated changes over long periods of time. This list does summarize a few of the key points, however:

Key Topic

- **Traditional firewall:** An NGFW performs traditional firewall features, like stateful firewall filtering, NAT/PAT, and VPN termination.
- **Application Visibility and Control (AVC):** This feature looks deep into the application layer data to identify the application. For instance, it can identify the application based on the data, rather than port number, to defend against attacks that use random port numbers.
- **Advanced Malware Protection:** NGFW platforms run multiple security services, not just as a platform to run a separate service, but for better integration of functions. A network-based antimalware function can run on the firewall itself, blocking file transfers that would install malware, and saving copies of files for later analysis.

- **URL Filtering:** This feature examines the URLs in each web request, categorizes the URLs, and either filters or rate limits the traffic based on rules. The Cisco Talos security group monitors and creates reputation scores for each domain known in the Internet, with URL filtering being able to use those scores in its decision to categorize, filter, or rate limit.
- **NGIPS:** The Cisco NGFW products can also run their NGIPS feature along with the firewall.

Note that for any of the services that benefit from being in the same path that packets traverse, like a firewall, it makes sense that over time those functions could migrate to run on the same product. So, when the design needs both a firewall and IPS at the same location in the network, these NGFW products can run the NGIPS feature as shown in the combined device in Figure 5-10.

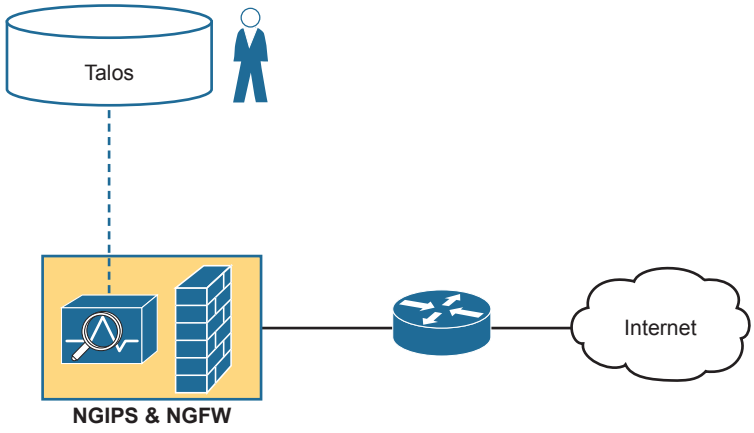


Figure 5-10 *Next-Generation Firewall with Next-Generation IPS Module*

Cisco Next-Generation IPS

The Cisco next-generation IPS (NGIPS) products have followed a similar path as the Cisco NGFW products. Cisco first added NGIPS features primarily through its Sourcefire acquisition, with the now-current (in 2019) Cisco IPS products also using the Firepower name. In fact, as a product line, the hardware NGFW and NGIPS products are the same products, with the ability to run both the NGFW and NGIPS.

As with the NGFW, the NGIPS adds features to a traditional IPS. For instance, one of the biggest issues with a traditional IPS comes with the volume of security events logged by the IPS. For instance:

1. An IPS compares the signature database, which lists all known exploits, to all messages.
2. It generates events, often far more than the security staff can read.
3. The staff must mentally filter events to find the proverbial needle in the haystack, possible only through hard work, vast experience, and a willingness to dig.

An NGIPS helps with this issue in a couple of ways. First, an NGIPS examines the context by gathering data from all the hosts and the users of those hosts. The NGIPS will know the OS, software revision levels, what apps are running, open ports, the transport protocols and port numbers in use, and so on. Armed with that data, the NGIPS can make much more intelligent choices about what events to log.

For instance, consider an NGIPS placed into a network to protect a campus LAN where end users connect, but no data center exists in that part of the network. Also, all PCs happen to be running Windows, and possibly the same version, by corporate policy. The signature database includes signatures for exploits of Linux hosts, Macs, Windows version nonexistent in that part of the network, and exploits that apply to server applications that are not running on those hosts. After gathering those facts, an NGIPS can suggest de-emphasizing checks for exploits that do not apply to those endpoints, spending more time and focus on events that could occur, greatly reducing the number of events logged.

The following list mentions a few of the Cisco NGIPS features:

Key Topic

- **Traditional IPS:** An NGIPS performs traditional IPS features, like using exploit signatures to compare packet flows, creating a log of events, and possibly discarding and/or redirecting packets.
- **Application Visibility and Control (AVC):** As with NGFWs, an NGIPS has the ability to look deep into the application layer data to identify the application.
- **Contextual Awareness:** NGFW platforms gather data from hosts—OS, software version/level, patches applied, applications running, open ports, applications currently sending data, and so on. Those facts inform the NGIPS as to the often more limited vulnerabilities in a portion of the network so that the NGIPS can focus on actual vulnerabilities while greatly reducing the number of logged events.
- **Reputation-Based Filtering:** The Cisco Talos security intelligence group researches security threats daily, building the data used by the Cisco security portfolio. Part of that data identifies known bad actors, based on IP address, domain, name, or even specific URL, with a reputation score for each. A Cisco NGIPS can perform reputation-based filtering, taking the scores into account.
- **Event Impact Level:** Security personnel need to assess the logged events, so an NGIPS provides an assessment based on impact levels, with characterizations as to the impact if an event is indeed some kind of attack.

If you want to learn a little more about these topics for your own interest, let me refer you to a couple of resources. First, check out articles and blog posts from the Cisco Talos Intelligence Group (www.talosintelligence.com). The Cisco Talos organization researches security issues around the globe across the entire spectrum of security products. Additionally, one Cisco Press book has some great information about both next-generation firewalls and IPSs, written at a level appropriate as a next step. If you want to read more, check out this book with the long name: *Integrated Security Technologies and Solutions, Volume I: Cisco Security Solutions for Advanced Threat Protection with Next Generation Firewall, Intrusion Prevention, AMP, and Content Security* (or just use its ISBN, 9781587147067), with one chapter each on NGFW and NGIPS.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 5-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 5-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Do labs		Blog
Review command tables		Book

Review All the Key Topics

Key Topic

Table 5-5 Key Topics for Chapter 5

Key Topic Element	Description	Page Number
List	Commands whose passwords are encrypted by **service password-encryption**	89
List	Rules for when IOS uses the password set with the **enable password** versus **enable secret** commands	91
List	Logic by which IOS can use the **enable secret** hash when a user types a clear-text password to reach enable mode	92
List	Rule for combinations of the **username** command	94
Figure 5-6	Typical client filtering by firewall at Internet edge	98
Figure 5-8	Firewall security zones with DMZ	99
List	Features of next-generation firewalls	101
List	Features of next-generation IPSs	103

Key Terms You Should Know

enable secret, local username, MD5 hash, username secret, firewall, IPS, next-generation firewall (NGFW), next-generation IPS (NGIPS), Application Visibility and Control

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included free with this book. The Sim Lite with this book includes a couple of labs about various password-related topics. Also, check the author's blog site pages for configuration exercises (Config Labs) at https://blog.certskills.com/config-labs.

Command References

Tables 5-6 and 5-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 5-6 Chapter 5 Configuration Commands

Command	Mode/Purpose/Description
line console 0	Command that changes the context to console configuration mode.
line vty *1st-vty last-vty*	Command that changes the context to vty configuration mode for the range of vty lines listed in the command.
login	Console and vty configuration mode. Tells IOS to prompt for a password.
password *pass-value*	Console and vty configuration mode. Lists the password required if the **login** command is configured.
login local	Console and vty configuration mode. Tells IOS to prompt for a username and password, to be checked against locally configured **username** global configuration commands.
username *name* [**algorithm-type md5 | sha256 | scrypt**] **secret** *pass-value*	Global command. Defines one of possibly multiple usernames and associated passwords, stored as a hashed value (default MD5), with other hash options as well.
username *name* **password** *pass-value*	Global command. Defines a username and password, stored in clear text in the configuration by default.
crypto key generate rsa [modulus 512 | 768 | 1024]	Global command. Creates and stores (in a hidden location in flash memory) the keys required by SSH.
transport input {**telnet | ssh | all | none**}	vty line configuration mode. Defines whether Telnet and/or SSH access is allowed into this switch.
[no] service password-encryption	Global command that encrypts all clear-text passwords in the running-config. The **no** version of the command disables the encryption of passwords when the password is set.
enable password *pass-value*	Global command to create the enable password, stored as a clear text instead of a hashed value.
enable [algorithm-type md5 | sha256 | scrypt] secret *pass-value*	Global command to create the enable password, stored as a hashed value instead of clear text, with the hash defined by the algorithm type.
no enable secret **no enable password**	Global command to delete the **enable secret** or **enable password** commands, respectively.
access-class *number* | *name* **in**	A vty mode command that enables inbound ACL checks against Telnet and SSH clients connecting to the router.

Table 5-7 Chapter 5 EXEC Command Reference

Command	Purpose
show running-config | section vty	Lists the vty lines and subcommands from the configuration.
show running-config | section con	Lists the console and subcommands from the configuration.
show running-config | include enable	Lists all lines in the configuration with the word *enable*.

CHAPTER 6

Implementing Switch Port Security

This chapter covers the following exam topics:

5.0 Security Fundamentals

5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)

In modern networks, security must be implemented in depth. The security architecture should use firewalls and intrusion prevention systems (IPS) at strategic locations, and hosts should use antivirus and antimalware tools. Routers, which already need to exist throughout the enterprise at the edge between local-area networks and wide-area networks, can be configured with IP access control lists to filter packets related to different IP address ranges in that enterprise.

LAN switches have a unique opportunity as a security enforcement point, particularly LAN switches connected to endpoint devices. Attackers often launch attacks from the endpoints connected to an enterprise LAN switch. The attacker might gain physical access to the endpoint or first infect the device to then launch an attack. Additionally, a mobile device can become infected while outside the company network and then later connect to the company network, with the attack launching at that point.

Engineers should assume that attacks might be launched from end-user devices connected directly to access ports on the enterprise's LAN switches, so Cisco switches include a number of useful tools to help prevent several types of attacks. This chapter discusses one such tool: port security. Chapter 8, "DHCP Snooping and ARP Inspection," discusses two other switch security tools that take advantage of the switch's access layer role, with Chapter 7, "Implementing DHCP," providing the background details needed to understand the tools in Chapter 8.

This short chapter takes a straightforward approach to the port security feature. The first section discusses the concepts, configuration, and verification, using the primary port security operational mode: shutdown mode. The second section then discusses some of the intricacies of the three operational modes: shutdown, verify, and restrict.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 6-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Port Security Concepts and Configuration	1–3
Port Security Violation Modes	4, 5

1. Which of the following is required when configuring port security with sticky learning?
   a. Setting the maximum number of allowed MAC addresses on the interface with the **switchport port-security maximum** interface subcommand.
   b. Enabling port security with the **switchport port-security** interface subcommand.
   c. Defining the specific allowed MAC addresses using the **switchport port-security mac-address** interface subcommand.
   d. All the other answers list required commands.

2. A Cisco Catalyst switch connects to what should be individual user PCs. Each port has the same port security configuration, configured as follows:

```
interface range gigabitethernet 0/1 - 24
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
```

   Which of the following answers describe the result of the port security configuration created with these commands? (Choose two answers.)
   a. Prevents unknown devices with unknown MAC addresses from sending data through the switch ports.
   b. If a user connects a switch to the cable, prevents multiple devices from sending data through the port.
   c. Will allow any one device to connect to each port and *will* save that device's MAC address into the startup-config.
   d. Will allow any one device to connect to each port but *will not* save that device's MAC address into the startup-config.

3. Which of the following commands list the MAC address table entries for MAC addresses configured by port security? (Choose two answers.)
   a. **show mac address-table dynamic**
   b. **show mac address-table**
   c. **show mac address-table static**
   d. **show mac address-table port-security**

4. The **show port-security interface f0/1** command lists a port status of secure-down. Which one of the following answers must be true about this interface at this time?
   a. The **show interface status** command lists the interface status as connected.
   b. The **show interface status** command lists the interface status as err-disabled.
   c. The **show port-security interface** command could list a mode of shutdown or restrict, but not protect.
   d. The **show port-security interface** command could list a violation counter value of 10.

5. A switch's port Gi0/1 has been correctly enabled with port security. The configuration sets the violation mode to restrict. A frame that violates the port security policy enters the interface, followed by a frame that does not. Which of the following answers correctly describe what happens in this scenario? (Choose two answers.)
   a. The switch puts the interface into an err-disabled state when the first frame arrives.
   b. The switch generates syslog messages about the violating traffic for the first frame.
   c. The switch increments the violation counter for Gi0/1 by 1.
   d. The switch discards both the first and second frame.

Foundation Topics

Port Security Concepts and Configuration

If the network engineer knows what devices should be cabled and connected to particular interfaces on a switch, the engineer can use *port security* to restrict that interface so that only the expected devices can use it. This reduces exposure to attacks in which the attacker connects a laptop to some unused switch port. When that inappropriate device attempts to send frames to the switch interface, the switch can take different actions, ranging from simply issuing informational messages to effectively shutting down the interface.

Port security identifies devices based on the source MAC address of Ethernet frames that the devices send. For example, in Figure 6-1, PC1 sends a frame, with PC1's MAC address as the source address. SW1's F0/1 interface can be configured with port security, and if so, SW1 would examine PC1's MAC address and decide whether PC1 was allowed to send frames into port F0/1.

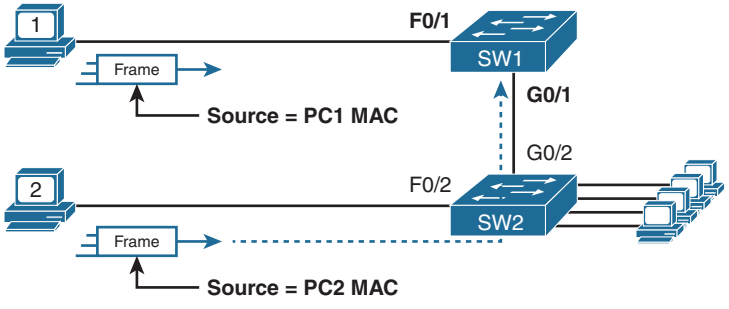


Figure 6-1 *Source MAC Addresses in Frames as They Enter a Switch*

Port security also has no restrictions on whether the frame came from a local device or was forwarded through other switches. For example, switch SW1 could use port security on its G0/1 interface, checking the source MAC address of the frame from PC2, when forwarded up to SW1 from SW2.

Port security has several flexible options, but all operate with the same core concepts. First, switches enable port security per port, with different settings available per port. Each port has a maximum number of allowed MAC addresses, meaning that for all frames entering that port, only that number of *different* source MAC addresses can be used before port security thinks a violation has occurred. When a frame with a new source MAC address arrives, pushing the number of MAC addresses past the allowed maximum, a port security violation occurs. At that point, the switch takes action—by default, discarding all future incoming traffic on that port.

The following list summarizes these ideas common to all variations of port security:

Key Topic

- It examines frames received on the interface to determine if a violation has occurred.
- It defines a maximum number of unique source MAC addresses allowed for all frames coming in the interface.
- It keeps a list and counter of all unique source MAC addresses on the interface.
- It monitors newly learned MAC addresses, considering those MAC addresses to cause a violation if the newly learned MAC address would push the total number of MAC table entries for the interface past the configured maximum allowed MAC addresses for that port.
- It takes action to discard frames from the violating MAC addresses, plus other actions depending on the configured violation mode.

Those rules define the basics, but port security allows other options as well, including options like these:

- Define a maximum of three MAC addresses, defining all three specific MAC addresses.
- Define a maximum of three MAC addresses but allow those addresses to be dynamically learned, allowing the first three MAC addresses learned.
- Define a maximum of three MAC addresses, predefining one specific MAC address, and allowing two more to be dynamically learned.

You might like the idea of predefining the MAC addresses for port security, but finding the MAC address of each device can be a bother. Port security provides a useful compromise using a feature called *sticky secure MAC addresses*. With this feature, port security learns the MAC addresses off each port so that you do not have to preconfigure the values. It also adds the learned MAC addresses to the port security configuration (in the running-config file). This feature helps reduce the big effort of finding out the MAC address of each device.

As you can see, port security has a lot of detailed options. The next few sections walk you through these options to pull the ideas together.

Configuring Port Security

Port security configuration involves several steps. First, port security works on both access ports and trunk ports, but it requires you to statically configure the port as a trunk or an

access port, rather than let the switch dynamically decide whether to use trunking. The following configuration checklist details how to enable port security, set the maximum allowed MAC addresses per port, and configure the actual MAC addresses:

Key Topic

Step 1. Use the **switchport mode access** or the **switchport mode trunk** interface subcommands, respectively, to make the switch interface either a static access or trunk interface.

Step 2. Use the **switchport port-security** interface subcommand to enable port security on the interface.

Step 3. (Optional) Use the **switchport port-security maximum** *number* interface subcommand to override the default maximum number of allowed MAC addresses associated with the interface (1).

Step 4. (Optional) Use the **switchport port-security violation** {**protect** | **restrict** | **shutdown**} interface subcommand to override the default action to take upon a security violation (shutdown).

Step 5. (Optional) Use the **switchport port-security mac-address** *mac-address* interface subcommand to predefine any allowed source MAC addresses for this interface. Use the command multiple times to define more than one MAC address.

Step 6. (Optional) Use the **switchport port-security mac-address sticky** interface subcommand to tell the switch to "sticky learn" dynamically learned MAC addresses.

To demonstrate how to configure this variety of the settings, Figure 6-2 and Example 6-1 show four examples of port security. Three ports operate as access ports, while port F0/4, connected to another switch, operates as a trunk.

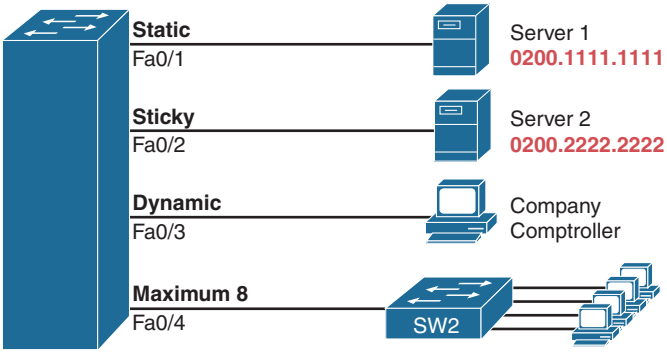


Figure 6-2 *Port Security Configuration Example*

Answers to the "Do I Know This Already?" quiz:

1 B **2** B, D **3** B, C **4** B **5** B, C

Key Topic

Example 6-1 *Variations on Port Security Configuration*

```
SW1# show running-config
(Lines omitted for brevity)

interface FastEthernet0/1
 switchport mode access
 switchport port-security
 switchport port-security mac-address 0200.1111.1111
!
interface FastEthernet0/2
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
!
interface FastEthernet0/3
 switchport mode access
 switchport port-security
!
interface FastEthernet0/4
 switchport mode trunk
 switchport port-security
 switchport port-security maximum 8
```


First, scan the configuration for all four interfaces in Example 6-1, focusing on the first two interface subcommands in each case. Note that the first three interfaces in the example use the same first two interface subcommands, matching the first two configuration steps noted before Figure 6-2. The **switchport port-security** command enables port security, with all defaults, with the **switchport mode access** command meeting the requirement to configure the port as either an access or trunk port. The final port, F0/4, has a similar configuration, except that it has been configured as a trunk rather than as an access port.

Next, scan all four interfaces again, and note that the configuration differs on each interface after those first two interface subcommands. Each interface simply shows a different example for perspective.

The first interface, FastEthernet 0/1, adds one optional port security subcommand: **switchport port-security mac-address 0200.1111.1111**, which defines a specific source MAC address. With the default maximum source address setting of 1, only frames with source MAC 0200.1111.1111 will be allowed in this port. When a frame with a source other than 0200.1111.1111 enters F0/1, the switch would normally perform MAC address learning and want to add the new source MAC address to the MAC address table. Port security will see that action as learning one too many MAC addresses on the port, taking the default violation action to disable the interface.

As a second example, FastEthernet 0/2 uses the same logic as FastEthernet 0/1, except that it uses the sticky learning feature. For port F0/2, the configuration of the **switchport port-security mac-address sticky** command tells the switch to dynamically learn source MAC addresses and add **port-security** commands to the running-config. Example 6-2 shows the running-config file that lists the sticky-learned MAC address in this case.

Example 6-2 *Configuration Added by the Port Security Sticky Feature*

```
SW1# show running-config interface f0/2
Building configuration...
Current configuration : 188 bytes
!
interface FastEthernet0/2
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
 switchport port-security mac-address sticky 0200.2222.2222
```

Port security does not save the configuration of the sticky addresses, so use the **copy running-config startup-config** command if desired.

The other two interfaces in Example 6-1 do not predefine MAC addresses, nor do they sticky-learn the MAC addresses. The only difference between these two interfaces' port security configuration is that FastEthernet 0/4 supports eight MAC addresses because it connects to another switch and should receive frames with multiple source MAC addresses. Interface F0/3 uses the default maximum of one MAC address.

NOTE Switches can also use port security on voice ports and EtherChannels. For voice ports, make sure to configure the maximum MAC address to at least two (one for the phone, or for a PC connected to the phone). On EtherChannels, the port security configuration should be placed on the port-channel interface, rather than the individual physical interfaces in the channel.

Verifying Port Security

The **show port-security interface** command provides the most insight to how port security operates, as shown in Example 6-3. This command lists the configuration settings for port security on an interface; plus it lists several important facts about the current operation of port security, including information about any security violations. The two commands in the example show interfaces F0/1 and F0/2, based on Example 6-1's configuration.

Example 6-3 *Using Port Security to Define Correct MAC Addresses of Particular Interfaces*

```
SW1# show port-security interface fastEthernet 0/1
Port Security              : Enabled
Port Status                : Secure-shutdown
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0013.197b.5004:1
```

```
Security Violation Count   : 1

SW1# show port-security interface fastEthernet 0/2
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 1
Last Source Address:Vlan   : 0200.2222.2222:1
Security Violation Count   : 0
```

The two commands in Example 6-3 confirm that a security violation has occurred on FastEthernet 0/1, but no violations have occurred on FastEthernet 0/2. The **show port- security interface fastethernet 0/1** command shows that the interface is in a *secure-shutdown* state, which means that the interface has been disabled because of port security. In this case, another device connected to port F0/1, sending a frame with a source MAC address other than 0200.1111.1111, is causing a violation. However, port Fa0/2, which used sticky learning, simply learned the MAC address used by Server 2.

Port Security MAC Addresses

To complete this chapter, take a moment to think about Layer 2 switching, along with all those examples of output from the **show mac address-table dynamic** EXEC command.

Once a switch port has been configured with port security, the switch no longer considers MAC addresses associated with that port as being dynamic entries as listed with the **show mac address-table dynamic** EXEC command. Even if the MAC addresses are dynamically learned, once port security has been enabled, you need to use one of these options to see the MAC table entries associated with ports using port security:

- **show mac address-table secure:** Lists MAC addresses associated with ports that use port security
- **show mac address-table static:** Lists MAC addresses associated with ports that use port security, as well as any other statically defined MAC addresses

Example 6-4 proves the point. It shows two commands about interface F0/2 from the port security example shown in Figure 6-2 and Example 6-1. In that example, port security was configured on F0/2 with sticky learning, so from a literal sense, the switch learned a MAC address off that port (0200.2222.2222). However, the **show mac address-table dynamic** command does not list the address and port because IOS considers that MAC table entry to be a static entry. The **show mac address-table secure** command does list the address and port.

Example 6-4 *Using the* **secure** *Keyword to See MAC Table Entries When Using Port Security*

```
SW1# show mac address-table secure interface F0/2
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type         Ports
----    -----------       --------     -----
 1      0200.2222.2222    STATIC       Fa0/2
Total Mac Addresses for this criterion: 1

SW1# show mac address-table dynamic interface f0/2
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type         Ports
----    -----------       --------     -----
SW1#
```

Port Security Violation Modes

The first half of the chapter discussed many details of port security, but it mostly ignored one major feature: the port security violation mode. The violation mode defines how port security should react when a violation occurs.

First, to review, what is a port security violation? Any received frame that breaks the port security rules on an interface. For example:

- For an interface that allows any two MAC addresses, a violation occurs when the total of preconfigured and learned MAC addresses on the interface exceeds the configured maximum of two.
- For an interface that predefines all the specific MAC addresses allowed on the interface, a violation occurs when the switch receives a frame whose source MAC is not one of those configured addresses.

With port security, each switch port can be configured to use one of three violation modes that defines the actions to take when a violation occurs. All three options cause the switch to discard the offending frame (a frame whose source MAC address would push the number of learned MAC addresses over the limit). However, the modes vary in how many other steps they take. For instance, some modes include the action of the switch generating syslog messages and SNMP Trap messages, while some define the action to disable the interface. Table 6-2 lists the three modes, their actions, along with the keywords that enable each mode on the **switchport port-security violation** {**protect | restrict | shutdown**} interface subcommand.

Key Topic

Table 6-2 Actions When Port Security Violation Occurs

Option on the switchport port-security violation Command	Protect	Restrict	Shutdown
Discards offending traffic	Yes	Yes	Yes
Sends log and SNMP messages	No	Yes	Yes
Disables the interface by putting it in an err-disabled state, discarding all traffic	No	No	Yes

Because IOS reacts so differently with shutdown mode as compared to restrict and protect modes, the next few pages explain the differences—first for shutdown mode, then for the other two modes.

Port Security Shutdown Mode

When the (default) shutdown violation mode is used and a port security violation occurs on a port, port security stops all frame forwarding on the interface, both in and out of the port. In effect, it acts as if port security has shut down the port; however, it does not literally configure the port with the **shutdown** interface subcommand. Instead, port security uses the err-disabled feature. Cisco switches use the err-disabled state for a wide range of purposes, but when using port security shutdown mode and a violation occurs, the following happens:

Key Topic

- The switch interface state (per **show interfaces** and **show interfaces status**) changes to an err-disabled state.
- The switch interface port security state (per **show port-security**) changes to a secure-down state.
- The switch stops sending and receiving frames on the interface.

Once port security has placed a port in err-disabled state, by default the port remains in an err-disabled state until someone takes action. To recover from an err-disabled state, the interface must be shut down with the **shutdown** command and then enabled with the **no shutdown** command. Alternately, the switch can be configured to automatically recover from the err-disabled state, when caused by port security, with these commands:

- **errdisable recovery cause psecure-violation:** A global command to enable automatic recovery for interfaces in an err-disabled state caused by port security
- **errdisable recovery interval** *seconds*: A global command to set the time to wait before recovering the interface

To take a closer look at shutdown mode, start by checking the configuration state of the switch. You can check the port security configuration on any interface with the **show port-security interface** *type number* command, as seen back in Example 6-2, but the **show port-security** command (as listed in Example 6-5) shows briefer output, with one line per enabled interface.

Example 6-5 *Confirming the Port Security Violation Mode*

```
SW1# show port-security
Secure Port  MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
                (Count)       (Count)          (Count)
---------------------------------------------------------------------------
     Fa0/13              1            1                  1         Shutdown
---------------------------------------------------------------------------
Total Addresses in System (excluding one mac per port) : 0
Max Addresses limit in System (excluding one mac per port) : 8192
```

Note that for these next examples, a switch has configured port security on port Fa0/13 only. In this case, the switch appears to be configured to support one MAC address, has already reached that total, and has a security violation action of "shutdown."

Next, Example 6-6 shows the results after a port security violation has already occurred on port F0/13. The first command confirms the err-disabled state (per the **show interfaces status** command) and the secure-shutdown state (per the **show port-security** command).

Example 6-6 *Port Security Status in Shutdown Mode After a Violation*

```
! The next lines show the log message generated when the violation occurred.
Jul 31 18:00:22.810: %PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred,
caused by MAC address d48c.b57d.8200 on port FastEthernet0/13


! The next command shows the err-disabled state, implying a security violation.
SW1# show interfaces Fa0/13 status

Port     Name                Status        Vlan   Duplex  Speed  Type
Fa0/13                       err-disabled  1      auto     auto  10/100BaseTX
!
! The next command's output has shading for several of the most important facts.
SW1# show port-security interface Fa0/13
Port Security               : Enabled
Port Status                 : Secure-shutdown
Violation Mode              : Shutdown
Aging Time                  : 0 mins
Aging Type                  : Absolute
SecureStatic Address Aging  : Disabled
Maximum MAC Addresses       : 1
Total MAC Addresses         : 1
Configured MAC Addresses    : 1
Sticky MAC Addresses        : 0
Last Source Address:Vlan    : 0200.3333.3333:2
Security Violation Count    : 1
```

The output of the **show port-security interface** command lists the current port-security status (secure-shutdown) as well as the configured mode (shutdown). The last line of output lists the number of violations that caused the interface to fail to an err-disabled state, while

the second-to-last line identifies the MAC address and VLAN of the device that caused the violation.

Figure 6-3 summarizes these behaviors, assuming the same scenario shown in the example.

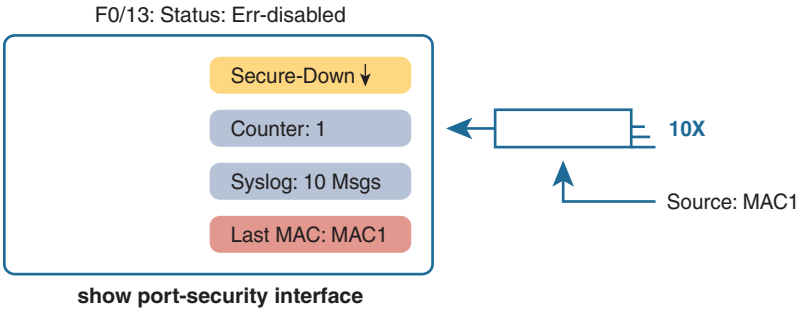


Figure 6-3 *Summary of Actions: Port Security Violation Mode Shutdown*

Note that the violations counter notes the number of times the interface has been moved to the err-disabled (secure-shutdown) state. For instance, the first time it fails, the counter increments to 1; while err-disabled, many frames can arrive, but the counter remains at 1. Later, after an engineer has recovered the interface from the err-disabled state with a **shutdown/no shutdown**, another violation that causes the interface to fail to an err-disabled state will cause the counter to increment to 2.

Port Security Protect and Restrict Modes

The restrict and protect violation modes take a much different approach to securing ports. These modes still discard offending traffic, but the interface remains in a connected (up/up) state and in a port security state of secure-up. As a result, the port continues to forward good traffic but discards offending traffic.

Having a port in a seemingly good state that also discards traffic can be a challenge when troubleshooting. Basically, you have to know about the feature and then know how to tell when port security is discarding some traffic on a port even though the interface status looks good.

With protect mode, the only action the switch takes for a frame that violates the port security rules is to discard the frame. The switch does not change the port to an err-disabled state, does not generate messages, and does not even increment the violations counter.

Example 6-7 shows a sample with protect mode after several violations have occurred. Note that the **show** command confirms the mode (protect) as configured in the top part of the example, with a port security state of secure-up—a state that will not change in protect mode. Also, note that the counter at the bottom shows 0, even though several violations have occurred, because protect mode does not count the violating frames.

Example 6-7 *Port Security Using Protect Mode*

```
SW1# show running-config
! Lines omitted for brevity
interface FastEthernet0/13
  switchport mode access
  switchport port-security
```

```
  switchport port-security mac-address 0200.1111.1111
  switchport port-security violation protect
! Lines omitted for brevity

SW1# show port-security interface Fa0/13
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Protect
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0
```

NOTE The small particulars of the violation counters and last source address might be slightly different with some older switch models and IOS versions. Note that this edition's testing is based on 2960XR switches running IOS 15.2.(6)E2.

While shutdown mode disables the interface, and protect mode does nothing more than discard the offending traffic, restrict mode provides a compromise between the other two modes. If Example 6-7 had used the restrict violation mode instead of protect, the port status would have also remained in a secure-up state; however, IOS would show some indication of port security activity, such as an accurate incrementing violation counter, as well as syslog messages. Example 6-8 shows an example of the violation counter and ends with an example port security syslog message. In this case, 97 incoming frames so far violated the rules, with the most recent frame having a source MAC address of 0200.3333.3333 in VLAN 1.

Example 6-8 *Port Security Using Violation Mode Restrict*

```
SW1# show port-security interface fa0/13
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Restrict
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0200.3333.3333:1
Security Violation Count   : 97
```

```
!
! The following log message also points to a port security issue.
!
01:46:58: %PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused by
MAC address 0200.3333.3333 on port FastEthernet0/13.
```

Figure 6-4 summarizes the key points about the restrict mode for port security. In this case, the figure matches the same scenario as the example again, with 97 total violating frames arriving so far, with the most recent being from source MAC address MAC3.

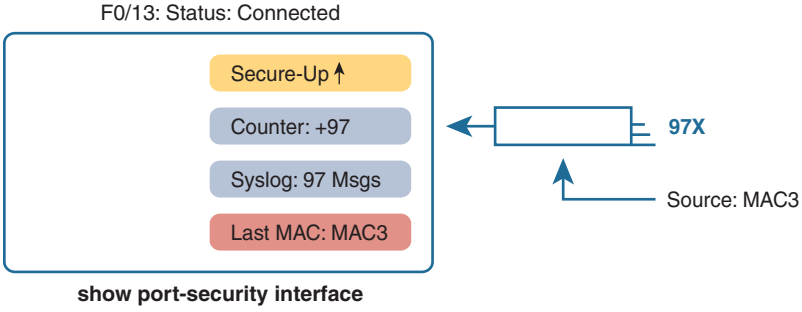


Figure 6-4 *Summary of Actions: Port Security Violation Mode Restrict*

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 6-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 6-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Review memory tables		Book, website
Review config checklists		Book, website
Do labs		Sim Lite, blog
Watch Video		Website

Review All the Key Topics

Key Topic

Table 6-4 Key Topics for Chapter 6

Key Topic Element	Description	Page Number
List	Summary of port security concepts	109
List	Port security configuration checklist	110
Example 6-1	Port security configuration samples	111
Table 6-2	Port security actions and the results of each action	115
List	Switch actions when a port security violation occurs	115

Key Terms You Should Know

port security, violation mode, error disabled (err-disable)

Do Labs

The Sim Lite software is a version of Pearson's full simulator learning product with a subset of the labs, included free with this book. The Sim Lite with this book includes a couple of labs about port security. Also, check the author's blog site pages for configuration exercises (Config Labs) at https://blog.certskills.com/config-labs.

Command References

Tables 6-5 and 6-6 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 6-5 Chapter 6 Configuration Command Reference

Command	Mode/Purpose/Description
switchport mode {access | trunk}	Interface configuration mode command that tells the switch to always be an access port, or always be a trunk port
switchport port-security mac-address *mac-address*	Interface configuration mode command that statically adds a specific MAC address as an allowed MAC address on the interface
switchport port-security mac-address sticky	Interface subcommand that tells the switch to learn MAC addresses on the interface and add them to the configuration for the interface as secure MAC addresses
switchport port-security maximum *value*	Interface subcommand that sets the maximum number of static secure MAC addresses that can be assigned to a single interface
switchport port-security violation {protect | restrict | shutdown}	Interface subcommand that tells the switch what to do if an inappropriate MAC address tries to access the network through a secure switch port

Command	Mode/Purpose/Description
errdisable recovery cause psecure-violation	Global command that enables the automatic recovery from err-disabled state for ports that reach that state due to port security violations
errdisable recovery interval *seconds*	Global command that sets the delay, in seconds, before a switch attempts to recover an interface in err-disabled mode, regardless of the reason for that interface being in that state
shutdown **no shutdown**	Interface subcommands that administratively disable and enable an interface, respectively

Table 6-6 Chapter 6 EXEC Command Reference

Command	Purpose
show running-config	Lists the currently used configuration
show running-config | interface *type number*	Displays the running-configuration excerpt of the listed interface and its subcommands only
show mac address-table dynamic [**interface** *type number*]	Lists the dynamically learned entries in the switch's address (forwarding) table
show mac address-table secure [**interface** *type number*]	Lists MAC addresses defined or learned on ports configured with port security
show mac address-table static [**interface** *type number*]	Lists static MAC addresses and MAC addresses learned or defined with port security
show interfaces [**interface** *type number*] **status**	Lists one output line per interface (or for only the listed interface if included), noting the description, operating state, and settings for duplex and speed on each interface
show port-security interface *type number*	Lists an interface's port security configuration settings and security operational status
show port-security	Lists one line per interface that summarizes the port security settings for any interface on which it is enabled

CHAPTER 7

Implementing DHCP

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.10 Identify IP parameters for Client OS (Windows, Mac OS, Linux)

4.0 IP Services

4.3 Explain the role of DHCP and DNS within the network

4.6 Configure and verify DHCP client and relay

In the world of TCP/IP, the word *host* refers to any device with an IP address: your phone, your tablet, a PC, a server, a router, a switch—any device that uses IP to provide a service or just needs an IP address to be managed. The term *host* includes some less-obvious devices as well: the electronic advertising video screen at the mall, your electrical power meter that uses the same technology as mobile phones to submit your electrical usage information for billing, your new car.

No matter the type of host, any host that uses IPv4 needs four IPv4 settings to work properly:

- IP address
- Subnet mask
- Default routers
- DNS server IP addresses

This chapter discusses these basic IP settings on hosts. The chapter begins by discussing how a host can dynamically learn these four settings using the Dynamic Host Configuration Protocol (DHCP). The second half of this chapter then shows how to find the settings on hosts and the key facts to look for when displaying the settings.

Just a note about the overall flow of the chapters: This chapter does not discuss security topics, although it sits inside Part II, "Security Services." I located this DHCP-focused chapter here because Chapter 8, "DHCP Snooping and ARP Inspection," relies heavily on knowledge of DHCP.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 7-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Dynamic Host Configuration Protocol	1–4
Identifying Host IPv4 Settings	5, 6

1. A PC connects to a LAN and uses DHCP to lease an IP address for the first time. Of the usual four DHCP messages that flow between the PC and the DHCP server, which ones do the client send? (Choose two answers.)
   a. Acknowledgment
   b. Discover
   c. Offer
   d. Request
2. Which of the following kinds of information are part of a DHCP server configuration? (Choose two answers.)
   a. Ranges of IP addresses in subnets that the server should lease
   b. Ranges of IP addresses to not lease per subnet
   c. DNS server hostnames
   d. The default router IP and MAC address in each subnet
3. Which answers list a criterion for choosing which router interfaces need to be configured as a DHCP relay agent? (Choose two answers.)
   a. If the subnet off the interface does not include a DHCP server
   b. If the subnet off the interface does include a DHCP server
   c. If the subnet off the interface contains DHCP clients
   d. If the router interface already has an **ip address dhcp** command
4. A router connects to an Internet Service Provider (ISP) using its G0/0/0 interface, with the **ip address dhcp** command configured. What does the router do with the DHCP-learned default gateway information?
   a. The router ignores the default gateway value learned from the DHCP server.
   b. The router uses the default gateway just like a host, ignoring its routing table.
   c. The router forwards received packets based on its routing table but uses its default gateway setting to forward packets it generates itself.
   d. The router adds a default route based on the default gateway to its IP routing table.

5. In the following excerpt from a command on a Mac, which of the following parts of the output represent information learned from a DHCP server? (Choose two answers.)

```
Macprompt$ ifconfig en0
En1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=10b<RXCSUM,TXCSUM,VLAN_HWTAGGING,AV>
        ether 00:6d:e7:b1:9a:11
        inet 172.16.4.2 netmask 0xffffff00 broadcast 172.16.4.255
```

   a. 00:6d:e7:b1:9a:11
   b. 172.16.4.2
   c. 0xffffff00
   d. 172.16.4.255

6. Which of the following commands on a Windows OS should list both the IP address and DNS servers as learned with DHCP?

   a. **ifconfig**
   b. **ipconfig**
   c. **ifconfig /all**
   d. **ipconfig /all**

Foundation Topics

Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) provides one of the most commonly used services in a TCP/IP network. The vast majority of hosts in a TCP/IP network are user devices, and the vast majority of user devices learn their IPv4 settings using DHCP.

Using DHCP has several advantages over the other option of manually configuring IPv4 settings. The configuration of host IP settings sits in a DHCP server, with each client learning these settings using DHCP messages. As a result, the host IP configuration is controlled by the IT staff, rather than on local configuration on each host, resulting in fewer user errors. DHCP allows both the permanent assignment of host addresses, but more commonly, DHCP assigns a temporary lease of IP addresses. With these leases, the DHCP server can reclaim IP addresses when a device is removed from the network, making better use of the available addresses.

DHCP also enables mobility. For example, every time a user moves to a new location with a tablet computer—to a coffee shop, a client location, or back at the office—the user's device can connect to another wireless LAN, use DHCP to lease a new IP address in that LAN, and begin working on the new network. Without DHCP, the user would have to ask for information about the local network and configure settings manually, with more than a few users making mistakes.

Although DHCP works automatically for user hosts, it does require some preparation from the network, with some configuration on routers. In some enterprise networks, that router

configuration can be a single command on many of the router's LAN interfaces (**ip helper-address** *server-ip*), which identifies the DHCP server by its IP address. In other cases, the router acts as the DHCP server. Regardless, the routers have some role to play.

This first major section of the chapter takes a tour of DHCP, including concepts and the router configuration to enable the routers to work well with a separate DHCP server.

DHCP Concepts

Sit back for a moment and think about the role of DHCP for a host computer. The host acts as a DHCP client. As a DHCP client, the host begins with no IPv4 settings—no IPv4 address, no mask, no default router, and no DNS server IP addresses. But a DHCP client does have knowledge of the DHCP protocol, so the client can use that protocol to (a) discover a DHCP server and (b) request to lease an IPv4 address.

DHCP uses the following four messages between the client and server. (Also, as a way to help remember the messages, note that the first letters spell DORA):

Discover: Sent by the DHCP client to find a willing DHCP server

Offer: Sent by a DHCP server to offer to lease to that client a specific IP address (and inform the client of its other parameters)

Request: Sent by the DHCP client to ask the server to lease the IPv4 address listed in the Offer message

Acknowledgment: Sent by the DHCP server to assign the address and to list the mask, default router, and DNS server IP addresses

DHCP clients, however, have a somewhat unique problem: they do not have an IP address yet, but they need to send these DHCP messages inside IP packets. To make that work, DHCP messages make use of two special IPv4 addresses that allow a host that has no IP address to still be able to send and receive messages on the local subnet:

Key Topic

0.0.0.0: An address reserved for use as a source IPv4 address for hosts that do not yet have an IP address.

255.255.255.255: The local broadcast IP address. Packets sent to this destination address are broadcast on the local data link, but routers do not forward them.

To see how these addresses work, Figure 7-1 shows an example of the IP addresses used between a host (A) and a DHCP server on the same LAN. Host A, a client, sends a Discover message, with source IP address of 0.0.0.0 because host A does not have an IP address to use yet. Host A sends the packet to destination 255.255.255.255, which is sent in a LAN broadcast frame, reaching all hosts in the subnet. The client hopes that there is a DHCP server on the local subnet. Why? Packets sent to 255.255.255.255 only go to hosts in the local subnet; router R1 will not forward this packet.

NOTE Figure 7-1 shows one example of the addresses that can be used in a DHCP request. This example shows details assuming the DHCP client chooses to use a DHCP option called the *broadcast flag*; all examples in this book assume the broadcast flag is used.

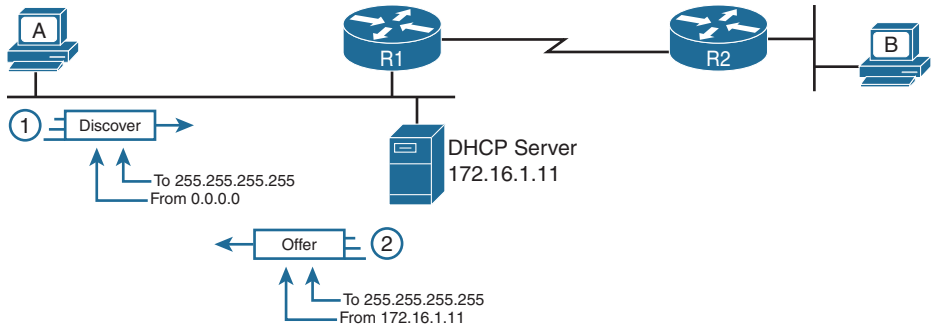


Figure 7-1 *DHCP Discover and Offer*

Now look at the Offer message sent back by the DHCP server. The server sets the destination IP address to 255.255.255.255 again. Why? Host A still does not have an IP address, so the server cannot send a packet directly to host A. So, the server sends the packet to "all local hosts in the subnet" address (255.255.255.255). (The packet is also encapsulated in an Ethernet broadcast frame.)

Note that all hosts in the subnet receive the Offer message. However, the original Discover message lists a number called the client ID, which includes the host's MAC address, that identifies the original host (host A in this case). As a result, host A knows that the Offer message is meant for host A. The rest of the hosts will receive the Offer message, but notice that the message lists another device's DHCP client ID, so the rest of the hosts ignore the Offer message.

Supporting DHCP for Remote Subnets with DHCP Relay

Network engineers have a major design choice to make with DHCP: Do they put a DHCP server in every LAN subnet or locate a DHCP server in a central site? The question is legitimate. Cisco routers can act as the DHCP server, so a distributed design could use the router at each site as the DHCP server. With a DHCP server in every subnet, as shown in Figure 7-1, the protocol flows stay local to each LAN.

However, a centralized DHCP server approach has advantages as well. In fact, some Cisco design documents suggest a centralized design as a best practice, in part because it allows for centralized control and configuration of all the IPv4 addresses assigned throughout the enterprise.

With a centralized DHCP server, those DHCP messages that flowed only on the local subnet in Figure 7-1 somehow need to flow over the IP network to the centralized DHCP server and back. To make that work, the routers connected to the remote LAN subnets need an interface subcommand: the **ip helper-address** *server-ip* command.

The **ip helper-address** *server-ip* subcommand tells the router to do the following for the messages coming in an interface, from a DHCP client:

Answers to the "Do I Know This Already?" quiz:

1 B, D **2** A, B **3** A, C **4** D **5** B, C **6** D

Key Topic

1. Watch for incoming DHCP messages, with destination IP address 255.255.255.255.
2. Change that packet's source IP address to the router's incoming interface IP address.
3. Change that packet's destination IP address to the address of the DHCP server (as configured in the **ip helper-address** command).
4. Route the packet to the DHCP server.

This command gets around the "do not route packets sent to 255.255.255.255" rule by changing the destination IP address. Once the destination has been set to match the DHCP server's IP address, the network can route the packet to the server.

NOTE This feature, by which a router relays DHCP messages by changing the IP addresses in the packet header, is called *DHCP relay*.

Figure 7-2 shows an example of the process. Host A sits on the left, as a DHCP client. The DHCP server (172.16.2.11) sits on the right. R1 has an **ip helper-address 172.16.2.11** command configured, under its G0/0 interface. At step 1, router R1 notices the incoming DHCP packet destined for 255.255.255.255. Step 2 shows the results of changing both the source and destination IP address, with R1 routing the packet.

Key Topic

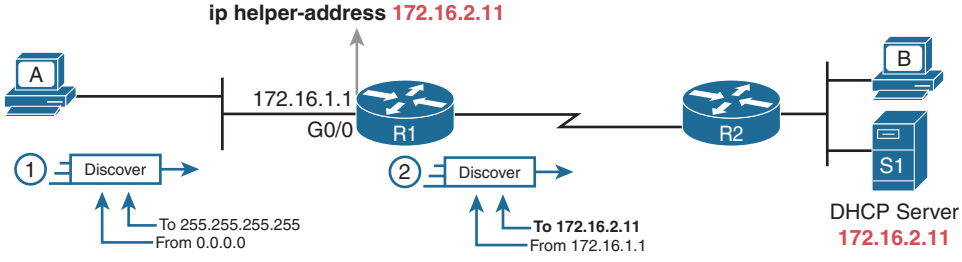


Figure 7-2 *IP Helper Address Effect*

The router uses a similar process for the return DHCP messages from the server. First, for the return packet from the DHCP server, the server simply reverses the source and destination IP address of the packet received from the router (relay agent). For example, in Figure 7-2, the Discover message lists source IP address 172.16.1.1, so the server sends the Offer message back to destination IP address 172.16.1.1.

When a router receives a DHCP message, addressed to one of the router's own IP addresses, the router realizes the packet might be part of the DHCP relay feature. When that happens, the DHCP relay agent (router R1) needs to change the destination IP address, so that the real DHCP client (host A), which does not have an IP address yet, can receive and process the packet.

Figure 7-3 shows one example of how these addresses work, when R1 receives the DHCP Offer message sent to R1's own 172.16.1.1 address. R1 changes the packet's destination to 255.255.255.255 and forwards it out G0/0, because the packet was destined to G0/0's 172.16.1.1 IP address. As a result, all hosts in that LAN (including the DHCP client A) will receive the message.

Many enterprise networks use a centralized DHCP server, so the normal router configuration includes an **ip helper-address** command on every LAN interface/subinterface. With that standard configuration, user hosts off any router LAN interface can always reach the DHCP server and lease an IP address.

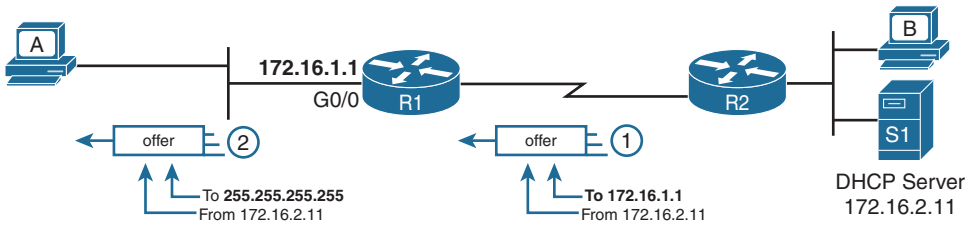


Figure 7-3 *IP Helper Address for the Offer Message Returned from the DHCP Server*

Information Stored at the DHCP Server

A DHCP server might sound like some large piece of hardware, sitting in a big locked room with lots of air conditioning to keep the hardware cool. However, like most servers, the server is actually software, running on some server OS. The DHCP server could be a piece of software downloaded for free and installed on an old PC. However, because the server needs to be available all the time, to support new DHCP clients, most companies install the software on a very stable and highly available data center, with high availability features. The DHCP service is still created by software, however.

To be ready to answer DHCP clients and to supply them with an IPv4 address and other information, the DHCP server (software) needs configuration. DHCP servers typically organize these IPv4 settings per subnet, because the information the server tells the client is usually the same for all hosts in the same subnet, but slightly different for hosts in different subnets. For example, IP addressing rules tell us that all hosts on the same subnet should use the same mask but hosts in different subnets would have a different default gateway setting.

The following list shows the types of settings the DHCP server needs to know to support DHCP clients:

Subnet ID and mask: The DHCP server can use this information to know all addresses in the subnet. (The DHCP server knows to not lease the subnet ID or subnet broadcast address.)

Reserved (excluded) addresses: The server needs to know which addresses in the subnet to *not* lease. This list allows the engineer to reserve addresses to be used as static IP addresses. For example, most router and switch IP addresses, server addresses, and addresses of most anything other than user devices use a statically assigned IP address. Most of the time, engineers use the same convention for all subnets, either reserving the lowest IP addresses in all subnets or reserving the highest IP addresses in all subnets.

Default router(s): This is the IP address of the router on that subnet.

DNS IP address(es): This is a list of DNS server IP addresses.

Figure 7-4 shows the concept behind the preconfiguration on a DHCP server for two LAN-based subnets, 172.16.1.0/24 and 172.16.2.0/24. The DHCP server sits on the right. For each subnet, the server defines all the items in the list. In this case, the configuration reserves the lowest IP addresses in the subnet to be used as static addresses.

The configuration can list other parameters as well. For example, it can set the time limit for leasing an IP address. The server leases an address for a time (usually a number of days), and then the client can ask to renew the lease. If the client does not renew, the server can reclaim the IP address and put it back in the pool of available IP addresses. The server configuration sets the maximum time for the lease.

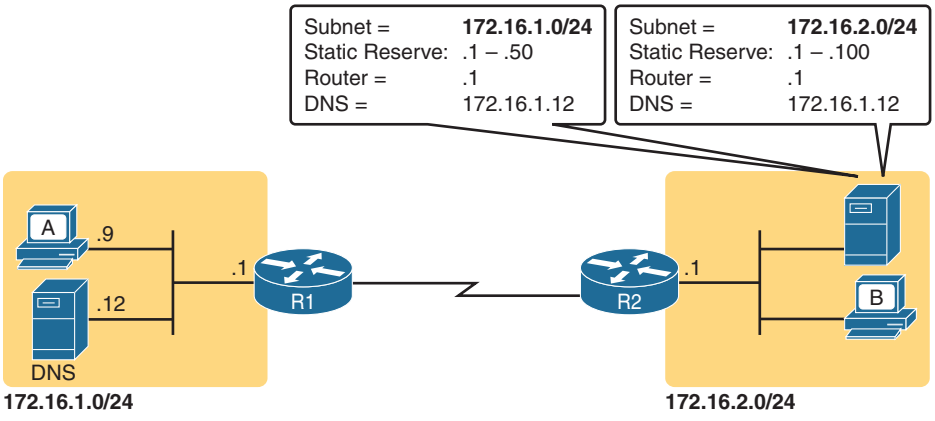


Figure 7-4 *Preconfiguration on a DHCP Server*

DHCP uses three allocation modes, based on small differences in the configuration at the DHCP server. *Dynamic allocation* refers to the DHCP mechanisms and configuration described throughout this chapter. Another method, *automatic allocation*, sets the DHCP lease time to infinite. As a result, once the server chooses an address from the pool and assigns the IP address to a client, the IP address remains with that same client indefinitely. A third mode, *static allocation*, preconfigures the specific IP address for a client based on the client's MAC address. That specific client is the only client that then uses the IP address. (Note that this chapter shows examples and configuration for dynamic allocation only.)

Additionally, the DHCP server can be configured to supply some other useful configuration settings. For instance, a server can supply the IP address of a Trivial File Transfer Protocol (TFTP) server. TFTP servers provide a basic means of storing files that can then be transferred to a client host. As it turns out, Cisco IP phones rely on TFTP to retrieve several configuration files when the phone initializes. DHCP plays a key role by supplying the IP address of the TFTP server that the phones should use.

Configuring DHCP Features on Routers and Switches

Cisco routers and switches support a variety of features. Routers can be configured to act as a DHCP server with just a few straightforward commands—a feature useful in the lab and in some limited cases. More commonly, the enterprise uses a centralized DHCP server (that does not run on a router) but with the router DHCP relay feature on most every router interface. Finally, Cisco routers and switches can also act as DHCP clients, learning their IP addresses from a DHCP server.

This section discusses the DHCP configuration topics mentioned for the current exam topics. Those include the router DHCP relay feature and the configuration to enable DHCP client services on both switches and routers.

NOTE The CCNA 200-301 exam blueprint does not mention the DHCP server function, but many people like to use the IOS DHCP server in the lab for testing with DHCP. If you are interested in how to configure a DHCP server on a router, refer to Appendix D, "Topics from Previous Editions."

Configuring DHCP Relay

Configuring DHCP relay requires a simple decision and a single straightforward configuration command. First, you must identify the interfaces that need the feature. The DHCP relay feature must be configured for any router interface that connects to a subnet where

- DHCP clients exist in the subnet
- DHCP servers do not exist in the subnet

Once such interfaces have been identified, the configuration requires the **ip helper-address** interface subcommand on each of those interfaces. For instance, with earlier Figure 7-3, R1's G0/0 interface needs to be configured with the **ip helper-address 172.16.2.11** interface subcommand. Once enabled on an interface, the IOS DHCP relay agent makes changes in the incoming DHCP messages' addresses as described earlier in the chapter. Without the DHCP relay agent, the DHCP request never arrives at the server.

To verify the relay agent, you can use the **show running-config** command and look for the single configuration command or use the **show ip interface g0/0** command as shown in Example 7-1. The highlighted line confirms the configured setting. Note that if there were no **ip helper-address** commands configured on the interface, the text would instead read "Helper address is not set."

Example 7-1 *Listing the Current Helper Address Setting with* **show ip interface**

```
R1# show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet address is 172.16.1.1/24
  Broadcast address is 255.255.255.255
  Address determined by non-volatile memory
  MTU is 1500 bytes
  Helper address is 172.16.2.11
! Lines omitted for brevity (about 20 lineSc
```

Configuring a Switch as DHCP Client

A switch can act as a DHCP client to lease its IP address. In most cases, you will want to instead use a static IP address so that the staff can more easily identify the switch's address for remote management. However, as an example of how a DHCP client can work, this next topic shows how to configure and verify DHCP client operations on a switch.

NOTE Chapter 6, "Configuring Basic Switch Management," in *CCNA 200-301 Official Cert Guide, Volume 1*, also shows this same example of how to configure a switch to be a DHCP client. This chapter repeats the example here so you can see all the related DHCP configuration details in a single place in this volume.

To configure a switch to use DHCP to lease an address, configure a switch's IP address as normal, but with the **ip address dhcp** interface subcommand. Example 7-2 shows a sample.

Example 7-2 *Switch Dynamic IP Address Configuration with DHCP*

```
Emma# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Emma(config)# interface vlan 1
Emma(config-if)# ip address dhcp
Emma(config-if)# no shutdown
Emma(config-if)# ^Z
Emma#
00:38:20: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
00:38:21: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to up
```

To verify that DHCP worked, start with the traditional way to check IP addresses on switch VLAN interfaces: the **show interfaces vlan** *x* command as demonstrated in Example 7-3. First, check the interface state, because the switch does not attempt DHCP until the VLAN interface reaches an up/up state. Notably, if you forget to issue the **no shutdown** command, the VLAN 1 interface will remain in a shutdown state and listed as "administratively down" in the **show** command output.

Example 7-3 *Verifying DHCP-Learned IP Address on a Switch*

```
Emma# show interfaces vlan 1
Vlan1 is up, line protocol is up
  Hardware is EtherSVI, address is 0019.e86a.6fc0 (bia 0019.e86a.6fc0)
  Internet address is 192.168.1.101/24
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
! lines omitted for brevity
```

The second half of Example 7-3 shows the **show interfaces vlan** *x* command output, which lists the interface's IP address on the third line. If you statically configure the IP address, the IP address will always be listed; however, when using DHCP, this line only exists if DHCP succeeded. Also, note that when present, the output does not state whether the address was statically configured or learned with DHCP. The output lists 192.168.1.101 as the address, but with no information to identify whether the IP address is a static or DHCP-learned IP address.

To see more details specific to DHCP, instead use the **show dhcp lease** command to see the (temporarily) leased IP address and other parameters. (Note that the switch does not store the DHCP-learned IP configuration in the running-config file.) Example 7-4 shows sample output. Note also that the switch learns its default-gateway setting using DHCP as well.

Key Topic

Example 7-4 *Verifying DHCP-Learned Information on a Switch*

```
Emma# show dhcp lease
Temp IP addr: 192.168.1.101   for peer on Interface: Vlan1
Temp sub net mask: 255.255.255.0
   DHCP Lease server: 192.168.1.1, state: 3 Bound
   DHCP transaction id: 1966
   Lease: 86400 secs,  Renewal: 43200 secs,  Rebind: 75600 secs
Temp default-gateway addr: 192.168.1.1
```

```
   Next timer fires after: 11:59:45
   Retry count: 0   Client-ID: cisco-0019.e86a.6fc0-Vl1
   Hostname: Emma

Emma# show ip default-gateway
192.168.1.1
```

Configuring a Router as DHCP Client

Just as with switches, you can configure router interfaces to lease an IP address using DHCP rather than using a static IP address, although those cases will be rare. In most every case it makes more sense to statically configure router interface IP addresses with the address listed in the **ip address** *address mask* interface subcommand. However, configuring a router to lease an address using DHCP makes sense in some cases with a router connected to the Internet; in fact, most every home-based router does just that.

A router with a link to the Internet can learn its IP address and mask with DHCP and also learn the neighboring ISP router's address as the default gateway. Figure 7-5 shows an example, with three routers on the left at one enterprise site. Router R1 uses DHCP to learn its IP address (192.0.2.2) from the ISP router over a connection to the Internet.

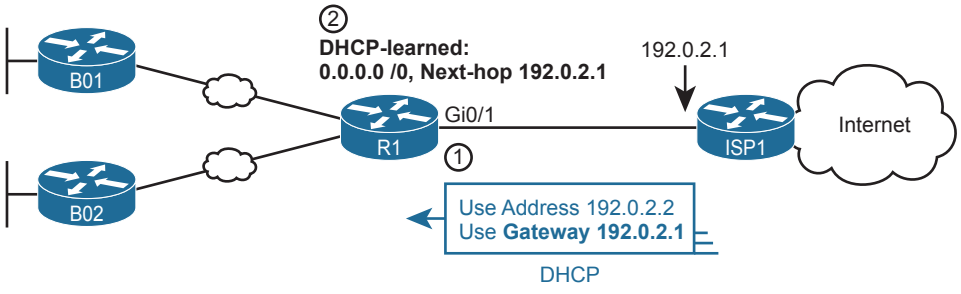


Figure 7-5 *Enterprise Router Building and Advertising Default Routes with DHCP Client*

The DHCP process supplies a default gateway IP address to router R1, but routers do not normally use a default gateway setting; only hosts use a default gateway setting. However, the router takes advantage of that information by turning that default gateway IP address into the basis for a default route. For instance, in Figure 7-5, router R1 dynamically adds a default route to its routing table with the default gateway IP address from the DHCP message—which is the ISP router's IP address—as the next-hop address. At that point, R1 has a good route to use to forward packets into the Internet.

Additionally, router R1 can distribute that default route to the rest of the routers using an interior routing protocol like OSPF. See the section titled "OSPF Default Routes" in Chapter 20 of the *CCNA 200-301 Official Cert Guide, Volume 1*, for more information.

Example 7-5 shows the configuration on router R1 to match Figure 7-5. Note that it begins with R1 configuring its G0/1 interface to use DHCP to learn the IP address to use on the interface, using the **ip address dhcp** command.

Example 7-5 *Learning an Address and Default Static Route with DHCP*

```
R1# configure terminal
R1(config)# interface gigabitethernet0/1
R1(config-if)# ip address dhcp
R1(config-if)# end
R1#
R1# show ip route static
! Legend omitted
Gateway of last resort is 192.0.2.1 to network 0.0.0.0

S*     0.0.0.0/0 [254/0] via 192.0.2.1
```

The end of the example shows the default route added to R1's routing table as a result of learning a default gateway address of 192.0.2.1 from DHCP. Oddly, IOS displays this route as a static route (destination 0.0.0.0/0), although the route is learned dynamically based on the DHCP-learned default gateway. To recognize this route as a DHCP-learned default route, look to the administrative distance value of 254. IOS uses a default administrative distance of 1 for static routes configured with the **ip route** configuration command but a default of 254 for default routes added because of DHCP.

Identifying Host IPv4 Settings

Whether learned using DHCP or not, every host that uses IP version 4 needs to have some settings to work correctly. This second major division of the chapter examines those settings and shows examples of those settings on Windows, Linux, and macOS.

Host Settings for IPv4

To work correctly, an IPv4 host needs to know these values:

Key Topic

- DNS server IP addresses
- Default gateway (router) IP address
- Device's own IP address
- Device's own subnet mask

To review the basics, the host must know the IP address of one or more DNS servers to send the servers' name resolution requests. For enterprises, the servers may reside in the enterprise, as shown in Figure 7-6. The host on the left (sometimes called an endpoint) typically knows the addresses of at least two DNS servers for redundancy. If the first DNS fails to respond, the endpoint can then attempt name resolution with the next DNS server.

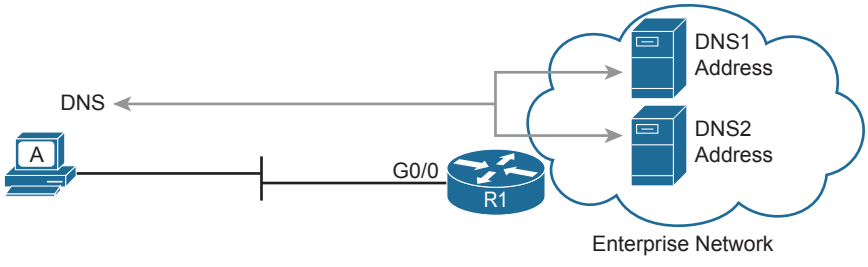


Figure 7-6 *Host A Needs to Know the IP Address of the DNS Servers*

Each endpoint needs to know the IP address of a router that resides in the same subnet. The endpoint uses that router as its default router or default gateway, as shown in Figure 7-7. From a host logic perspective, the host can then forward packets destined for addresses outside the subnet to the default router, with that router then forwarding the packet based on its routing table.

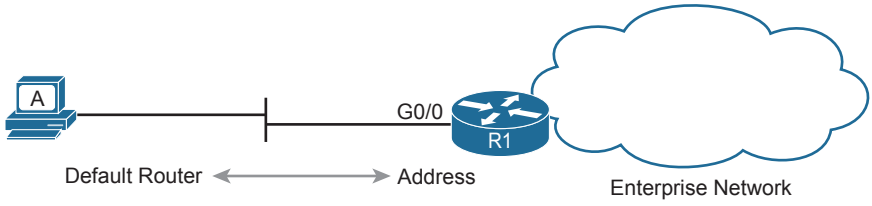


Figure 7-7 *Host Default Router Setting Should Equal Router Interface Address*

Of course, each device needs its own IP address and subnet mask. Equally as important, note that the host and the default router need to agree as to the addresses inside the subnet. The host will use the address and mask to do the math to determine which addresses are in the same subnet and which are in other subnets. For routing to work correctly, the default router's interface address and mask should result in the same definition of the subnet with the same addresses, as shown in Figure 7-8.

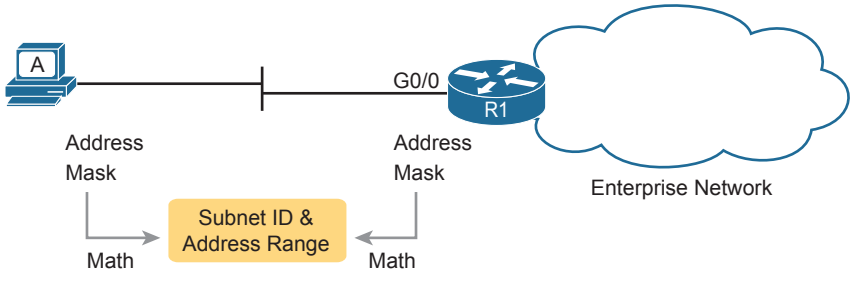


Figure 7-8 *The Need for Subnet Agreement Between Host and Default Router*

The rest of this section shows examples of the display of these settings in the graphical user interface (GUI) and command-line interface (CLI) of three different host operating systems.

Host IP Settings on Windows

Most every OS in the world—certainly the more common OSs people work with every day—have a fairly easy-to-reach settings window that lists most if not all the IPv4 settings in one place. For example, Figure 7-9 shows the Network configuration screen from a Windows 10 host from the network area of the Windows Control Panel. This particular example shows the big four settings: address, mask, router, and DNS.

However, beyond the GUI, most OSs have a variety of networking commands available from a command line. With all Windows versions, the **ipconfig** and **ipconfig /all** commands supply the most direct help, as shown in Example 7-6. As you can see, both list the address, mask, and default gateway, with the **ipconfig /all** command also listing the DNS server settings.

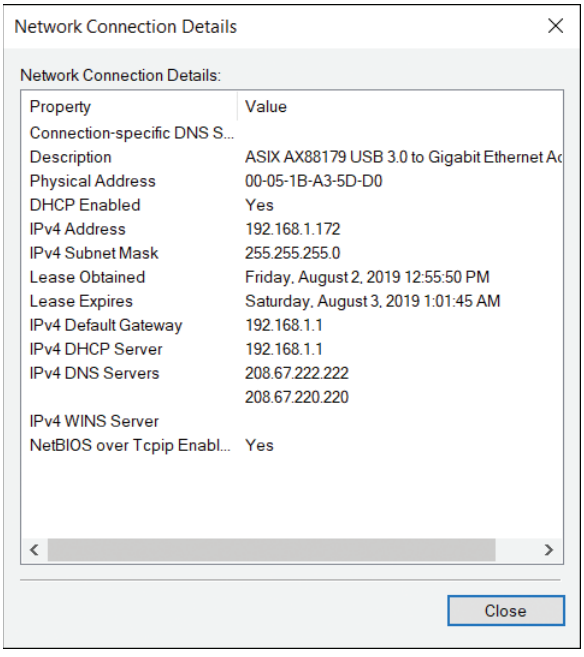


Figure 7-9 *IP Address, Mask, and Default Router Settings on Windows*

Key Topic

Example 7-6 ipconfig *and* ipconfig /all *(Windows)*

```
C:\DOCUME1\OWNER> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet3:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.172
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

C:\DOCUME1\OWNER> ipconfig /all
! Lines omitted for brevity
Ethernet adapter Ethernet 3:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : ASIX AX88179 USB 3.0 to Gigabit Ethernet
Adapter
   Physical Address. . . . . . . . . : 00-05-1B-A3-5D-D0
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.1.172(Preferred)
```

```
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Friday, August 2, 2019 12:55:50 PM
   Lease Expires . . . . . . . . . . : Saturday, August 3, 2019 1:01:45 AM
   Default Gateway . . . . . . . . . : 192.168.1.1
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DNS Servers . . . . . . . . . . . : 208.67.222.222
                                       208.67.220.220
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Another common command on most user host OSs is the **netstat -rn** command. This command lists the host's IP routing table. Of interest, the top of the table lists a route based on the default gateway, with the destination subnet and mask listed as 0.0.0.0 and 0.0.0.0. The top of the output also lists several other routes related to having a working interface, like a route to the subnet connected to the interface. Example 7-7 lists an excerpt from the **netstat -rn** command from the same Windows host, with the default route and the route to the local subnet (192.168.1.0) listed. Note that a gateway of "on-link" means that the PC thinks the destination is on the local subnet (link).

Example 7-7 **netstat -rn** *Command (Windows)*

```
C:\DOCUME1\OWNER> netstat -rn

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.1.1    192.168.1.172     25
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      169.254.0.0      255.255.0.0         On-link   169.254.244.178    291
  169.254.244.178  255.255.255.255         On-link   169.254.244.178    291
  169.254.255.255  255.255.255.255         On-link   169.254.244.178    291
      192.168.1.0    255.255.255.0         On-link     192.168.1.172    281
    192.168.1.172  255.255.255.255         On-link     192.168.1.172    281
    192.168.1.255  255.255.255.255         On-link     192.168.1.172    281
! Lines omitted for brevity
```

Host IP Settings on macOS

Although the particulars vary, like Windows, macOS has both a graphical interface to see network settings and a variety of network commands. This section shows examples of each, beginning with Figure 7-10. It shows the network settings in macOS for an Ethernet interface, with the address, mask, default router, and DNS server addresses. Also note the setting states that the interface is using DHCP.

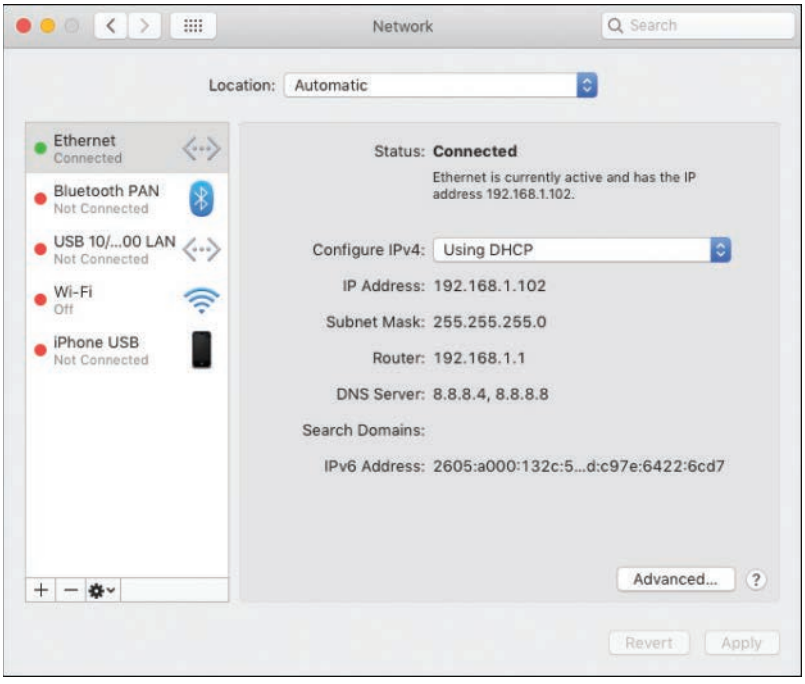


Figure 7-10 *IP Address, Mask, and Default Router Settings on macOS*

macOS and Linux both support the **ifconfig** command to list information similar to the Windows **ipconfig /all** command. (Note that **ifconfig** does not have an **/all** option.) Of note, the **ifconfig** command does not list the default gateway or DNS servers, so Example 7-8 includes two other macOS commands that supply those details.

Key Topic

Example 7-8 **ifconfig, networksetup -getinfo,** *and* **networksetup -getdnsservers** *(macOS)*

```
Wendell-Odoms-iMac:~ wendellodom$ ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=10b<RXCSUM,TXCSUM,VLAN_HWTAGGING,AV>
        ether 0c:4d:e9:a9:9c:41
        inet 192.168.1.102 netmask 0xffffff00 broadcast 192.168.1.255
! IPv6 details omitted for brevity
        media: autoselect (1000baseT <full-duplex,flow-control,energy-efficient-
ethernet>)
        status: active

Wendell-Odoms-iMac:~ wendellodom$ networksetup -getinfo Ethernet
DHCP Configuration
IP address: 192.168.1.102
Subnet mask: 255.255.255.0
Router: 192.168.1.1
Client ID:
IPv6: Automatic
```

```
IPv6 IP address: none
IPv6 Router: none
Ethernet Address: 0c:4d:e9:a9:9c:41

Wendell-Odoms-iMac:~ wendellodom$ networksetup -getdnsservers Ethernet
8.8.8.4
8.8.8.8
```

Like Windows, macOS adds a default route to its host routing table based on the default gateway, as well as a route to the local subnet calculated based on the IP address and mask learned with DHCP. And like Windows, macOS uses the **netstat -rn** command to list those routes—but with several differences in the output. Of note in the macOS sample shown in Example 7-9, the output represents the default route using the word *default* rather than the paired numbers 0.0.0.0 and 0.0.0.0 for the destination subnet and mask.

Example 7-9 **netstat -rn** *Command (macOS)*

```
C:\DOCUME1\OWNER> netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags        Refs      Use   Netif Expire
default            192.168.1.1        UGSc           92        0     en0
127                127.0.0.1          UCS             0        0     lo0
127.0.0.1          127.0.0.1          UH              4     1950     lo0
169.254            link#5             UCS             2        0     en0      !
169.254.210.104    0:5:1b:a3:5d:d0    UHLSW           0        0     en0      !
192.168.1          link#5             UCS             9        0     en0      !
192.168.1.1/32     link#5             UCS             1        0     en0      !
192.168.1.1        60:e3:27:fb:70:97  UHLWIir        12     2502     en0   1140
192.168.1.102/32   link#5             UCS             0        0     en0      !
! lines omitted for brevity
```

Host IP Settings on Linux

On Linux, the graphical windows to display network settings differ for many reasons. First, the Linux world includes a large number of different Linux versions or distributions. Additionally, Linux separates the OS from the desktop (the graphical interface) so that a user of one Linux distribution can choose between different desktop interfaces. As a result, you will see different GUI screens to display the Linux network settings.

For perspective, this section shows a few examples from the MATE desktop included in the Ubuntu MATE Linux distribution (www.ubuntu-mate.org). First, the image in Figure 7-11 shows details for a wireless LAN adapter and includes the IPv4 address, mask, default router, and primary DNS IP address.

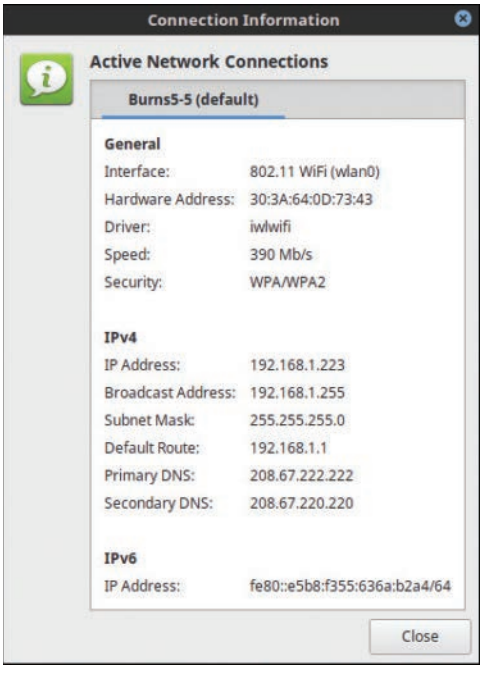


Figure 7-11 *IP Address, Mask, and Default Router Settings on Linux*

From the command line, Linux hosts will often support a large set of commands. However, an older set of commands, referenced together as *net-tools*, has been deprecated in Linux, to the point that some Linux distributions do not include net-tools. (You can easily add net-tools to most Linux distributions.) The net-tools library includes **ifconfig** and **netstat -rn**. To replace those tools, Linux uses the iproute library, which includes a set of replacement commands and functions, many performed with the **ip** command and some parameters.

NOTE Check out this link for a broader comparison of the commands: https://access.redhat.com/sites/default/files/attachments/rh_ip_command_cheatsheet_1214_jcs_print.pdf.

Example 7-10 shows a sample of the **ifconfig** command for the same interface detailed in Figure 7-11. Note that it lists the Ethernet MAC and IPv4 addresses, along with the subnet mask, similar to the macOS version of the command. However, on Linux, it also shows some interface counters.

Example 7-10 **ifconfig** *and* **ip address** *Commands (Linux)*

```
chris@LL ~ $ ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 30:3a:64:0d:73:43
          inet addr:192.168.1.223  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::e5b8:f355:636a:b2a4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2041153 errors:0 dropped:0 overruns:0 frame:0
          TX packets:712814 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```
          RX bytes:2677874115 (2.6 GB)  TX bytes:134076542 (134.0 MB)

chris@LL ~ $ ip address
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default
qlen 1000
    link/ether 30:3a:64:0d:73:43 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.223/24 brd 192.168.1.255 scope global wlan0
       valid_lft forever preferred_lft forever
    inet6 fe80::e5b8:f355:636a:b2a4/64 scope link
       valid_lft forever preferred_lft forever
```

The bottom of the example shows the command from the iproute package that replaces **ifconfig**, namely the **ip address**. Note that it shows the same basic addressing information, just with the subnet mask shown in prefix notation rather than in dotted decimal.

Linux has long supported the **netstat -rn** command as well, as part of the net-tools package, with a sample shown in Example 7-11. The output lists a default route, but with a style that shows the destination as 0.0.0.0. As usual, the default route points to the default gateway as learned with DHCP: 192.168.1.1. It also lists a route to the local subnet (192.168.1.0 as highlighted toward the bottom of the output).

Example 7-11 **netstat -rn** *and* **ip route** *Commands (Linux)*

```
chris@LL ~ $ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG        0 0          0 wlan0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0          0 wlan0
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 wlan0

chris@LL ~ $ ip route
default via 192.168.1.1 dev wlan0  proto static  metric 600
169.254.0.0/16 dev wlan0  scope link  metric 1000
192.168.1.0/24 dev wlan0  proto kernel  scope link  src 192.168.1.223  metric 600
chris@LL ~ $
```

The bottom of the example shows the command meant to replace **netstat -rn: ip route**. Note that it also shows a default route that references the default router, along with a route for the local subnet.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 7-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 7-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review command tables		Book

Review All the Key Topics

Key Topic

Table 7-3 Key Topics for Chapter 7

Key Topic Element	Description	Page Number
List	Definitions of special IPv4 addresses 0.0.0.0 and 255.255.255.255	125
List	Four logic steps created by the **ip helper-address** command	127
Figure 7-2	What the **ip helper-address** command changes in a DHCP Discover message	127
List	The two facts that must be true about a subnet for a router to need to be a DHCP relay agent for that subnet	130
Example 7-4	Switch commands that confirm the details of DHCP client operations based on the **ip address dhcp** interface subcommand	131
List	The IPv4 settings expected on an end-user host	133
Example 7-6	Output from a Windows **ipconfig /all** command	135
Example 7-8	Output from a macOS **ifconfig** command plus two **networksetup** commands	137

7

Key Terms You Should Know

DHCP client, DHCP server, DHCP relay agent, default gateway, DNS server

Command References

Tables 7-4, 7-5, and 7-6 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 7-4 Chapter 7 Configuration Command Reference

Command	Description
ip helper-address *IP-address*	An interface subcommand that tells the router to notice local subnet broadcasts (to 255.255.255.255) that use UDP, and change the source and destination IP address, enabling DHCP servers to sit on a remote subnet
ip address dhcp	An interface subcommand that tells the router or switch to use DHCP to attempt to lease a DHCP address from a DHCP server

Table 7-5 Chapter 7 EXEC Command Reference

Command	Description
show arp, show ip arp	Command that lists the router's IPv4 ARP table
show dhcp lease	Switch command that lists information about addresses leased because of the configuration of the **ip address dhcp** command
show ip default-gateway	Switch command that lists the switch's default gateway setting, no matter whether learned by DHCP or statically configured

Table 7-6 Chapter 7 Generic Host Networking Command Reference

Command	Description
ipconfig /all	(Windows) Lists IP address, mask, gateway, and DNS servers
ifconfig	(Mac, Linux) Lists IP address and mask for an interface
networksetup -getinfo *interface*	(Mac) Lists IP settings including default router
networksetup -getdnsservers *interface*	(Mac) Lists DNS servers used
netstat -rn	(Windows, Mac, Linux) Lists the host's routing table, including a default route that uses the DHCP-learned default gateway
arp -a	(Windows, Mac, Linux) Lists the host's ARP table
ip address	(Linux) Lists IP address and mask information for interfaces; the Linux replacement for **ifconfig**
ip route	(Linux) Lists routes, including the default route and a route to the local subnet; the Linux replacement for **netstat -rn**

This page intentionally left blank

CHAPTER 8

DHCP Snooping and ARP Inspection

This chapter covers the following exam topics:

5.0 Security Fundamentals

5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)

To understand the kinds of risks that exist in modern networks, you have to first understand the rules. Then you have to think about how an attacker might take advantage of those rules in different ways. Some attacks might cause harm as part of a denial-of-service (DoS) attack, while a reconnaissance attack may gather more data to prepare for some other attack. For every protocol and function you learn in networking, there are possible methods to take advantage of those features to give an attacker an advantage.

This chapter discusses two switch features that help prevent some types of attacks that can result in the attacker getting copies of packets sent to/from a legitimate host. One of these features, DHCP Snooping, notices DHCP messages that fall outside the normal use of DHCP—messages that may be part of an attack—and discards those messages. It also watches the DHCP messages that flow through a LAN switch, building a table that lists the details of legitimate DHCP flows, so that other switch features can know what legitimate DHCP leases exist for devices connected to the switch.

The second such feature, Dynamic ARP Inspection (DAI), also helps prevent packets being redirected to an attacking host. Some ARP attacks try to convince hosts to send packets to the attacker's device instead of the true destination. The switch watches ARP messages as they flow through the switch. The switch checks incoming ARP messages, checking those against normal ARP operation as well as checking the details against other data sources, including the DHCP Snooping binding table. When the ARP message does not match the known information about the legitimate addresses in the network, the switch filters the ARP message.

This chapter examines DHCP Snooping concepts and configuration in the first major section and DAI in the second.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 8-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
DHCP Snooping	1–4
Dynamic ARP Inspection	5–7

1. An engineer hears about DHCP Snooping and decides to implement it. Which of the following are the devices on which DHCP Snooping could be implemented? (Choose two answers.)
   a. Layer 2 switches
   b. Routers
   c. Multilayer switches
   d. End-user hosts
2. Layer 2 switch SW2 connects a Layer 2 switch (SW1), a router (R1), a DHCP server (S1), and three PCs (PC1, PC2, and PC3). All PCs are DHCP clients. Which of the following are the most likely DHCP Snooping trust state configurations on SW2 for the ports connected to the listed devices? (Choose two answers.)
   a. The port connected to the router is untrusted.
   b. The port connected to switch SW1 is trusted.
   c. The port connected to PC1 is untrusted.
   d. The port connected to PC3 is trusted.
3. Switch SW1 needs to be configured to use DHCP Snooping in VLAN 5 and only VLAN 5. Which commands must be included, assuming at least one switch port in VLAN 5 must be an untrusted port? (Choose two answers.)
   a. **no ip dhcp snooping trust**
   b. **ip dhcp snooping untrust**
   c. **ip dhcp snooping**
   d. **ip dhcp snooping vlan 5**
4. On a multilayer switch, a switch needs to be configured to perform DHCP Snooping on some Layer 2 ports in VLAN 3. Which command may or may not be needed depending on whether the switch also acts as a DHCP relay agent?
   a. **no ip dhcp snooping information option**
   b. **ip dhcp snooping limit rate 5**
   c. **errdisable recovery cause dhcp-rate-limit**
   d. **ip dhcp snooping vlan 3**

5. Switch SW1 has been configured to use Dynamic ARP Inspection with DHCP Snooping in VLAN 5. An ARP request arrives on port G0/1. Which answer describes two items DAI always compares regardless of the configuration?
   a. The message's ARP origin hardware address and the message's Ethernet header source MAC address
   b. The message's ARP origin hardware address and the DHCP Snooping binding table
   c. The message's ARP target IP address and the DHCP Snooping binding table
   d. The message's ARP target IP address and the switch's ARP table
6. Switch SW1 needs to be configured to use Dynamic ARP Inspection along with DHCP Snooping in VLAN 6 and only VLAN 6. Which commands must be included, assuming at least one switch port in VLAN 6 must be a trusted port? (Choose two answers.)
   a. **no ip arp inspection untrust**
   b. **ip arp inspection trust**
   c. **ip arp inspection**
   d. **ip arp inspection vlan 6**
7. A Layer 2 switch needs to be configured to use Dynamic ARP Inspection along with DHCP Snooping. Which command would make DAI monitor ARP message rates on an interface at an average rate of 4 received ARP messages per second? (Choose two answers.)
   a. **ip arp inspection limit rate 4 burst interval 2**
   b. **ip arp inspection limit rate 10 burst interval 2**
   c. **ip arp inspection limit rate 16 burst interval 4**
   d. **ip arp inspection limit rate 4**

Foundation Topics

DHCP Snooping

DHCP servers play a vital role in most every network today, with almost every user endpoint using DHCP to learn its IP address, mask, default gateway, and DNS server IP addresses. Chapter 7, "Implementing DHCP," shows how DHCP should work under normal circumstances. This section now examines how attackers might use DHCP for their own ends and how two specific tools—DHCP Snooping and Dynamic ARP Inspection (DAI)—help defeat those attacks.

This section begins with an examination of the need for DHCP Snooping concepts including the types of attacks it can try to prevent, followed by details of how to configure DHCP Snooping.

DHCP Snooping Concepts

DHCP Snooping on a switch acts like a firewall or an ACL in many ways. It analyzes incoming messages on the specified subset of ports in a VLAN. DHCP Snooping never filters

non-DHCP messages, but it may choose to filter DHCP messages, applying logic to make a choice—allow the incoming DHCP message or discard the message.

While DHCP itself provides a Layer 3 service, DHCP Snooping operates on LAN switches and is commonly used on Layer 2 LAN switches and enabled on Layer 2 ports. The reason to put DHCP Snooping on the switch is that the function needs to be performed between a typical end-user device—the type of device that acts as a DHCP client—and DHCP servers or DHCP relay agents.

Figure 8-1 shows a sample network that provides a good backdrop to discuss DHCP Snooping. First, all devices connect to Layer 2 switch SW2, with all ports as Layer 2 switchports, all in the same VLAN. The typical DHCP clients sit on the right of the figure. The left shows other devices that could be the path through which to reach a DHCP server.

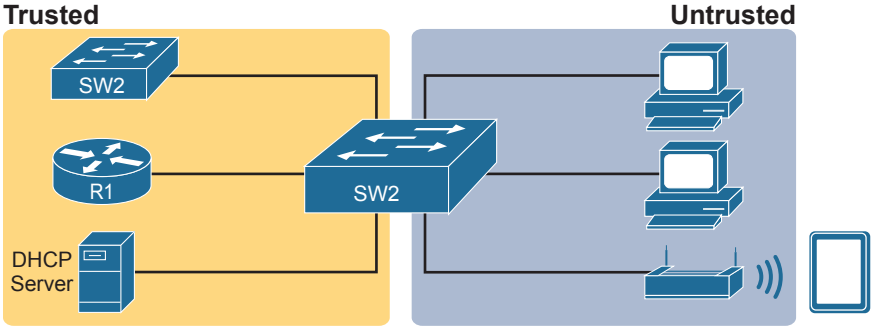


Figure 8-1 *DHCP Snooping Basics: Client Ports Are Untrusted*

DHCP Snooping works first on all ports in a VLAN, but with each port being trusted or untrusted by DHCP Snooping. To understand why, consider this summary of the general rules used by DHCP Snooping. Note that the rules differentiate between messages normally sent by servers (like DHCPOFFER and DHCPACK) versus those normally sent by DHCP clients:

- DHCP messages received on an untrusted port, for messages normally sent by a server, will always be discarded.
- DHCP messages received on an untrusted port, as normally sent by a DHCP client, may be filtered if they appear to be part of an attack.
- DHCP messages received on a trusted port will be forwarded; trusted ports do not filter (discard) any DHCP messages.

A Sample Attack: A Spurious DHCP Server

To give you some perspective, Figure 8-2 shows a legitimate user's PC on the far right and the legitimate DHCP server on the far left. However, an attacker has connected his laptop to the LAN and started his DHCP attack by acting like a DHCP server. Following the steps in the figure, assume PC1 is attempting to lease an IP address while the attacker is making his attack:

1. PC1 sends a LAN broadcast with PC1's first DHCP message (DHCPDISCOVER).
2. The attacker's PC—acting as a spurious DHCP server—replies to the DHCPDISCOVER with a DHCPOFFER.

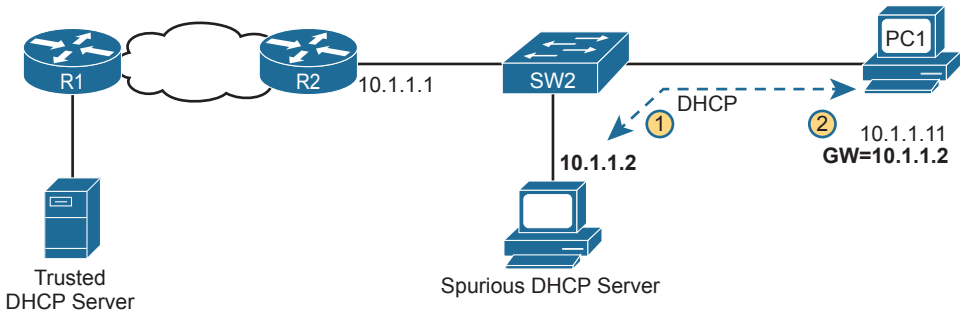


Figure 8-2 *DHCP Attack Supplies Good IP Address but Wrong Default Gateway*

In this example, the DHCP server created and used by the attacker actually leases a useful IP address to PC1, in the correct subnet, with the correct mask. Why? The attacker wants PC1 to function, but with one twist. Notice the default gateway assigned to PC1: 10.1.1.2, which is the attacker's PC address, rather than 10.1.1.1, which is router R1's address. Now PC1 thinks it has all it needs to connect to the network, and it does—but now all the packets sent by PC1 to what it thinks is its default router flow first through the attacker's PC, creating a man-in-the-middle attack, as shown in Figure 8-3.

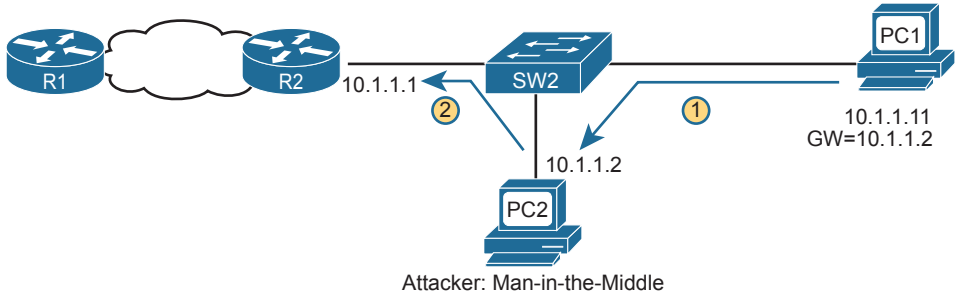


Figure 8-3 *Unfortunate Result: DHCP Attack Leads to Man-in-the-Middle*

Note that the legitimate DHCP also returns a DHCPOFFER message to host PC1, but most hosts use the first received DHCPOFFER, and the attacker will likely be first in this scenario.

The two steps in the figure show data flow once DHCP has completed. For any traffic destined to leave the subnet, PC1 sends its packets to its default gateway, 10.1.1.2, which happens to be the attacker. The attacker forwards the packets to R1. The PC1 user can connect to any and all applications just like normal, but now the attacker can keep a copy of anything sent by PC1.

DHCP Snooping Logic

The preceding example shows just one attack in which the attacker acts like a DHCP server (spurious DHCP server). DHCP Snooping defeats such attacks by making most ports

---

Answers to the "Do I Know This Already?" quiz:

1 A, C **2** B, C **3** C, D **4** A **5** B **6** B, D **7** C, D

untrusted, which by definition would filter the DHCP server messages that arrive on the untrusted ports. For instance, in Figures 8-2 and 8-3, making the port connected to the attacker, a DHCP Snooping untrusted port defeats the attack.

To appreciate the broader set of DHCP Snooping rules and logic, it helps to have a handy reference of some of the more common DHCP messages and processes. For a quick review, the normal message flow includes this sequence: DISCOVER, OFFER, REQUEST, ACK (DORA). In particular:

- Clients send DISCOVER and REQUEST.
- Servers send OFFER and ACK.

Additionally, DHCP clients also use the DHCP RELEASE and DHCP DECLINE messages. When a client has a working lease for an address but no longer wants to use the address, the DHCP client can tell the DHCP server it no longer needs the address, releasing it back to the DHCP server, with the DHCP RELEASE message. Similarly, a client can send a DHCP DECLINE message to turn down the use of an IP address during the normal DORA flow on messages.

Now to the logic for DHCP Snooping untrusted ports. Figure 8-4 summarizes the ideas, with two switch ports. On the left, the switch port connects to a DHCP server, so it should be trusted; otherwise DHCP would not work, because the switch would filter all DHCP messages sent by the DHCP server. On the right, PC1 connects to an untrusted port with a DHCP client.

Key Topic

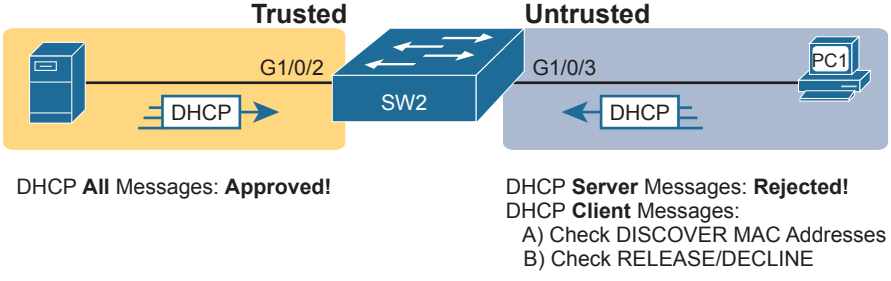


Figure 8-4 *Summary of Rules for DHCP Snooping*

The following list summarizes the DHCP Snooping rules:

Key Topic

1. Examine all incoming DHCP messages.
2. If normally sent by servers, discard the message.
3. If normally sent by clients, filter as follows:
   a. For DISCOVER and REQUEST messages, check for MAC address consistency between the Ethernet frame and the DHCP message.
   b. For RELEASE or DECLINE messages, check the incoming interface plus IP address versus the DHCP Snooping binding table.
4. For messages not filtered that result in a DHCP lease, build a new entry to the DHCP Snooping binding table.

The next few pages complete the discussion of concepts by explaining a little more about steps 3 and 4 in the list.

Filtering DISCOVER Messages Based on MAC Address

DHCP Snooping does one straightforward check for the most common client-sent messages: DISCOVER and REQUEST. First, note that DHCP messages define the chaddr (client hardware address) field to identify the client. Hosts on LANs include the device's MAC address as part of chaddr. As usual, Ethernet hosts encapsulate the DHCP messages inside Ethernet frames, and those frames of course include a source MAC address—an address that should be the same MAC address used in the DHCP chaddr field. DHCP Snooping does a simple check to make sure those values match.

Figure 8-5 shows how an attacker could attempt to overload the DHCP server and lease all the addresses in the subnet. The attacker's PC uses pseudo MAC address A, so all three DISCOVER messages in the figure show a source Ethernet address of "A." However, each message (in the DHCP data) identifies a different MAC address in the chaddr value (shown as MAC1, MAC2, and MAC3 in the figure for brevity), so from a DHCP perspective, each message appears to be a different DHCP request. The attacker can attempt to lease every IP address in the subnet so that no other hosts could obtain a lease.

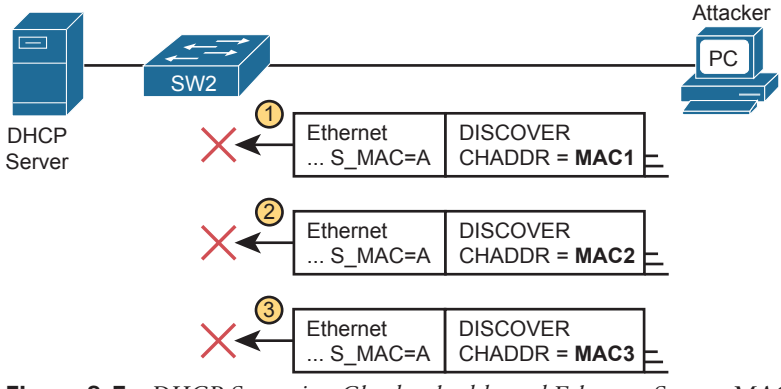


Figure 8-5 *DHCP Snooping Checks chaddr and Ethernet Source MAC*

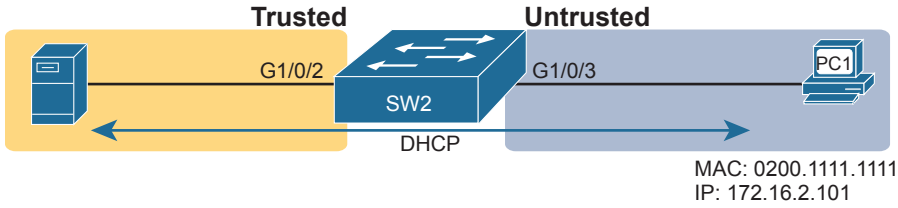
The core feature of DHCP Snooping defeats this type of attack on untrusted ports. It checks the Ethernet header source MAC address and compares that address to the MAC address in the DHCP header, and if the values do not match, DHCP Snooping discards the message.

Filtering Messages that Release IP Addresses

Before looking at the next bit of logic, you need to first understand the DHCP Snooping binding table.

DHCP Snooping builds the DHCP Snooping binding table for all the DHCP flows it sees that it allows to complete. That is, for any working legitimate DHCP flows, it keeps a list of some of the important facts. Then DHCP Snooping, and other features like Dynamic ARP Inspection, can use the table to make decisions.

As an example, consider Figure 8-6, which repeats the same topology as Figure 8-4, now with one entry in its DHCP Snooping binding table.



SW2 DHCP Snooping Binding Table

MAC	IP	VLAN	Interface
0200.1111.1111	172.16.2.101	11	G1/0/3

Figure 8-6 *Legitimate DHCP Client with DHCP Binding Entry Built by DHCP Snooping*

In this simple network, the DHCP client on the right leases IP address 172.16.2.101 from the DHCP server on the left. The switch's DHCP Snooping feature combines the information from the DHCP messages, with information about the port (interface G1/0/3, assigned to VLAN 11 by the switch), and puts that in the DHCP Snooping binding table.

DHCP Snooping then applies additional filtering logic that uses the DHCP Snooping binding table: it checks client-sent messages like RELEASE and DECLINE that would cause the DHCP server to be allowed to release an address. For instance, a legitimate user might lease address 172.16.2.101, and at some point release the address back to the server; however, before the client has finished with its lease, an attacker could send DHCP RELEASE message to release that address back into the pool. The attacker could then immediately try to lease that address, hoping the DHCP server assigns that same 172.16.2.101 address to the attacker.

Figure 8-7 shows an example. PC1 already has a DHCP address (172.16.2.101), with SW2 listing an entry in the DHCP Snooping binding table. The figure shows the action by which the attacker off port G1/0/5 attempts to release PC1's address. DHCP Snooping compares the incoming message, incoming interface, and matching table entry:

1. The incoming message is a DHCP RELEASE message in port G1/0/5 listing address 172.16.2.101.
2. The DHCP Snooping binding table lists 172.16.2.101 as being originally leased via messages arriving on port G1/0/3.
3. DHCP Snooping discards the DHCP RELEASE message.

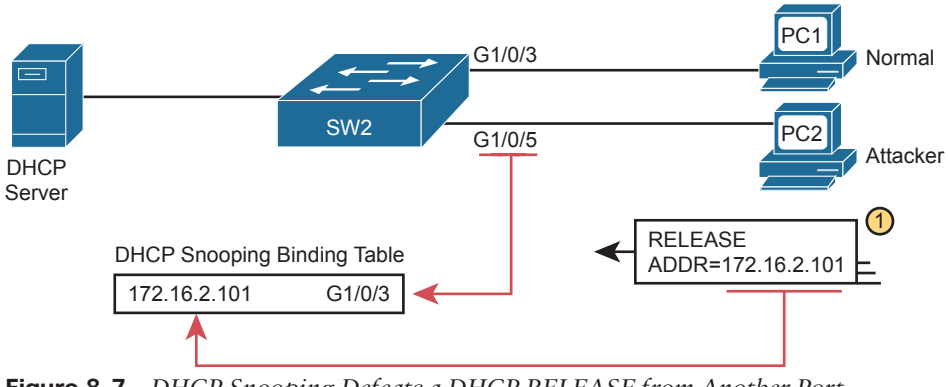


Figure 8-7 *DHCP Snooping Defeats a DHCP RELEASE from Another Port*

DHCP Snooping Configuration

DHCP Snooping requires several configuration steps to make it work. First, you need to use a pair of associated global commands: one to enable DHCP Snooping and another to list the VLANs on which to use DHCP Snooping. Both must be included for DHCP Snooping to operate.

Second, while not literally required, you will often need to configure a few ports as trusted ports. Most switches that use DHCP Snooping for a VLAN have some trusted ports and some untrusted ports, and with a default of untrusted, you need to configure the trusted ports.

This section begins with an example that shows how to configure a typical Layer 2 switch to use DHCP Snooping, with required commands as just described, and with other optional commands.

Configuring DHCP Snooping on a Layer 2 Switch

The upcoming examples all rely on the topology illustrated in Figure 8-8, with Layer 2 switch SW2 as the switch on which to enable DHCP Snooping. The DHCP server sits on the other side of the WAN, on the left of the figure. As a result, SW2's port connected to router R2 (a DHCP relay agent) needs to be trusted. On the right, two sample PCs can use the default untrusted setting.

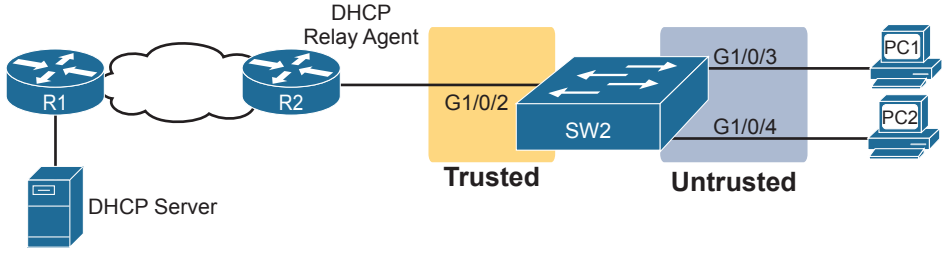


Figure 8-8 *Sample Network Used in DHCP Snooping Configuration Examples*

Switch SW2 places all the ports in the figure in VLAN 11, so to enable DHCP Snooping in VLAN 11, SW2 requires two commands, as shown near the top of Example 8-1: **ip dhcp snooping** and **ip dhcp snooping vlan 11**. Then, to change the logic on port G1/0/2 (connected to the router) to be trusted, the configuration includes the **ip dhcp snooping trust** interface subcommand.

Key Topic

Example 8-1 *DHCP Snooping Configuration to Match Figure 8-8*

```
ip dhcp snooping
ip dhcp snooping vlan 11
no ip dhcp snooping information option
!
interface GigabitEthernet1/0/2
 ip dhcp snooping trust
```

Note that the **no ip dhcp snooping information option** command in Example 8-1 will be explained in a better context just after Example 8-2 but is listed in Example 8-1 to make the example complete.

With this configuration, the switch follows the logic steps detailed in the earlier section titled "DHCP Snooping Logic." To see some support for that claim, look at Example 8-2, which shows the output from the **show ip dhcp snooping** command on switch SW2.

Example 8-2 *SW2 DHCP Snooping Status*

```
SW2# show ip dhcp snooping
Switch DHCP snooping is enabled
Switch DHCP gleaning is disabled
DHCP snooping is configured on following VLANs:
11
DHCP snooping is operational on following VLANs:
11
Smartlog is configured on following VLANs:
none
Smartlog is operational on following VLANs:
none
DHCP snooping is configured on the following L3 Interfaces:

Insertion of option 82 is disabled
   circuit-id default format: vlan-mod-port
   remote-id: bcc4.938b.a180 (MAC)
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Verification of giaddr field is enabled
DHCP snooping trust/rate is configured on the following Interfaces:

Interface                    Trusted    Allow option    Rate limit (pps)
-----------------------      -------    ------------    ----------------
GigabitEthernet1/0/2         yes        yes             unlimited
  Custom circuit-ids:
```

The highlighted lines in the example point out a few of the key configuration settings. Starting at the top, the first two confirm the configuration of the **ip dhcp snooping** and **ip dhcp snooping vlan 11** commands, respectively. Also, the highlighted lines at the bottom of the output show a section that lists trusted ports—in this case, only port G1/0/2.

Also, you might have noticed that highlighted line in the middle that states **Insertion of option 82 is disabled**. That line confirms the addition of the **no ip dhcp information option** command to the configuration back in Example 8-1. To understand why the example includes this command, consider these facts about DHCP relay agents:

- DHCP relay agents add new fields to DHCP requests—defined as option 82 DHCP header fields (in RFC 3046).
- DHCP Snooping uses default settings that work well if the switch acts as a Layer 3 switch and as a DHCP relay agent, meaning that the switch should insert the DHCP option 82 fields into DHCP messages. In effect, the switch defaults to use **ip dhcp snooping information option**.

- When the switch does not also act as a DHCP relay agent, the default setting stops DHCP from working for end users. The switch sets fields in the DHCP messages as if it were a DHCP relay agent, but the changes to those messages cause most DHCP servers (and most DHCP relay agents) to ignore the received DHCP messages.
- The conclusion: To make DHCP Snooping work on a switch that is not also a DHCP relay agent, disable the option 82 feature using the **no ip dhcp snooping information option** global command.

That concludes the DHCP Snooping configuration that is both required and that you will most often need to make the feature work. The rest of this section discusses a few optional DHCP Snooping features.

Limiting DHCP Message Rates

Knowing that DHCP Snooping prevents their attacks, what might attackers do in response? Devise new attacks, including attacking DHCP Snooping itself.

One way to attack DHCP Snooping takes advantage of the fact that it uses the general-purpose CPU in a switch. Knowing that, attackers can devise attacks to generate large volumes of DHCP messages in an attempt to overload the DHCP Snooping feature and the switch CPU itself. The goal can be as a simple denial-of-service attack or a combination of attacks that might cause DHCP Snooping to fail to examine every message, allowing other DHCP attacks to then work.

To help prevent this kind of attack, DHCP Snooping includes an optional feature that tracks the number of incoming DHCP messages. If the number of incoming DHCP messages exceeds that limit over a one-second period, DHCP Snooping treats the event as an attack and moves the port to an err-disabled state. Also, the feature can be enabled both on trusted and untrusted interfaces.

Although rate limiting DHCP messages can help, placing the port in an err-disabled state can itself create issues. As a reminder, once in the err-disabled state, the switch will not send or receive frames for the interface. However, the err-disabled state might be too severe an action because the default recovery action for an err-disabled state requires the configuration of a **shutdown** and then a **no shutdown** subcommand on the interface.

To help strike a better balance, you can enable DHCP Snooping rate limiting and then also configure the switch to automatically recover from the port's err-disabled state, without the need for a **shutdown** and then **no shutdown** command.

Example 8-3 shows how to enable DHCP Snooping rate limits and err-disabled recovery. First, look at the lower half of the configuration, to the interfaces, to see the straightforward setting of the per-interface limits using the **ip dhcp snooping rate limit** *number* interface subcommands. The top of the configuration uses two global commands to tell IOS to recover from an err-disabled state if it is caused by DHCP Snooping, and to use a nondefault number of seconds to wait before recovering the interface. Note that the configuration in Example 8-3 would rely on the core configuration for DHCP Snooping as shown in Example 8-1.

Example 8-3 *Configuring DHCP Snooping Message Rate Limits*

```
errdisable recovery cause dhcp-rate-limit
errdisable recovery interval 30
!
interface GigabitEthernet1/0/2
 ip dhcp snooping limit rate 10
!
interface GigabitEthernet1/0/3
 ip dhcp snooping limit rate 2
```

A repeat of the **show ip dhcp snooping** command now shows the rate limits near the end of the output, as noted in Example 8-4.

Example 8-4 *Confirming DHCP Snooping Rate Limits*

```
SW2# show ip dhcp snooping
! Lines omitted for brevity

Interface                  Trusted    Allow option    Rate limit (pps)
-----------------------    -------    ------------    ----------------
GigabitEthernet1/0/2       yes        yes             10
  Custom circuit-ids:
GigabitEthernet1/0/3       no         no              2
  Custom circuit-ids:
```

DHCP Snooping Configuration Summary

The following configuration checklist summarizes the commands included in this section about how to configure DHCP Snooping.

Key Topic

Config Checklist

Step 1. Configure this pair of commands (both required):

- **A.** Use the **ip dhcp snooping** global command to enable DHCP Snooping on the switch.
- **B.** Use the **ip dhcp snooping vlan** *vlan-list* global command to identify the VLANs on which to use DHCP Snooping.

Step 2. **(Optional):** Use the **no ip dhcp snooping information option** global command on Layer 2 switches to disable the insertion of DHCP Option 82 data into DHCP messages, specifically on switches that do not act as a DHCP relay agent.

Step 3. Configure the **ip dhcp snooping trust** interface subcommand to override the default setting of not trusted.

Step 4. **(Optional):** Configure DHCP Snooping rate limits and err-disabled recovery:

- **Step A.** **(Optional):** Configure the **ip dhcp snooping limit rate** *number* interface subcommand to set a limit of DHCP messages per second.
- **Step B.** **(Optional):** Configure the **no ip dhcp snooping limit rate** *number* interface subcommand to remove an existing limit and reset the interface to use the default of no rate limit.

Step C. **(Optional):** Configure the **errdisable recovery cause dhcp-rate-limit** global command to enable the feature of automatic recovery from err-disabled mode, assuming the switch placed the port in err-disabled state because of exceeding DHCP Snooping rate limits.

Step D. **(Optional):** Configure the **errdisable recovery interval** *seconds* global commands to set the time to wait before recovering from an interface err-disabled state (regardless of the cause of the err-disabled state).

Dynamic ARP Inspection

The Dynamic ARP Inspection (DAI) feature on a switch examines incoming ARP messages on untrusted ports to filter those it believes to be part of an attack. DAI's core feature compares incoming ARP messages with two sources of data: the DHCP Snooping binding table and any configured ARP ACLs. If the incoming ARP message does not match the tables in the switch, the switch discards the ARP message.

This section follows the same sequence as with the DHCP Snooping section, first examining the concepts behind DAI and ARP attacks, and then showing how to configure DAI with both required and optional features.

DAI Concepts

To understand the attacks DAI can prevent, you need to be ready to compare normal ARP operations with the abnormal use of ARP used in some types of attacks. This section uses that same flow, first reviewing a few important ARP details, and then showing how an attacker can just send an ARP reply—called a gratuitous ARP—triggering hosts to add incorrect ARP entries to their ARP tables.

Review of Normal IP ARP

If all you care about is how ARP works normally, with no concern about attacks, you can think of ARP to the depth shown in Figure 8-9. The figure shows a typical sequence. Host PC1 needs to send an IP packet to its default router (R2), so PC1 first sends an ARP request message in an attempt to learn the MAC address associated with R2's 172.16.2.2 address. Router R2 sends back an ARP reply, listing R2's MAC address (note the figure shows pseudo MAC addresses to save space).

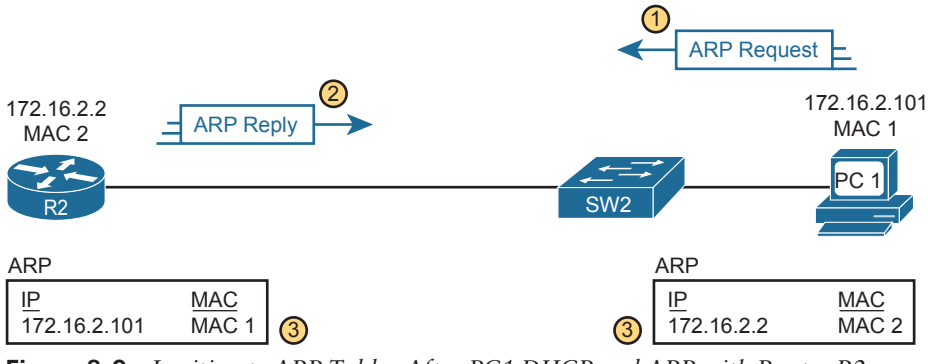


Figure 8-9 *Legitimate ARP Tables After PC1 DHCP and ARP with Router R2*

The ARP tables at bottom of the figure imply an important fact: both hosts learn the other host's MAC address with this two-message flow. Not only does PC1 learn R2's MAC address based on the ARP reply (message 2), but router R2 learns PC1's IP and MAC address because of the ARP request (message 1). To see why, take a look at the more detailed view of those messages as shown in Figure 8-10.

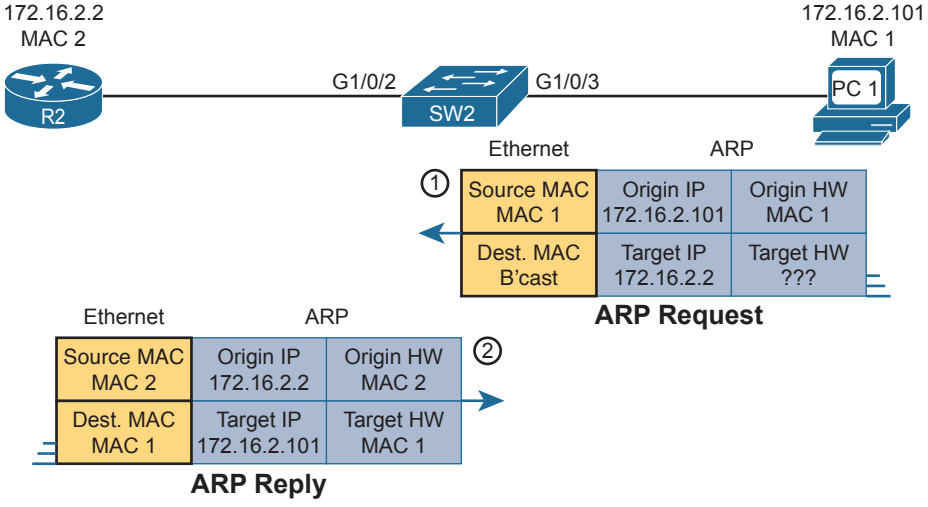


Figure 8-10 *A Detailed Look at ARP Request and Reply*

The ARP messages define origin IP and hardware (MAC) address fields as well as target IP and hardware address fields. The origin should list the sending device's IP address and MAC, no matter whether the message is an ARP reply or ARP request. For instance, message 1 in the figure, sent by PC1, lists PC1's IP and MAC addresses in the origin fields, which is why router R2 could learn that information. PC2 likewise learns of R2's MAC address per the origin address fields in the ARP reply.

8

Gratuitous ARP as an Attack Vector

Normally, a host uses ARP when it knows the IP address of another host and wants to learn that host's MAC address. However, for legitimate reasons, a host might also want to inform all the hosts in the subnet about its MAC address. That might be useful when a host changes its MAC address, for instance. So, ARP supports the idea of a gratuitous ARP message with these features:

- It is an ARP reply.
- It is sent without having first received an ARP request.
- It is sent to an Ethernet destination broadcast address so that all hosts in the subnet receive the message.

For instance, if a host's MAC address is MAC A, and it changes to MAC B, to cause all the other hosts to update their ARP tables, the host could send a gratuitous ARP that lists an origin MAC of MAC B.

Attackers can take advantage of gratuitous ARPs because they let the sending host make other hosts change their ARP tables. Figure 8-11 shows just such an example initiated by PC A

(an attacker) with a gratuitous ARP. However, this ARP lists PC1's IP address but a different device's MAC address (PC A) at step 1, causing the router to update its ARP table (step 2).

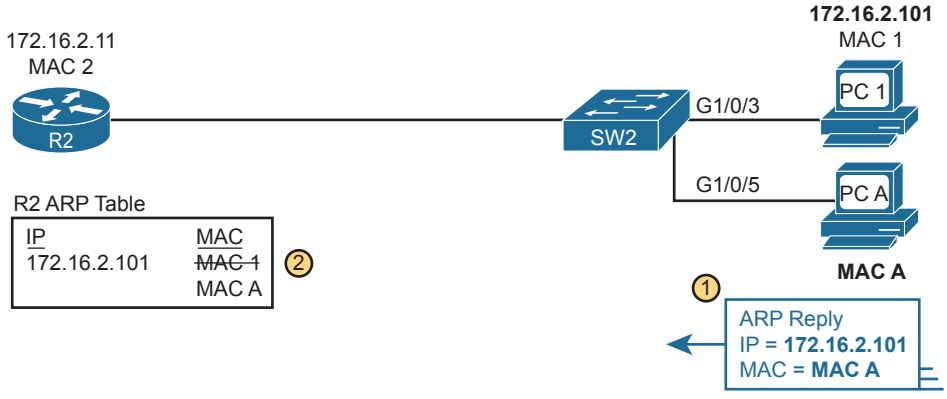


Figure 8-11 *Nefarious Use of ARP Reply Causes Incorrect ARP Data on R2*

At this point, when R2 forwards IP packets to PC1's IP address (172.16.2.101), R2 will encapsulate them in an Ethernet frame with PC A as the destination rather than with PC1's MAC address. At first, this might seem to stop PC1 from working, but instead it could be part of a man-in-the-middle attack so that PC A can copy every message. Figure 8-12 shows the idea of what happens at this point:

1. PC1 sends messages to some server on the left side of router R2.
2. The server replies to PC1's IP address, but R2 forwards that packet to PC A's MAC address, rather than to PC1.
3. PC A copies the packet for later processing.
4. PC A forwards the packet inside a new frame to PC1 so that PC1 still works.

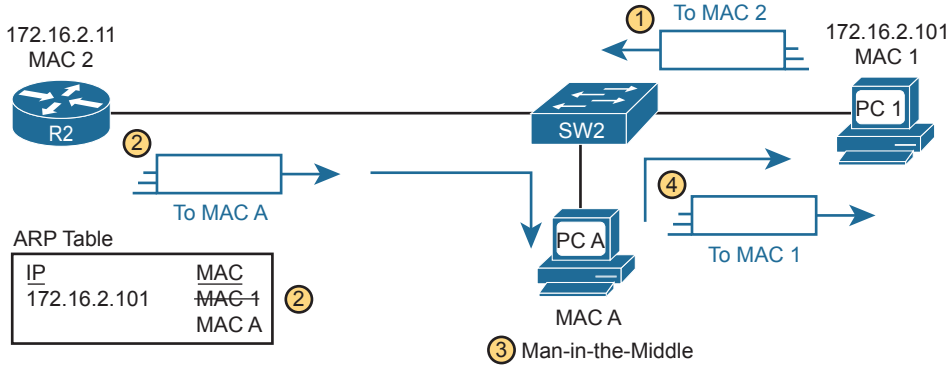


Figure 8-12 *Man-in-the-Middle Attack Resulting from Gratuitous ARP*

Dynamic ARP Inspection Logic

DAI has a variety of features that can prevent these kinds of ARP attacks. To understand how, consider the sequence of a typical client host with regards to both DHCP and ARP. When a host does not have an IP address yet—that is, before the DHCP process

completes—it does not need to use ARP. Once the host leases an IP address and learns its subnet mask, it needs ARP to learn the MAC addresses of other hosts or the default router in the subnet, so it sends some ARP messages. In short, DHCP happens first, then ARP.

DAI takes an approach for untrusted interfaces that confirms an ARP's correctness based on DHCP Snooping's data about the earlier DHCP messages. The correct normal DHCP messages list the IP address leased to a host as well as that host's MAC address. The DHCP Snooping feature also records those facts into the switch's DHCP Snooping binding table.

For any DAI untrusted ports, DAI compares the ARP message's origin IP and origin MAC address fields to the DHCP Snooping binding table. If found in the table, DAI allows the ARP through, but if not, DAI discards the ARP. For instance, Figure 8-13 shows step 1 in which the attacker at PC A attempts the gratuitous ARP shown earlier in Figure 8-11. At step 2, DAI makes a comparison to the DHCP Snooping binding table, not finding a match with MAC A along with IP address 172.16.2.101, so DAI would discard the message.

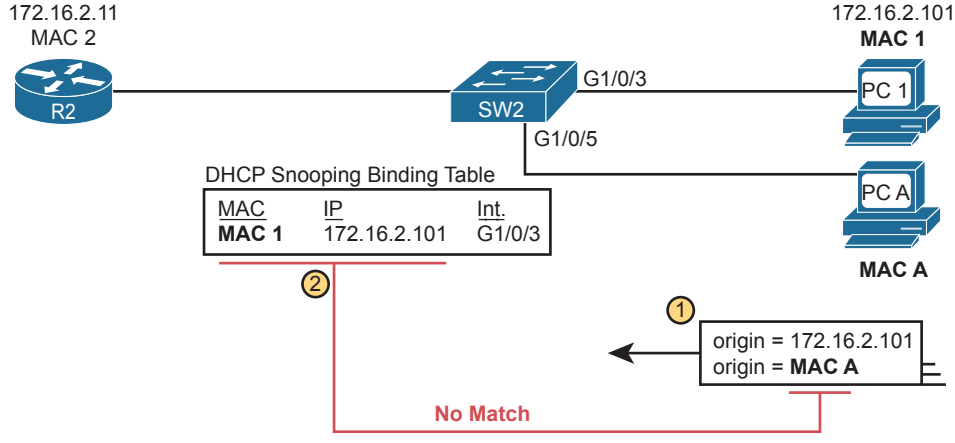


Figure 8-13 *DAI Filtering ARP Based on DHCP Snooping Binding Table*

DAI works with the idea of trusted and untrusted ports with the same general rules as DHCP Snooping. Access ports connected to end-user devices are often untrusted by both DHCP Snooping and DAI. Ports connected to other switches, routers, the DHCP server—anything other than links to end-user devices—should be trusted by DAI.

Note that although DAI can use the DHCP Snooping table as shown here, it can also use similar statically configured data that lists correct pairs of IP and MAC addresses via a tool called *ARP ACLs*. Using ARP ACLs with DAI becomes useful for ports connected to devices that use static IP addresses rather than DHCP. Note that DAI looks for both the DCHP Snooping binding data and ARP ACLs.

Beyond that core feature, note that DAI can optionally perform other checks as well. For instance, the Ethernet header that encapsulates the ARP should have addresses that match the ARP origin and target MAC addresses. Figure 8-14 shows an example of the comparison of the Ethernet source MAC address and the ARP message origin hardware field.

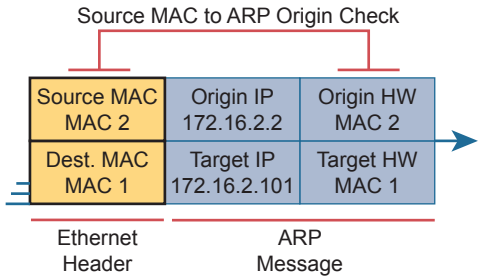


Figure 8-14 *DAI Filtering Checks for Source MAC Addresses*

DAI can be enabled to make the comparisons shown in the figure, discarding these messages:

- Messages with an Ethernet header source MAC address that is not equal to the ARP origin hardware (MAC) address
- ARP reply messages with an Ethernet header destination MAC address that is not equal to the ARP target hardware (MAC) address
- Messages with unexpected IP addresses in the two ARP IP address fields

Finally, like DHCP Snooping, DAI does its work in the switch CPU rather than in the switch ASIC, meaning that DAI itself can be more susceptible to DoS attacks. The attacker could generate large numbers of ARP messages, driving up CPU usage in the switch. DAI can avoid these problems through rate limiting the number of ARP messages on a port over time.

Dynamic ARP Inspection Configuration

Configuring DAI requires just a few commands, with the usual larger variety of optional configuration settings. This section examines DAI configuration, first with mostly default settings and with reliance on DHCP Snooping. It then shows a few of the optional features, like rate limits, automatic recovery from err-disabled state, and how to enable additional checks of incoming ARP messages.

Configuring ARP Inspection on a Layer 2 Switch

Before configuring DAI, you need to think about the feature and make a few decisions based on your goals, topology, and device roles. The decisions include the following:

- Choose whether to rely on DHCP Snooping, ARP ACLs, or both.
- If using DHCP Snooping, configure it and make the correct ports trusted for DHCP Snooping.
- Choose the VLAN(s) on which to enable DAI.
- Make DAI trusted (rather than the default setting of untrusted) on select ports in those VLANs, typically for the same ports you trusted for DHCP Snooping.

All the configuration examples in this section use the same sample network used in the DHCP Snooping configuration topics, repeated here as Figure 8-15. Just as with DHCP Snooping, switch SW2 on the right should be configured to trust the port connected to the router (G1/0/2), but not trust the two ports connected to the PCs.

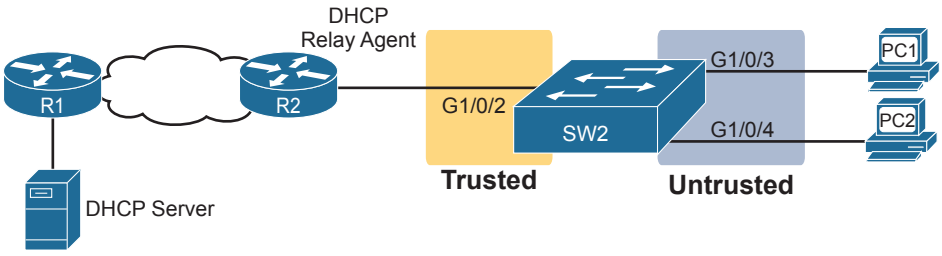


Figure 8-15 *Sample Network Used in ARP Inspection Configuration Examples*

Example 8-5 shows the required configuration to enable DAI on switch SW2 in Figure 8-15—a configuration that follows a similar progression compared to DHCP Snooping. All ports in the figure connect to VLAN 11, so to enable DAI in VLAN 11, just add the **ip arp inspection vlan 11** global command. Then, to change the logic on port G1/0/2 (connected to the router) to be trusted by DAI, add the **ip arp inspection trust** interface subcommand.

Example 8-5 *IP ARP Inspection Configuration to Match Figure 8-15*

```
ip arp inspection vlan 11
!
interface GigabitEthernet1/0/2
 ip arp inspection trust
```

Example 8-5 configures DAI, but it omits both DHCP Snooping and ARP ACLs. (If you were to configure a switch only with commands shown in Example 8-5, the switch would filter all ARPs entering all untrusted ports in VLAN 11.) Example 8-6 shows a complete and working DAI configuration that adds the DHCP Snooping configuration to match the DAI configuration in Example 8-5. Note that Example 8-6 combines Example 8-1's earlier DHCP Snooping configuration for this same topology to the DAI configuration just shown in Example 8-5, with highlights for the DAI-specific configuration lines.

Key Topic

Example 8-6 *IP DHCP Snooping Configuration Added to Support DAI*

```
ip arp inspection vlan 11
ip dhcp snooping
ip dhcp snooping vlan 11
no ip dhcp snooping information option
!
interface GigabitEthernet1/0/2
 ip dhcp snooping trust
 ip arp inspection trust
```

Remember, DHCP occurs first with DHCP clients, and then they send ARP messages. With the configuration in Example 8-6, the switch builds its DHCP Snooping binding table by analyzing incoming DHCP messages. Next, any incoming ARP messages on DAI untrusted ports must have matching information in that binding table.

Example 8-7 confirms the key facts about correct DAI operation in this sample network based on the configuration in Example 8-6. The **show ip arp inspection** command gives both configuration settings along with status variables and counters. For instance, the

highlighted lines show the total ARP messages received on untrusted ports in that VLAN and the number of dropped ARP messages (currently 0).

Example 8-7 *SW2 IP ARP Inspection Status*

```
SW2# show ip arp inspection

Source Mac Validation      : Disabled
Destination Mac Validation : Disabled
IP Address Validation      : Disabled

 Vlan     Configuration    Operation   ACL Match          Static ACL
 ----     -------------    ---------   ---------          ----------
   11     Enabled          Active

 Vlan     ACL Logging      DHCP Logging      Probe Logging
 ----     -----------      ------------      -------------
   11     Deny             Deny              Off

 Vlan       Forwarded         Dropped     DHCP Drops       ACL Drops
 ----       ---------         -------     ----------       ---------
   11              59               0              0               0

 Vlan   DHCP Permits    ACL Permits  Probe Permits   Source MAC Failures
 ----   ------------    -----------  -------------   -------------------
   11              7              0             49                     0

 Vlan   Dest MAC Failures   IP Validation Failures   Invalid Protocol Data
 ----   -----------------   ----------------------   ---------------------

 Vlan   Dest MAC Failures   IP Validation Failures   Invalid Protocol Data
 ----   -----------------   ----------------------   ---------------------
   11                   0                        0                       0
SW2# show ip dhcp snooping binding
MacAddress          IpAddress       Lease(sec) Type            VLAN  Interface
------------------  -------------   ---------- -------------   ----  --------------------
02:00:11:11:11:11   172.16.2.101    86110      dhcp-snooping    11   GigabitEthernet1/0/3
02:00:22:22:22:22   172.16.2.102    86399      dhcp-snooping    11   GigabitEthernet1/0/4
Total number of bindings: 2
```

The end of Example 8-7 shows an example of the **show ip dhcp snooping binding** command on switch SW2. Note that the first two columns list a MAC and IP address as learned from the DHCP messages. Then, imagine an ARP message arrives from PC1, a message that should list PC1's 0200.1111.1111 MAC address and 172.16.2.101 as the origin MAC and IP address, respectively. Per this output, the switch would find that matching data and allow the ARP message.

Example 8-8 shows some detail of what happens when switch SW2 receives an invalid ARP message on port G1/0/4 in Figure 8-15. In this case, to create the invalid ARP message,

PC2 in the figure was configured with a static IP address of 172.16.2.101 (which is PC1's DHCP-leased IP address). The highlights in the log message at the top of the example show PC2's claimed origin MAC and origin IP addresses in the ARP message. If you refer back to the bottom of Example 8-7, you can see that this origin MAC/IP pair does not exist in the DHCP Snooping binding table, so DAI rejects the ARP message.

Example 8-8 *Sample Results from an ARP Attack*

```
Jul 25 14:28:20.763: %SW_DAI-4-DHCP_SNOOPING_DENY: 1 Invalid ARPs (Req) on Gi1/0/4,
vlan 11.([0200.2222.2222/172.16.2.101/0000.0000.0000/172.16.2.1/09:28:20 EST Thu Jul
25 2019])

SW2# show ip arp inspection statistics

 Vlan      Forwarded         Dropped     DHCP Drops      ACL Drops
 ----      ---------         -------     ----------      ---------
   11             59              17             17              0

 Vlan   DHCP Permits    ACL Permits  Probe Permits   Source MAC Failures
 ----   ------------    -----------  -------------   -------------------
   11              7              0             49                     0

 Vlan   Dest MAC Failures   IP Validation Failures   Invalid Protocol Data
 ----   -----------------   ----------------------   ---------------------
   11                   0                        0                       0
```

The statistics from the **show ip arp inspection** command also confirms that the switch has dropped some ARP messages. The highlighted lines in the middle of the table show 17 total dropped ARP messages in VLAN 11. That same highlighted line confirms that it dropped all 17 because of the DHCP Snooping binding table ("DHCP Drops"), with zero dropped due to an ARP ACL ("ACL Drops").

Limiting DAI Message Rates

Like DHCP Snooping, DAI can also be the focus of a DoS attack with the attacker generating a large number of ARP messages. Like DHCP Snooping, DAI supports the configuration of rate limits to help prevent those attacks, with a reaction to place the port in an err-disabled state, and with the ability to configure automatic recovery from that err-disabled state.

The DHCP Snooping and DAI rate limits do have some small differences in operation, defaults, and in configuration, as follows:

- DAI defaults to use rate limits for all interfaces (trusted and untrusted), with DHCP Snooping defaulting to not use rate limits.
- DAI allows the configuration of a burst interval (a number of seconds), so that the rate limit can have logic like "*x* ARP messages over *y* seconds" (DHCP Snooping does not define a burst setting).

It helps to look at DAI and DHCP Snooping rate limit configuration together to make comparisons, so Example 8-9 shows both. The example repeats the exact same DHCP Snooping

commands in earlier Example 8-3 but adds the DAI configuration (highlighted). The configuration in Example 8-7 could be added to the configuration shown in Example 8-6 for a complete DHCP Snooping and DAI configuration.

Example 8-9 *Configuring ARP Inspection Message Rate Limits*

```
errdisable recovery cause dhcp-rate-limit
errdisable recovery cause arp-inspection
errdisable recovery interval 30
!
interface GigabitEthernet1/0/2
 ip dhcp snooping limit rate 10
 ip arp inspection limit rate 8
!
interface GigabitEthernet1/0/3
 ip dhcp snooping limit rate 2
 ip arp inspection limit rate 8 burst interval 4
```

Example 8-10 lists output that confirms the configuration settings. For instance, Example 8-9 configures port G1/0/2 with a rate of 8 messages for each (default) burst of 1 second; the output in Example 8-10 for interface G1/0/2 also lists a rate of 8 and burst interval of 1. Similarly, Example 8-9 configures port G1/0/3 with a rate of 8 over a burst of 4 seconds, with Example 8-10 confirming those same values for port G1/0/3. Note that the other two interfaces in Example 8-10 show the default settings of a rate of 15 messages over a one-second burst.

Example 8-10 *Confirming ARP Inspection Rate Limits*

```
SW2# show ip arp inspection interfaces
 Interface        Trust State     Rate (pps)     Burst Interval
 ---------------  -----------     ----------     --------------
 Gi1/0/1          Untrusted               15                  1
 Gi1/0/2          Trusted                  8                  1
 Gi1/0/3          Untrusted                8                  4
 Gi1/0/4          Untrusted               15                  1
! Lines omitted for brevity
```

Configuring Optional DAI Message Checks

As mentioned in the section titled "Dynamic ARP Inspection Logic," DAI always checks the ARP message's origin MAC and origin IP address fields versus some table in the switch, but it can also perform other checks. Those checks require more CPU, but they also help prevent other types of attacks.

Example 8-11 shows how to configure those three additional checks. Note that you can configure one, two, or all three of the options: just configure the **ip arp inspection validate** command again with all the options you want in one command, and it replaces the previous global configuration command. The example shows the three options, with the **src-mac** (source mac) option configured.

Example 8-11 *Confirming ARP Inspection Rate Limits*

```
SW2# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.

SW2(config)# ip arp inspection validate ?
  dst-mac  Validate destination MAC address
  ip       Validate IP addresses
  src-mac  Validate source MAC address

SW2(config)# ip arp inspection validate src-mac
SW2(config)# ^Z
SW2#
SW2# show ip arp inspection

Source Mac Validation      : Enabled
Destination Mac Validation : Disabled
IP Address Validation      : Disabled
```

IP ARP Inspection Configuration Summary

The following configuration checklist summarizes the commands included in this section about how to configure Dynamic IP ARP Inspection:

Key Topic

Config Checklist

Step 1. Use the **ip arp inspection vlan** *vlan-list* global command to enable Dynamic ARP Inspection (DAI) on the switch for the specified VLANs.

Step 2. Separate from the DAI configuration, also configure DHCP Snooping and/or ARP ACLs for use by DAI.

Step 3. Configure the **ip arp inspection trust** interface subcommand to override the default setting of not trusted.

Step 4. **(Optional):** Configure DAI rate limits and err-disabled recovery:

- **Step A.** **(Optional):** Configure the **ip arp inspection limit rate** *number* [**burst interval** *seconds*] interface subcommand to set a limit of ARP messages per second, or ARP messages for each configured interval.
- **Step B.** **(Optional):** Configure the **ip arp inspection limit rate none** interface subcommand to disable rate limits.
- **Step C.** **(Optional):** Configure the **errdisable recovery cause arp-inspection** global command to enable the feature of automatic recovery from err-disabled mode, assuming the switch placed the port in err-disabled state because of exceeding DAI rate limits.
- **Step D.** **(Optional):** Configure the **errdisable recovery interval** *seconds* global commands to set the time to wait before recovering from an interface err-disabled state (regardless of the cause of the err-disabled state).

Step 5. **(Optional):** Configure the **ip arp inspection validate** {[**dst-mac**] [**src-mac**] [**ip**]} global command to add DAI validation steps.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 8-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 8-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review config checklists		Book, website

Review All the Key Topics

Key Topic

Table 8-3 Key Topics for Chapter 8

Key Topic Element	Description	Page Number
Figure 8-4	DHCP filtering actions on trusted and untrusted ports	149
List	DHCP Snooping logic	149
Figure 8-6	DHCP Snooping binding table concept	151
Example 8-1	DHCP Snooping configuration	152
List	DHCP Snooping configuration checklist	155
Figure 8-10	Detail inside ARP messages with origin and target	157
List	Gratuitous ARP details	157
Figure 8-13	Core Dynamic ARP Inspection logic	159
Example 8-6	Dynamic ARP Inspection configuration with associated DHCP Snooping configuration	161
List	Dynamic ARP Inspection checklist	165

Key Terms You Should Know

DHCP Snooping, trusted port, untrusted port, DHCP Snooping binding table, Dynamic ARP Inspection, (ARP) origin IP address, (ARP) origin hardware address, ARP reply, gratuitous ARP

Command References

Tables 8-4 and 8-5 list the configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 8-4 Chapter 8 Configuration Command Reference

Command	Mode/Purpose/Description
ip dhcp snooping	Global command that enables DHCP Snooping if combined with enabling it on one or more VLANs
ip dhcp snooping vlan *vlan-list*	Global command that lists VLANs on which to enable DHCP Snooping, assuming the **ip dhcp snooping** command is also configured
[**no**] **ip dhcp snooping information option**	Command that enables (or disables with **no** option) the feature of inserting DHCP option 82 parameters by the switch when also using DHCP Snooping
[**no**] **ip dhcp snooping trust**	Interface subcommand that sets the DHCP Snooping trust state for an interface (default **no**, or untrusted)
ip dhcp snooping limit rate *number*	Interface subcommand that sets a limit to the number of incoming DHCP messages processed on an interface, per second, before DHCP Snooping discards all other incoming DHCP messages in that same second
err-disable recovery cause dhcp-rate-limit	Global command that enables the switch to automatically recover an err-disabled interface if set to that state because of exceeding a DHCP rate limit setting
err-disable recovery interval *seconds*	Global command that sets the number of seconds IOS waits before recovering any err-disabled interfaces which, per various configuration settings, should be recovered automatically
err-disable recovery cause arp-inspection	Global command that enables the switch to automatically recover an err-disabled interface if set to that state because of an ARP Inspection violation

Table 8-5 Chapter 8 EXEC Command Reference

Command	Purpose
show ip dhcp snooping	Lists a large variety of DHCP Snooping configuration settings
show ip dhcp snooping statistics	Lists counters regarding DHCP Snooping behavior on the switch
show ip dhcp snooping binding	Displays the contents of the dynamically created DHCP Snooping binding table
show ip arp inspection	Lists both configuration settings for Dynamic ARP Inspection (DAI) as well as counters for ARP messages processed and filtered
show ip arp inspection statistics	Lists the subset of the **show ip arp inspection** command output that includes counters

Part II Review

Keep track of your part review progress with the checklist shown in Table P2-1. Details on each task follow the table.

Table P2-1 Part II Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Review Videos		

Repeat All DIKTA Questions

For this task, use the PTP software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Use Per-Chapter Interactive Review Elements

Using the companion website, browse through the interactive review elements, such as memory tables and key term flashcards, to review the content from each chapter.

Labs

Depending on your chosen lab tool, here are some suggestions for what to do in the lab:

Pearson Network Simulator: If you use the full Pearson CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog Config Labs: The author's blog (https://blog.certskills.com) includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book by using the menus to navigate to the per-chapter content and then finding all config labs related to that chapter. (You can see more detailed instructions at https://blog.certskills.com/config-labs.)

Other: If using other lab tools, here are a few suggestions: make sure to experiment with the variety of configuration topics in this part, including router and switch passwords, switch port security, Dynamic ARP Inspection, and DHCP Snooping.

Watch Videos

Two chapters in this part mention videos included as extra material related to those chapters. Check out the reference in Chapter 4 to a video about using RADIUS protocol, as well as Chapter 6's reference to a video about troubleshooting switch port security.

Part III shifts to a variety of topics that can be found in most every network. None are required for a network to work, but many happen to be useful services. Most happen to use IP or support the IP network in some way, so Part III groups the topics together as IP Services.

Part III begins and ends with chapters that examine a series of smaller topics. First, Chapter 9 examines several IP services for which the CCNA exam requires you to develop configuration and verification skills. Those services include logging and syslog, the Network Time Protocol (NTP), as well as two related services: CDP and LLDP.

Chapter 12, at the end of Part III, closes with another series of smaller topics—although the CCNA 200-301 exam topics require only conceptual knowledge, not configuration skills for these topics. This chapter includes First Hop Redundancy Protocols (FHRPs), Simple Network Management Protocol (SNMP), and two related protocols: TFTP and FTP.

The two middle chapters in Part III also focus on IP-based services, beginning with Chapter 10's examination of Network Address Translation (NAT). Almost every network uses NAT with IPv4, although in many cases, the firewall implements NAT. This chapter shows how to configure and verify NAT in a Cisco router.

Chapter 11 at first may give the appearance of a large chapter about one topic—Quality of Service—and it does focus on QoS; however, QoS by nature includes a wide variety of individual QoS tools. This chapter walks you through the basic concepts of the primary QoS features.

Part III

IP Services

CHAPTER 9

Device Management Protocols

This chapter covers the following exam topics:

2.0 Network Access

2.3 Configure and verify Layer 2 discovery protocols (Cisco Discovery Protocol and LLDP)

4.0 IP Services

4.2 Configure and verify NTP operating in a client and server mode

4.5 Describe the use of syslog features including facilities and levels

This chapter begins Part III with a discussion of the concepts, configuration, and verification of three functions found on Cisco routers and switches. These functions focus more on managing the network devices themselves than on managing the network that devices create.

The first major section of this chapter focuses on log messages and syslog. Most computing devices have a need to notify the administrator of any significant issue; generally, across the world of computing, messages of this type are called log messages. Cisco devices generate log messages as well. The first section shows how a Cisco device handles those messages and how you can configure routers and switches to ignore the messages or save them in different ways.

Next, different router and switch functions benefit from synchronizing their time-of-day clocks. Like most every computing device, routers and switches have an internal clock function to keep time. Network Time Protocol (NTP) provides a means for devices to synchronize their time, as discussed in the second section.

The final major section focuses on two protocols that do the same kinds of work: Cisco Discovery Protocol (CDP) and Link Layer Discovery Protocol (LLDP). Both provide a means for network devices to learn about neighboring devices, without requiring that IPv4 or IPv6 be working at the time.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 9-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
System Message Logging (Syslog)	1–2
Network Time Protocol (NTP)	3–4
Analyzing Topology Using CDP and LLDP	5–6

1. What level of logging to the console is the default for a Cisco device?
   a. Informational
   b. Errors
   c. Warnings
   d. Debugging
2. What command limits the messages sent to a syslog server to levels 4 through 0?
   a. **logging trap 0-4**
   b. **logging trap 0,1,2,3,4**
   c. **logging trap 4**
   d. **logging trap through 4**
3. Which of the following is accurate about the NTP client function on a Cisco router?
   a. The client synchronizes its time-of-day clock based on the NTP server.
   b. It counts CPU cycles of the local router CPU to more accurately keep time.
   c. The client synchronizes its serial line clock rate based on the NTP server.
   d. The client must be connected to the same subnet as an NTP server.
4. The only NTP configuration on router R1 is the **ntp server 10.1.1.1** command. Which answers describe how NTP works on the router?
   a. As an NTP server only
   b. As an NTP client only
   c. As an NTP server only after the NTP client synchronizes with NTP server 10.1.1.1
   d. As an NTP server regardless of whether the NTP client synchronizes with NTP server 10.1.1.1
5. Imagine that a switch connects through an Ethernet cable to a router, and the router's host name is Hannah. Which of the following commands could tell you information about the IOS version on Hannah without establishing a Telnet connection to Hannah? (Choose two answers.)
   a. **show neighbors Hannah**
   b. **show cdp**
   c. **show cdp neighbors**
   d. **show cdp neighbors Hannah**
   e. **show cdp entry Hannah**
   f. **show cdp neighbors detail**

6. A switch is cabled to a router whose host name is Hannah. Which of the following LLDP commands could identify Hannah's model of hardware? (Choose two answers.)

   a. show neighbors
   b. show neighbors Hannah
   c. show lldp
   d. show lldp interface
   e. show lldp neighbors
   f. show lldp entry Hannah

Foundation Topics

System Message Logging (Syslog)

It is amazing just how helpful Cisco devices try to be to their administrators. When major (and even not-so-major) events take place, these Cisco devices attempt to notify administrators with detailed system messages. As you learn in this section, these messages vary from the very mundane to those that are incredibly important. Thankfully, administrators have a large variety of options for storing these messages and being alerted to those that could have the largest impact on the network infrastructure.

When an event happens that the device's OS thinks is interesting, how does the OS notify us humans? Cisco IOS can send the messages to anyone currently logged in to the device. It can also store the message so that a user can later look at the messages. The next few pages examine both topics.

NOTE The CCNA 200-301 exam topics list one exam topic about logging and syslog: "Describe the use of syslog features including facilities and levels." This exam topic does not require you to understand the related configuration. However, the configuration reveals many of the core concepts, so this section includes the configuration details as a means to help you understand how logging and syslog work.

Sending Messages in Real Time to Current Users

Cisco IOS running on a device at least tries to allow current users to see log messages when they happen. Not every router or switch may have users connected, but if some user is logged in, the router or switch benefits by making the network engineer aware of any issues.

By default, IOS shows log messages to console users for all severity levels of messages. That default happens because of the default **logging console** global configuration command. In fact, if you have been using a console port throughout your time reading this book, you likely have already noticed many syslog messages, like messages about interfaces coming up or going down.

For other users (that is, Telnet and SSH users), the device requires a two-step process before the user sees the messages. First, IOS has another global configuration setting—**logging monitor**—that tells IOS to enable the sending of log messages to all logged users. However, that default configuration is not enough to allow the user to see the log messages. The user must also issue the **terminal monitor** EXEC command during the login session, which tells IOS that this terminal session would like to receive log messages.

Figure 9-1 summarizes these key points about how IOS on a Cisco router or switch processes log messages for currently connected users. In the figure, user A sits at the console and always receives log messages. On the right, the fact that user B sees messages (because user B issued the **terminal monitor** command after login), and user C does not, shows that each user can control whether or not she receives log messages.

Key Topic

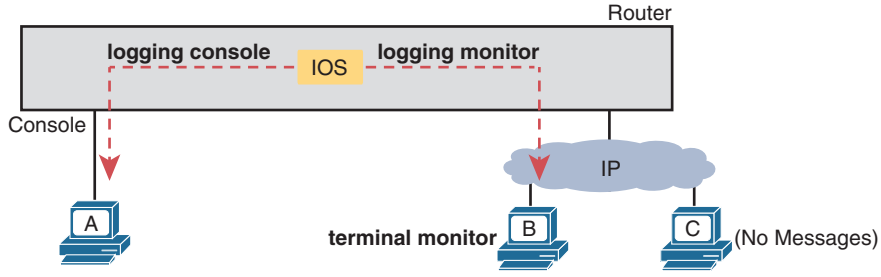


Figure 9-1 *IOS Processing for Log Messages to Current Users*

Storing Log Messages for Later Review

With logging to the console and to terminals, an event happens, IOS sends the messages to the console and terminal sessions, and then IOS can discard the message. However, clearly, it would be useful to keep a copy of the log messages for later review, so IOS provides two primary means to keep a copy.

IOS can store copies of the log messages in RAM by virtue of the **logging buffered** global configuration command. Then any user can come back later and see the old log messages by using the **show logging** EXEC command.

As a second option—an option used frequently in production networks—all devices store their log messages centrally to a syslog server. RFC 5424 defines the syslog protocol, which provides the means by which a device like a switch or router can use a UDP protocol to send messages to a syslog server for storage. All devices can send their log messages to the server. Later, a user can connect to the server (typically with a graphical user interface) and browse the log messages from various devices. To configure a router or switch to send log messages to a syslog server, add the **logging host** {*address* | *hostname*} global command, referencing the IP address or host name of the syslog server.

Figure 9-2 shows the ideas behind the buffered logging and syslog logging.

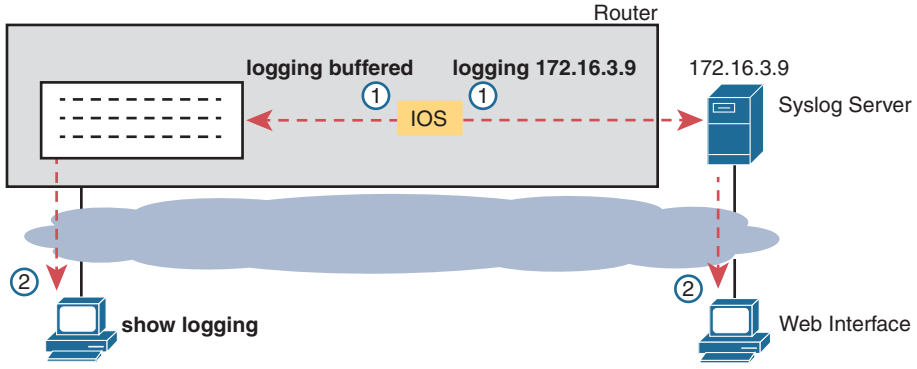


Figure 9-2 *IOS Storing Log Messages for Later View: Buffered and Syslog Server*

Log Message Format

IOS defines the format of log messages. The message begins with some data fields about the message, followed by some text more easily read by humans. For example, take a close look at this sample message:

```
*Dec 18 17:10:15.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/0, changed state to down
```

Notice that by default on this particular device, we see the following:

A timestamp: *Dec 18 17:10:15.079

The facility on the router that generated the message: %LINEPROTO

The severity level: 5

A mnemonic for the message: UPDOWN

The description of the message: Line protocol on Interface FastEthernet0/0, changed state to down

IOS dictates most of the contents of the messages, but you can at least toggle on and off the use of the timestamp (which is included by default) and a log message sequence number (which is not enabled by default). Example 9-1 reverses those defaults by turning off timestamps and turning on sequence numbers.

Example 9-1 *Disabling Timestamps and Enabling Sequence Numbers in Log Messages*

```
R1(config)# no service timestamps
R1(config)# service sequence-numbers
R1(config)# end
R1#
000011: %SYS-5-CONFIG_I: Configured from console by console
```

To see the change in format, look at the log message at the end of the example. As usual, when you exit configuration mode, the device issues yet another log message. Comparing

---

Answers to the "Do I Know This Already?" quiz:

1 D **2** C **3** A **4** C **5** E, F **6** E, F

this message to the previous example, you can see it now no longer lists the time of day but does list a sequence number.

Log Message Severity Levels

Log messages may just tell you about some mundane event, or they may tell you of some critical event. To help you make sense of the importance of each message, IOS assigns each message a severity level (as noted in the same messages in the preceding page or so). Figure 9-3 shows the severity levels: the lower the number, the more severe the event that caused the message. (Note that the values on the left and center are used in IOS commands.)

Key Topic

Keyword	Numeral	Description	
Emergency	0	System unusable	Severe
Alert	1	Immediate action required	
Critical	2	Critical Event (Highest of 3)	Impactful
Error	3	Error Event (Middle of 3)	
Warning	4	Warning Event (Lowest of 3)	
Notification	5	Normal, More Important	Normal
Informational	6	Normal, Less Important	
Debug	7	Requested by User Debug	Debug

Figure 9-3 *Syslog Message Severity Levels by Keyword and Numeral*

Figure 9-3 breaks the eight severity levels into four sections just to make a little more sense of the meaning. The two top levels in the figure are the most severe. Messages from this level mean a serious and immediate issue exists. The next three levels, called Critical, Error, and Warning, also tell about events that impact the device, but they are not as immediate and severe. For instance, one common log message about an interface failing to a physically down state shows as a severity level 3 message.

Continuing down the figure, IOS uses the next two levels (5 and 6) for messages that are more about notifying the user rather than identifying errors. Finally, the last level in the figure is used for messages requested by the **debug** command, as shown in an example later in this chapter.

Table 9-2 summarizes the configuration commands used to enable logging and to set the severity level for each type. When the severity level is set, IOS will send messages of that severity level and more severe ones (lower severity numbers) to the service identified in the command. For example, the command **logging console 4** causes IOS to send severity level 0–4 messages to the console. Also, note that the command to disable each service is the **no** version of the command, with *no* in front of the command (**no logging console**, **no logging monitor**, and so on).

Key Topic

Table 9-2 How to Configure Logging Message Levels for Each Log Service

Service	To Enable Logging	To Set Message Levels
Console	**logging console**	**logging console** *level-name* | *level-number*
Monitor	**logging monitor**	**logging monitor** *level-name* | *level-number*
Buffered	**logging buffered**	**logging buffered** *level-name* | *level-number*
Syslog	**logging host** *address* | *hostname*	**logging trap** *level-name* | *level-number*

Configuring and Verifying System Logging

With the information in Table 9-2, configuring syslog in a Cisco IOS router or switch should be relatively straightforward. Example 9-2 shows a sample, based on Figure 9-4. The figure shows a syslog server at IP address 172.16.3.9. Both switches and both routers will use the same configuration shown in Example 9-2, although the example shows the configuration process on a single device, router R1.

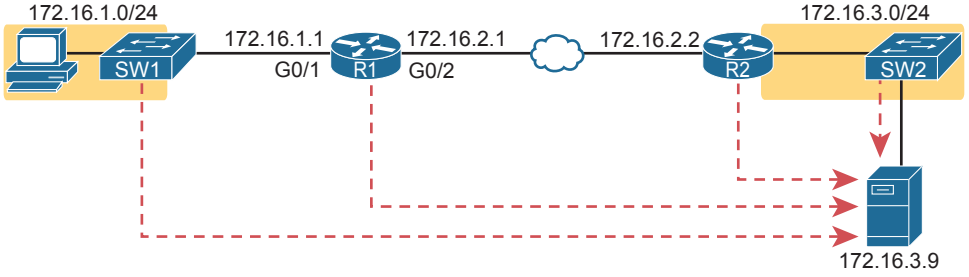


Figure 9-4 *Sample Network Used in Logging Examples*

Example 9-2 *Syslog Configuration on R1*

```
logging console 7
logging monitor debug
logging buffered 4
logging host 172.16.3.9
logging trap warning
```

First, note that the example configures the same message level at the console and for terminal monitoring (level 7, or debug), and the same level for both buffered and logging to the syslog server (level 4, or warning). The levels may be set using the numeric severity level or the name as shown earlier in Figure 9-3.

The **show logging** command confirms those same configuration settings and also lists the log messages per the logging buffered configuration. Example 9-3 shows a sample, with the configuration settings to match Example 9-2 highlighted in gray.

Example 9-3 *Viewing the Configured Log Settings per the Earlier Example*

```
R1# show logging
Syslog logging: enabled (0 messages dropped, 3 messages rate-limited, 0 flushes, 0
overruns, xml disabled, filtering disabled)

No Active Message Discriminator.

No Inactive Message Discriminator.

    Console logging: level debugging, 45 messages logged, xml disabled,
                     filtering disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level warnings, 0 messages logged, xml disabled,
                    filtering disabled
```

```
    Exception Logging: size (8192 bytes)
    Count and timestamp logging messages: disabled
    Persistent logging: disabled

No active filter modules.

    Trap logging: level warnings, 0 message lines logged
       Logging to 172.16.3.9 (udp port 514, audit disabled,
            link up),
            0 message lines logged,
            0 message lines rate-limited,
            0 message lines dropped-by-MD,
           xml disabled, sequence number disabled
           filtering disabled
       Logging Source-Interface: VRF Name:

Log Buffer (8192 bytes):
```

You might notice by now that knowing the names of all eight log message levels can be handy if you want to understand the output of the commands. Most of the **show** commands list the log message levels by name, not by number. As you can see in the gray highlights in this example, two levels list "debug," and two list "warning," even though some of the configuration commands referred to those levels by number.

Also, you cannot know this from the output, but in Example 9-3, router R1 has no buffered log messages. (Note the counter value of 0 for buffered logging messages.) If any log messages had been buffered, the actual log messages would be listed at the end of the command. In this case, I had just booted the router, and no messages had been buffered yet. (You could also clear out the old messages from the log with the **clear logging** EXEC command.)

The next example shows the difference between the current severity levels. This example shows the user disabling interface G0/1 on R1 with the **shutdown** command and then re-enabling it with the **no shutdown** command. If you look closely at the highlighted messages, you will see several severity 5 messages and one severity 3 message. The **logging buffered 4** global configuration command on R1 (see Example 9-2) means that R1 will not buffer the severity level 5 log messages, but it will buffer the severity level 3 message. Example 9-4 ends by showing that log message at the end of the output of the **show logging** command.

Example 9-4 *Seeing Severity 3 and 5 Messages at the Console, and Severity 3 Only in the Buffer*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# interface g0/1
R1(config-if)# shutdown
R1(config-if)#
*Oct 21 20:07:07.244: %LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to
administratively down
*Oct 21 20:07:08.244: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
```

```
net0/1, changed state to down
R1(config-if)# no shutdown
R1(config-if)#
*Oct 21 20:07:24.312: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to
up
*Oct 21 20:07:25.312: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net0/1, changed state to up
R1(config-if)# ^Z
R1#
*Oct 21 20:07:36.546: %SYS-5-CONFIG_I: Configured from console by console
R1# show logging
! Skipping about 20 lines, the same lines in Example 9-3, until the last few lines

Log Buffer (8192 bytes):

*Oct 21 20:07:24.312: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to
up
```

The debug Command and Log Messages

Of the eight log message severity levels, one level, debug level (7), has a special purpose: for messages generated as a result of a user logged in to the router or switch who issues a **debug** command.

The **debug** EXEC command gives the network engineer a way to ask IOS to monitor for certain internal events, with that monitoring process continuing over time, so that IOS can issue log messages when those events occur. The engineer can log in, issue the **debug** command, and move on to other work. The user can even log out of the device, and the debug remains enabled. IOS continues to monitor the request in that **debug** command and generate log messages about any related events. The debug remains active until some user issues the **no debug** command with the same parameters, disabling the debug.

NOTE While the **debug** command is just one command, it has a huge number of options, much like the **show** command may be one command, but it also has many, many options.

The best way to see how the **debug** command works, and how it uses log messages, is to see an example. Example 9-5 shows a sample debug of OSPF Hello messages for router R1 in Figure 9-4. The router (R1) enables OSPF on two interfaces and has established one OSPF neighbor relationship with router R2 (RID 2.2.2.2). The debug output shows one log message for the sent Hello on each of the four OSPF-enabled interfaces, as well as log messages for received Hello messages from each of the three OSPF neighbors.

Example 9-5 *Using* **debug ip ospf hello** *from R1's Console*

```
R1# debug ip ospf hello
OSPF hello debugging is on
R1#
*Aug 10 13:38:19.863: OSPF-1 HELLO Gi0/1: Send hello to 224.0.0.5 area 0 from
172.16.1.1
*Aug 10 13:38:21.199: OSPF-1 HELLO Gi0/2: Rcv hello from 2.2.2.2 area 0 172.16.2.2
```

```
*Aug 10 13:38:22.843: OSPF-1 HELLO Gi0/2: Send hello to 224.0.0.5 area 0 from
172.16.2.1
R1#
```

The console user sees the log messages created on behalf of that **debug** command after the debug command completes. Per the earlier configuration in Example 9-2, R1's **logging console 7** command tells us that the console user will receive severity levels 0–7, which includes level 7 debug messages. Note that with the current settings, these debug messages would not be in the local log message buffer (because of the level in the **logging buffered warning** command), nor would they be sent to the syslog server (because of the level in the **logging trap 4** command).

Note that the console user automatically sees the log messages as shown in Example 9-4. However, as noted in the text describing Figure 9-1, a user who connects to R1 would need to also issue the **terminal monitor** command to see those debug messages. For instance, anyone logged in with SSH at the time Example 9-4's output was gathered would not have seen the output, even with the **logging monitor debug** command configured on router R1, without first issuing a **terminal monitor** command.

Note that all enabled debug options use router CPU, which can cause problems for the router. You can monitor CPU use with the **show process cpu** command, but you should use caution when using **debug** commands carefully on production devices. Also, note the more CLI users that receive debug messages, the more CPU that is consumed. So, some installations choose to not include debug-level log messages for console and terminal logging, requiring users to look at the logging buffer or syslog for those messages, just to reduce router CPU load.

Network Time Protocol (NTP)

Each networking device has some concept of a date and a time-of-day clock. For instance, the log messages discussed in the first major section of this chapter had a timestamp with the date and time of day listed. Now imagine looking at all the log messages from all routers and switches stored at a syslog server. All those messages have a date and timestamp, but how do you make sure the timestamps are consistent? How do you make sure that all devices synchronize their time-of-day clocks so that you can make sense of all the log messages at the syslog server? How could you make sense of the messages for an event that impacted devices in three different time zones?

For example, consider the messages on two routers, R1 and R2, as shown in Example 9-6. Routers R1 and R2 do not synchronize their clocks. A problem keeps happening on the serial link between the two routers. A network engineer looks at all the log messages as stored on the syslog server. However, when the engineer sees some messages from R1, at 13:38:39 (around 1:40 p.m.), he does not think to look for messages from R2 that have a timestamp of around 9:45 a.m.

Example 9-6 *Log Messages from Routers R1 and R2, Compared*

```
*Oct 19 13:38:37.568: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on Serial0/0/0 from FULL
to DOWN, Neighbor Down: Interface down or detached
*Oct 19 13:38:40.568: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0,
changed state to down
```

```
! These messages happened on router R2
Oct 19 09:44:09.027: %LINK-3-UPDOWN: Interface Serial0/0/1, changed state to down
Oct 19 09:44:09.027: %OSPF-5-ADJCHG: Process 1, Nbr 1.1.1.1 on Serial0/0/1 from FULL
to DOWN, Neighbor Down: Interface down or detached
```

In reality, the messages in both parts of Example 9-6 happened within 0.5 seconds of each other because I issued a **shutdown** command on one of the routers. However, the two routers' time-of-day clocks were not synchronized, which makes the messages on the two routers look unrelated. With synchronized clocks, the two routers would have listed practically identical timestamps of almost the exact same time when these messages occurred, making it much easier to read and correlate messages.

Routers, switches, other networking devices, and pretty much every device known in the IT world has a time-of-day clock. For a variety of reasons, it makes sense to synchronize those clocks so that all devices have the same time of day, other than differences in time zone. The Network Time Protocol (NTP) provides the means to do just that.

NTP gives any device a way to synchronize their time-of-day clocks. NTP provides protocol messages that devices use to learn the timestamp of other devices. Devices send timestamps to each other with NTP messages, continually exchanging messages, with one device changing its clock to match the other, eventually synchronizing the clocks. As a result, actions that benefit from synchronized timing, like the timestamps on log messages, work much better.

This section works through a progression of topics that leads to the more common types of NTP configurations seen in real networks. The section begins with basic settings, like the timezone and initial configured time on a router or switch, followed by basic NTP configuration. The text then examines some NTP internals regarding how NTP defines the sources of time data (reference clocks) and how good each time source is (stratum). The section closes with more configuration that explains typical enterprise configurations, with multiple **ntp** commands for redundancy and the use of loopback interfaces for high availability.

Setting the Time and Timezone

NTP's job is to synchronize clocks, but NTP works best if you set the device clock to a reasonably close time before enabling the NTP client function with the **ntp server** command. For instance, my wristwatch says 8:52 p.m. right now. Before starting NTP on a new router or switch so that it synchronizes with another device, I should set the time to 8:52 p.m., set the correct date and timezone, and even tell the device to adjust for daylight savings time—and then enable NTP. Setting the time correctly gives NTP a good start toward synchronizing.

Example 9-7 shows how to set the date, time, timezone, and daylight savings time. Oddly, it uses two configuration commands (for the timezone and daylight savings time) and one EXEC command to set the date and time on the router.

Example 9-7 *Setting the Date/Time with* **clock set**, *Plus Timezone/DST*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# clock timezone EST -5
R1(config)# clock summer-time EDT recurring
R1(config)# ^Z
R1#
R1# clock set 20:52:49 21 October 2015
*Oct 21 20:52:49.000: %SYS-6-CLOCKUPDATE: System clock has been updated from 00:36:38
UTC Thu Oct 22 2015 to 20:52:49 UTC Wed Oct 21 2015, configured from console by
```

```
console.
R1# show clock
20:52:55.051 EDT Wed Oct 21 2015
```

Focus on the two configuration commands first. You should set the first two commands before setting the time of day with the **clock set** EXEC command because the two configuration commands impact the time that is set. In the first command, the **clock timezone** part defines the command and a keyword. The next parameter, "EST" in this case, is any value you choose, but choose the name of the timezone of the device. This value shows up in **show** commands, so although you make up the value, the value needs to be meaningful to all. I chose EST, the acronym for US Eastern Standard Time. The "-5" parameter means that this device is 5 hours behind Universal Time Coordinated (UTC).

The **clock summer-time** part of the second command defines what to do, again with the "EDT" being a field in which you could have used any value. However, you should use a meaningful value. This is the value shown with the time in **show** commands when daylight savings time is in effect, so I chose EDT because it is the acronym for daylight savings time in that same EST time zone. Finally, the **recurring** keyword tells the router to spring forward an hour and fall back an hour automatically over the years.

The **clock set** EXEC command then sets the time, day of the month, month, and year. However, note that IOS interprets the time as typed in the command in the context of the time zone and daylight savings time. In the example, the **clock set** command lists a time of 20:52:49 (the command uses a time syntax with a 24-hour format, not with a 12-hour format plus a.m./p.m.). As a result of that time plus the two earlier configuration commands, the **show clock** command (issued seconds later) lists that time, but also notes the time as EDT, rather than UTC time.

Basic NTP Configuration

With NTP, servers supply information about the time of day to clients, and clients react by adjusting their clocks to match. The process requires repeated small adjustments over time to maintain that synchronization. The configuration itself can be simple, or it can be extensive once you add security configuration and redundancy.

Cisco supplies two **ntp** configuration commands that dictate how NTP works on a router or switch, as follows:

Key Topic

- **ntp master** {*stratum-level*}: NTP server mode—the device acts only as an NTP server, and not as an NTP client. The device gets its time information from the internal clock on the device.
- **ntp server** {*address* | *hostname*}: NTP client/server mode—the device acts as both client and server. First, it acts as an NTP client, to synchronize time with a server. Once synchronized, the device can then act as an NTP server, to supply time to other NTP clients.

For an example showing the basic configuration syntax and **show** commands, consider Figure 9-5. With this simple configuration:

- R3 acts as an NTP server only.
- R2 acts in client/server mode—first as an NTP client to synchronize time with NTP server R3, then as a server to supply time to NTP client R1.

- R1 acts in client/server mode—first as an NTP client to synchronize time with NTP server R2. (R1 will be willing to act as a server, but no devices happen to reference R1 as an NTP server in this example.)

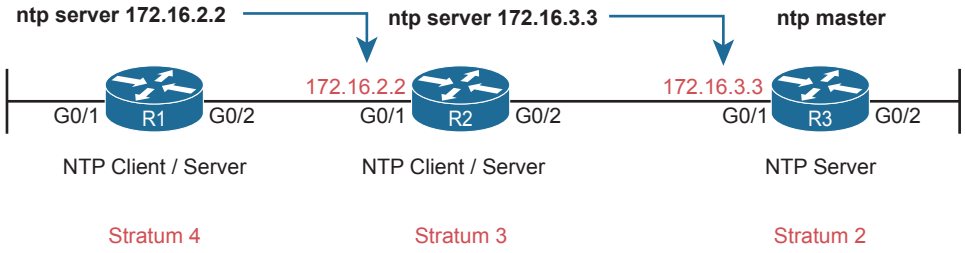


Figure 9-5 *R1 as NTP Client, R2 as Client/Server, R3 as Server*

As you can see, NTP requires little configuration to make it work with a single configuration command on each device. Example 9-8 collects the configuration from the devices shown in the figure for easy reference.

Example 9-8 *NTP Client/Server Configuration*

```
! Configuration on R1:
ntp server 172.16.2.2
! Configuration on R2:
ntp server 172.16.3.3
! Configuration on R3:
ntp master 2
```

Example 9-9 lists the output from the **show ntp status** command on R1, with the first line of output including a few important status items. First, it lists a status of synchronized, which confirms the NTP client has completed the process of changing its time to match the server's time. Any router acting as an NTP client will list "unsynchronized" in that first line until the NTP synchronization process completes with at least one server. It also confirms the IP address of the server—this device's *reference clock*—with the IP address configured in Example 9-8 (172.16.2.2).

Example 9-9 *Verifying NTP Client Status on R1*

```
R1# show ntp status
Clock is synchronized, stratum 4, reference is 172.16.2.2
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**21
ntp uptime is 1553800 (1/100 of seconds), resolution is 4000
reference time is DA5E7147.56CADEA7 (19:54:31.339 EST Thu Feb 4 2016)
clock offset is 0.0986 msec, root delay is 2.46 msec
root dispersion is 22.19 msec, peer dispersion is 5.33 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000009 s/s
system poll interval is 64, last update was 530 sec ago.
```

Next, look at the **show ntp associations** command output from both R1 and R2 as shown in Example 9-10. This command lists all the NTP servers that the local device can attempt to use, with status information about the association between the local device (client) and

the various NTP servers. Beginning with R1, note that it has one association (that is, relationship with an NTP server), based on the one **ntp server 172.16.2.2** configuration command on R1. The * means that R1 has successfully contacted the server. You will see similar data from the same command output taken from router R2.

Example 9-10 *Verifying NTP Client Status on R1 and R2*

```
R1# show ntp associations
! This output is taken from router R1, acting in client/server mode
   address    ref clock      st  when poll  reach  delay  offset  disp
*~172.16.2.2 172.16.3.3     3   50    64    377    1.223  0.090   4.469
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

```
R2# show ntp associations
! This output is taken from router R2, acting in client/server mode
 address       ref clock     st  when poll  reach  delay  offset  disp
*~172.16.3.3   127.127.1.1   2   49    64    377    1.220  -7.758  3.695
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

NTP Reference Clock and Stratum

NTP servers must learn the time from some device. For devices acting in NTP client/server mode, the device uses the NTP client function to learn the time. However, devices that act solely as an NTP server get their time from either internal device hardware or from some external clock using mechanisms other than NTP.

For instance, when configured with the **ntp master** command, a Cisco router/switch uses its internal device hardware to determine the time. All computers, networking devices included, need some means to keep time for a myriad of reasons, so they include both hardware components and software processes to keep time even over periods in which the device loses power.

Additionally, NTP servers and clients use a number to show the perceived accuracy of their reference clock data based on stratum level. The lower the stratum level, the more accurate the reference clock is considered to be. An NTP server that uses its internal hardware or external reference clock sets its own stratum level. Then, an NTP client adds 1 to the stratum level it learns from its NTP server, so that the stratum level increases the more hops away from the original clock source.

For instance, back in Figure 9-5, you can see the NTP primary server (R3) with a stratum of 2. R2, which references R3, adds 1 so it has a stratum of 3. R1 uses R2 as its NTP server, so R1 adds 1 to have a stratum of 4. These increasing stratum levels allow devices to refer to several NTP servers and then use time information from the best NTP server, best being the server with the lowest stratum level.

Routers and switches use the default stratum level of 8 for their internal reference clock based on the default setting of 8 for the stratum level in the **ntp master** [*stratum-level*] command. The command allows you to set a value from 1 through 15; in Example 9-8, the **ntp master 2** command set router R3's stratum level to 2.

NOTE NTP considers 15 to be the highest useful stratum level, so any devices that calculate their stratum as 16 consider the time data unusable and do not trust the time. So, avoid setting higher stratum values on the **ntp master** command.

To see the evidence, refer back to Example 9-10, which shows two commands based on the same configuration in Example 9-8 and Figure 9-5. The output highlights details about reference clocks and stratum levels, as follows:

R1: Per the configured **ntp server 172.16.2.2** command, the **show** command lists the same address (which is router R2's address). The ref clock (reference clock) and st (stratum) fields represent R2's reference clock as 172.16.3.3—in other words, R2's NTP server, which is R3 in this case. The st field value of 3 shows R2's stratum.

R2: Per the configured **ntp server 172.16.3.3** command, the **show** command lists 172,16,3,3, which is an address on router R3. The output notes R3's ref clock as 127.127.1.1—an indication that the server (R3) gets its clock internally. It lists R3's st (stratum) value of 2—consistent with the configured **ntp master 2** command on R3 (per Example 9-8).

On the NTP primary server itself (R3 in this case), the output has more markers indicating the use of the internal clock. Example 9-11 shows output from R3, with a reference clock of the 127.127.1.1 loopback address, used to refer to the fact that this router gets its clock data internally. Also, in the **show ntp associations** command output at the bottom, note that same address, along with a reference clock value of ".LOCL." In effect, R3, per the **ntp master** configuration command, has an association with its internal clock.

Example 9-11 *Examining NTP Server, Reference Clock, and Stratum Data*

```
R3# show ntp status
Clock is synchronized, stratum 2, reference is 127.127.1.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**20
ntp uptime is 595300 (1/100 of seconds), resolution is 4000
reference time is E0F9174C.87277EBB (16:13:32.527 daylight Sat Aug 10 2019)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 0.33 msec, peer dispersion is 0.23 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 16, last update was 8 sec ago.

R3# show ntp associations
  address         ref clock       st   when   poll reach  delay  offset   disp
*~127.127.1.1     .LOCL.           1     15     16   377  0.000   0.000  0.232
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

Redundant NTP Configuration

Instead of using a networking device as the reference clock for the enterprise, you can instead reference better time sources in the Internet or purchase a purpose-built NTP server that has better clocking hardware. For instance, an enterprise could use NTP to reference NTP servers that use an atomic clock as their reference source, like the NTP primary servers in Figure 9-6, which happen to be run by the US National Institute of Standards and Technology (NIST) (see tf.nist.gov).

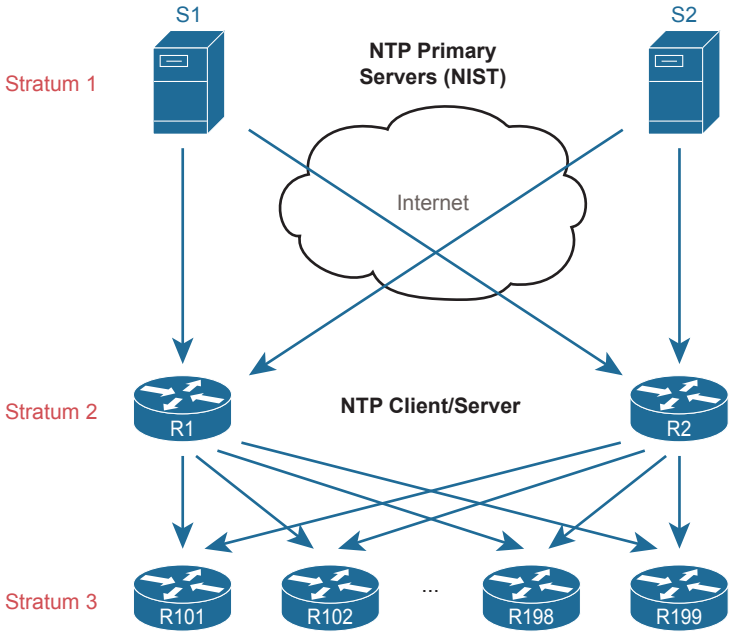


Figure 9-6 *Stratum Levels When Using an Internet-based Stratum 1 NTP Server*

NOTE While the common terms *NTP server mode* and *NTP client/server mode* are useful, the NTP RFCs (1305 and 5905) also use two other specific terms for similar ideas: *NTP primary server* and *NTP secondary server*. An NTP primary server acts only as a server, with a reference clock external to the device, and has a stratum level of 1, like the two NTP primary servers shown in Figure 9-6. NTP secondary servers are servers that use client/server mode as described throughout this section, relying on synchronization with some other NTP server.

For good design, the enterprise NTP configuration ought to refer to at least two external NTP servers for redundancy. Additionally, just a few enterprise devices should refer to those external NTP servers and then act as both NTP client and server. The majority of the devices in the enterprise, like those shown at the bottom of the figure, would act as NTP clients. Example 9-12 shows the configuration on router R1 and R2 in the figure to accomplish this design.

Example 9-12 *NTP Configuration on R1, R2 per Figure 9-6*

```
ntp server time-a-b-nist.gov
ntp server time-a-g.nist.gov
```

In addition to referencing redundant NTP primary servers, some routers in the enterprise need to be ready to supply clock data if those NTP primary servers become unreachable. An exposure exists with the configuration in Example 9-12 because if router R1 and R2 no longer hear NTP messages from the NTP servers in the Internet they will lose their only reference clock. After losing their reference clock, R1 and R2 could no longer be useful NTP servers to the rest of the enterprise.

To overcome this potential issue, the routers can also be configured with the **ntp master** command, resulting in this logic:

Key Topic

1. Establish an association with the NTP servers per the **ntp server** command.
2. Establish an association with your internal clock using the **ntp master** *stratum* command.
3. Set the stratum level of the internal clock (per the **ntp master** {*stratum-level*} command) to a higher (worse) stratum level than the Internet-based NTP servers.
4. Synchronize with the best (lowest) known time source, which will be one of the Internet NTP servers in this scenario

The logic has a few steps, but the configuration itself is simple, as shown in Example 9-13. Compared to Example 9-12, just add the **ntp master** command. The NTP servers used in this example have a stratum level of 1, so the use of the **ntp master 7** command, with a much higher stratum, will cause routers R1 and R2 to use one of the NIST NTP servers when available and use the internal clock source only when connectivity to the NIST servers is lost.

Example 9-13 *NTP Configuration on R1 and R2 to Protect Against Internet Failures*

```
ntp server time-a-b-nist.gov
ntp server time-a-g.nist.gov
ntp master 7
```

NTP Using a Loopback Interface for Better Availability

An NTP server will accept NTP messages arriving to any of its IPv4 addresses by default. However, the clients reference a specific IP address on the NTP server. That creates an availability issue.

For instance, consider the topology in Figure 9-7, with router R4 on the right acting as NTP server and the other routers acting as clients. R4 has three IP addresses that the clients could put in their **ntp server** *address* commands. Now consider what happens when one interface on R4 fails, but only one. No matter which of the three interfaces fails, that IP address on that interface cannot be used to send and receive packets. In that case, for any NTP clients that had referred to that specific IP address

- There would likely still be a route to reach R4 itself.
- The NTP client would not be able to send packets to the configured address because that interface is down.

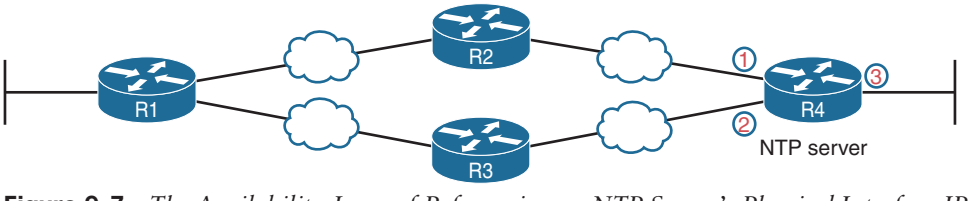


Figure 9-7 *The Availability Issue of Referencing an NTP Server's Physical Interface IP Address*

What is needed is a way to send a packet to R4, a way that is not tied to the state of any one interface. That is, as long as there is some path to send packets to R4 itself, allow NTP to keep working. The goal is to avoid the case in which a single interface failure on router R4 also causes NTP to fail.

Cisco uses the router loopback interface to meet that exact need. Loopback interfaces are virtual interfaces internal to Cisco IOS, created via the command **interface loopback** *number*, where the number is an integer. Once configured, that loopback interface exists inside that router and is not tied to any physical interface. A loopback interface can be assigned an IP address, routing protocols can advertise about the subnet, and you can ping/traceroute to that address. It acts like other physical interfaces in many ways, but once configured, it remains in an up/up state as long as

Key Topic

- The router remains up.
- You do not issue a **shutdown** command on that loopback interface.

NOTE This discussion is not about the special IPv4 loopback address 127.0.0.1. The loopback interface discussed in this section is a different concept altogether.

Example 9-14 shows the small configuration change that adds the loopback interface to the NTP configuration, which is based on Figure 9-5. In this case, the Example 9-14 configuration slightly changes the configuration shown earlier in Example 9-8. R1, still acting as client, now points to R2's new loopback interface IP address of 172.16.9.9. R2 now has configuration for a new loopback interface (loopback 0). R2 also has a command that tells it to use that loopback 0 interface's IP address as the source address when sending NTP packets.

Example 9-14 *NTP Client/Server Configuration on R1 and R2 Using a Loopback Interface*

```
! Configuration on R1, a client
ntp server 172.16.9.9
```

```
! Configuration on R2 for its server function
interface loopback 0
   ip address 172.16.9.9 255.255.255.0
!
ntp master 4
ntp source loopback 0
! Verification on router R2
R2# show interfaces loopback 0
Loopback0 is up, line protocol is up
   Hardware is Loopback
   Internet address is 172.16.9.9/24
! lines omitted for brevity
```

Loopback interfaces have a wide range of uses across IOS features. They are mentioned here with NTP because NTP is a feature that can benefit from using loopback interfaces. (As a reminder, OSPF happens to use loopback interfaces with OSPF configuration for a completely different purpose.)

Analyzing Topology Using CDP and LLDP

The first two major sections of this chapter showed two features—syslog and NTP—that work the same way on both routers and switches. This final section shows yet another feature common to both routers and switches, with two similar protocols: the Cisco Discovery Protocol (CDP) and the Link Layer Discovery Protocol (LLDP). This section focuses on CDP, followed by LLDP.

Examining Information Learned by CDP

CDP discovers basic information about neighboring routers and switches without needing to know the passwords for the neighboring devices. To discover information, routers and switches send CDP messages out each of their interfaces. The messages essentially announce information about the device that sent the CDP message. Devices that support CDP learn information about others by listening for the advertisements sent by other devices.

CDP discovers several useful details from the neighboring Cisco devices:

Key Topic

- **Device identifier:** Typically the host name
- **Address list:** Network and data-link addresses
- **Port identifier:** The interface on the remote router or switch on the other end of the link that sent the CDP advertisement
- **Capabilities list:** Information on what type of device it is (for example, a router or a switch)
- **Platform:** The model and OS level running on the device

CDP plays two general roles: to provide information to the devices to support some function and to provide information to the network engineers that manage the devices. For example, Cisco IP Phones use CDP to learn the data and voice VLAN IDs as configured on the access switch. For that second role, CDP has **show** commands that list information about neighboring devices, as well as information about how CDP is working. Table 9-3 describes the three **show** commands that list the most important CDP information.

Key Topic

Table 9-3 show cdp Commands That List Information About Neighbors

Command	Description
show cdp neighbors [*type number*]	Lists one summary line of information about each neighbor or just the neighbor found on a specific interface if an interface was listed
show cdp neighbors detail	Lists one large set (approximately 15 lines) of information, one set for every neighbor
show cdp entry *name*	Lists the same information as the **show cdp neighbors detail** command, but only for the named neighbor (case sensitive)

NOTE Cisco routers and switches support the same CDP commands, with the same parameters and same types of output.

The next example shows the power of the information in CDP commands. The example uses the network shown in Figure 9-8, with Example 9-15 listing the output of several **show cdp** commands.

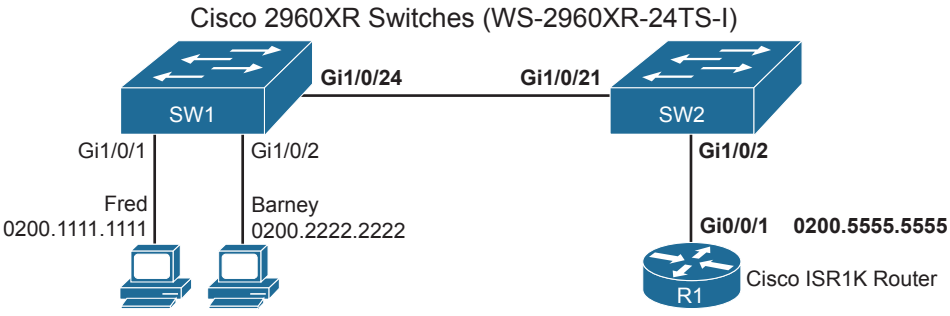


Figure 9-8 *Small Network Used in CDP Examples*

Example 9-15 **show cdp neighbors** *Command Examples: SW2*

```
SW2# show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
SW1              Gig 1/0/21        155              S I   WS-C2960X Gig 1/0/24
R1               Gig 1/0/2         131            R S I   C1111-8P  Gig 0/0/1

Total cdp entries displayed : 2
```

The **show cdp neighbors** command lists one line per neighbor. (Look for the Device ID column and the list that includes SW1 and R1.) Each of those two lines lists the most important topology information about each neighbor: the neighbor's host name (Device ID), the local device's interface, and the neighboring device's interface (under the Port heading).

Pay close attention to the local device's interface and the neighboring device's interface, comparing the example to the figure. For example, SW2's **show cdp neighbors** command lists an entry for SW1, with SW2's local interface of Gi0/2 and SW1's interface of Gi0/1 under the heading "Port ID."

This command also lists the platform, identifying the specific model of the neighboring router or switch. So, even using this basic information, you could either construct a figure like Figure 9-8 or confirm that the details in the figure are correct.

Figure 9-8 and Example 9-15 provide a good backdrop as to why devices learn about direct neighbors with CDP, but not other neighbors. First, CDP defines encapsulation that uses the data-link header, but no IP header. To ensure all devices receive a CDP message, the Ethernet header uses a multicast destination MAC address (0100.0CCC.CCCC). However, when any device that supports CDP receives a CP message, the device processes the message and then discards it, rather than forwarding it. So, for instance, when router R1 sends a CDP message to Ethernet multicast address 0100.0CCC.CCCC, switch SW2 receives it, processes it, but does not forward it to switch SW1—so SW1 will not list router R1 as a CDP neighbor.

Next, consider the **show cdp neighbors detail** command as shown in Example 9-16, again taken from switch SW2. This command lists more detail, as you might have guessed. The

detail lists the full name of the switch model (WS-2960XR-24TS-I) and the IP address configured on the neighboring device. You have to look closely, but the example has one long group of messages for each of the two neighbors; the example includes one comment line with gray highlight to help you find the dividing point between groups of messages.

Example 9-16 **show cdp neighbors detail** *Command on SW2*

```
SW2# show cdp neighbors detail
-------------------------
Device ID: SW1
Entry address(es):
  IP address: 1.1.1.1
Platform: cisco WS-C2960XR-24TS-I,  Capabilities: Switch IGMP
Interface: GigabitEthernet1/0/21,  Port ID (outgoing port): GigabitEthernet1/0/24
Holdtime : 144 sec

Version :
Cisco IOS Software, C2960X Software (C2960X-UNIVERSALK9-M), Version 15.2(6)E2, RELEASE
SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Thu 13-Sep-18 03:43 by prod_rel_team

advertisement version: 2
Protocol Hello:  OUI=0x00000C, Protocol ID=0x0112; payload len=27, value=00000000FFFFF
FFF010225010000000000000BCC4938BA180FF0000
VTP Management Domain: 'fred'
Native VLAN: 1
Duplex: full
Management address(es):
  IP address: 1.1.1.1

-------------------------
Device ID: R1
Entry address(es):
  IP address: 10.12.25.5
Platform: cisco C1111-8P,  Capabilities: Router Switch IGMP
Interface: GigabitEthernet1/0/2,  Port ID (outgoing port): GigabitEthernet0/0/1
Holdtime : 151 sec

Version :
Cisco IOS Software [Fuji], ISR Software (ARMV8EB_LINUX_IOSD-UNIVERSALK9_IAS-M), Ver-
sion 16.8.1, RELEASE SOFTWARE (fc3)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Tue 27-Mar-18 10:56 by mcpre

advertisement version: 2
```

```
VTP Management Domain: ''
Duplex: full
Management address(es):
  IP address: 10.12.25.5

Total cdp entries displayed : 2
```

NOTE The **show cdp entry** *name* command lists the exact same details shown in the output of the **show cdp neighbors detail** command, but for only the one neighbor listed in the command.

As you can see, you can sit on one device and discover a lot of information about a neighboring device—a fact that actually creates a security exposure. Cisco recommends that CDP be disabled on any interface that might not have a need for CDP. For switches, any switch port connected to another switch, a router, or to an IP phone should use CDP.

Finally, note that CDP shows information about directly connected neighbors. For instance, **show cdp neighbors** on SW1 would list an entry for SW2 in this case, but not R1, because R1 is not directly connected to SW1.

Configuring and Verifying CDP

Most of the work you do with CDP relates to what CDP can tell you with **show** commands. However, it is an IOS feature, so you can configure CDP and use some **show** commands to examine the status of CDP itself.

IOS typically enables CDP globally and on each interface by default. You can then disable CDP per interface with the **no cdp enable** interface subcommand and later re-enable it with the **cdp enable** interface subcommand. To disable and re-enable CDP globally on the device, use the **no cdp run** and **cdp run** global commands, respectively.

To examine the status of CDP itself, use the commands in Table 9-4.

Table 9-4 Commands Used to Verify CDP Operations

Command	Description
show cdp	States whether CDP is enabled globally and lists the default update and holdtime timers
show cdp interface [*type number*]	States whether CDP is enabled on each interface, or a single interface if the interface is listed, and states update and holdtime timers on those interfaces
show cdp traffic	Lists global statistics for the number of CDP advertisements sent and received

Example 9-17 lists sample output from each of the commands in Table 9-4, based on switch SW2 in Figure 9-8.

Example 9-17 **show cdp** *Commands That Show CDP Status*

```
SW2# show cdp
Global CDP information:
        Sending CDP packets every 60 seconds
        Sending a holdtime value of 180 seconds
        Sending CDPv2 advertisements is enabled

SW2# show cdp interface GigabitEthernet1/0/2
GigabitEthernet1/0/2 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds

SW2# show cdp traffic
CDP counters :
        Total packets output: 304, Input: 305
        Hdr syntax: 0, Chksum error: 0, Encaps failed: 0
        No memory: 0, Invalid packet: 0,
        CDP version 1 advertisements output: 0, Input: 0
        CDP version 2 advertisements output: 304, Input: 305
```

The first two commands in the example list two related settings about how CDP works: the send time and the hold time. CDP sends messages every 60 seconds by default, with a hold time of 180 seconds. The hold time tells the device how long to wait after no longer hearing from a device before removing those details from the CDP tables. You can override the defaults with the **cdp timer** *seconds* and **cdp holdtime** *seconds* global commands, respectively.

Examining Information Learned by LLDP

Cisco created the Cisco-proprietary CDP before any standard existed for a similar protocol. CDP has many benefits. As a Layer 2 protocol, sitting on top of Ethernet, it does not rely on a working Layer 3 protocol. It provides device information that can be useful in a variety of ways. Cisco had a need but did not see a standard that met the need, so Cisco made up a protocol, as has been the case many times over history with many companies and protocols.

Link Layer Discovery Protocol (LLDP), defined in IEEE standard 802.1AB, provides a standardized protocol that provides the same general features as CDP. LLDP has similar configuration and practically identical **show** commands as compared with CDP.

The LLDP examples all use the same topology used in the CDP examples per Figure 9-8 (the same figure used in the CDP examples). Example 9-18 lists switch SW2's LLDP neighbors as learned after LLDP was enabled on all devices and ports in that figure. The example highlights the items that match the similar output from the **show cdp neighbors** command listed at the end of the example, also from switch SW2.

Example 9-18 **show lldp neighbors** *on SW2 with Similarities to CDP Highlighted*

```
SW2# show lldp neighbors
Capability codes:
    (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
    (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other

Device ID           Local Intf     Hold-time  Capability      Port ID
R1                  Gi1/0/2        120        R               Gi0/0/1
SW1                 Gi1/0/21       120        B               Gi1/0/24

Total entries displayed: 2

SW2# show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
SW1              Gig 1/0/21        155              S I   WS-C2960X Gig 1/0/24
R1               Gig 1/0/2         131             R S I  C1111-8P  Gig 0/0/1
Total entries displayed: 2
```

The most important take-away from the output is the consistency between CDP and LLDP in how they refer to the interfaces. Both the **show cdp neighbors** and **show lldp neighbors** commands have "local intf" (interface) and "port ID" columns. These columns refer to the local device's interface and the neighboring device's interface, respectively.

However, the LLDP output in the example does differ from CDP in a few important ways:

Key Topic

- LLDP uses **B** as the capability code for switching, referring to **bridge**, a term for the device type that existed before switches that performed the same basic functions.
- LLDP does not identify IGMP as a capability, while CDP does (**I**).
- CDP lists the neighbor's **platform**, a code that defines the device type, while LLDP does not.
- LLDP lists capabilities with different conventions (see upcoming Example 9-19).

The first three items in the list are relatively straightforward, but that last item in the list requires a closer look with more detail. Interestingly, CDP lists all the capabilities of the neighbor in the **show cdp neighbors** command output, no matter whether the device currently enables all those features. LLDP instead lists the enables (configured) capabilities, rather than all supported capabilities, in the output from **show lldp neighbors** command.

LLDP makes the difference in a neighbor's total capabilities and configured capabilities with the **show lldp neighbors detail** and **show lldp entry** *hostname* commands. These commands provide identical detailed output, with the first command providing detail for all neighbors, and the second providing detail for the single listed neighbor. Example 9-19 shows the detail for neighbor R1.

Example 9-19 **show lldp entry r2** *Command on SW2*

```
SW2# show lldp entry R1

Capability codes:
    (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
    (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
------------------------------------------------
Local Intf: Gi1/0/2
Chassis id: 70ea.1a9a.d300
Port id: Gi0/0/1
Port Description: GigabitEthernet0/0/1
System Name: R1

System Description:
Cisco IOS Software [Fuji], ISR Software (ARMV8EB_LINUX_IOSD-UNIVERSALK9_IAS-M),
Version 16.8.1, RELEASE SOFTWARE (fc3)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Tue 27-Mar-18 10:56 by mcpre
Time remaining: 100 seconds
System Capabilities: B,R
Enabled Capabilities: R
Management Addresses:
    IP: 10.12.25.5
Auto Negotiation - not supported
Physical media capabilities - not advertised
Media Attachment Unit type - not advertised
Vlan ID: - not advertised


Total entries displayed: 1
```

First, regarding the device capabilities, note that the LLDP command output lists two lines about the neighbor's capabilities:

System Capabilities: What the device can do

Enabled Capabilities: What the device does now with its current configuration

For instance, in Example 9-19, the neighboring R1 claims the ability to perform routing and switching (codes **R** and **B**) but also claims to currently be using only its routing capability, as noted in the "enabled capabilities" line.

Also, take a moment to look at the output for the similarities to CDP. For instance, this output lists detail for neighbor, R1, which uses its local port G0/0/1, with a host name of R1. The output also notes the IOS name and version, from which an experienced person can infer the model number, but there is no explicit mention of the model.

NOTE LLDP uses the same messaging concepts as CDP, encapsulating messages directly in data-link headers. Devices do not forward LLDP messages so that LLDP learns only of directly connected neighbors. LLDP does use a different multicast MAC address (0180.C200.000E).

Configuring and Verifying LLDP

LLDP uses a similar configuration model as CDP, but with a few key differences. First, Cisco devices default to disable LLDP. Additionally, LLDP separates the sending and receiving of LLDP messages as separate functions. For instance, LLDP support processing receives LLDP messages on an interface so that the switch or router learns about the neighboring device while not transmitting LLDP messages to the neighboring device. To support that model, the commands include options to toggle on|off the transmission of LLDP messages separately from the processing of received messages.

The three LLDP configuration commands are as follows:

Key Topic

- **[no] lldp run:** A global configuration command that sets the default mode of LLDP operation for any interface that does not have more specific LLDP subcommands (**lldp transmit**, **lldp receive**). The **lldp run** global command enables LLDP in both directions on those interfaces, while **no lldp run** disables LLDP.
- **[no] lldp transmit:** An interface subcommand that defines the operation of LLDP on the interface regardless of the global **[no] lldp run** command. The **lldp transmit** interface subcommand causes the device to transmit LLDP messages, while **no lldp transmit** causes it to not transmit LLDP messages.
- **[no] lldp receive:** An interface subcommand that defines the operation of LLDP on the interface regardless of the global **[no] lldp run** command. The **lldp receive** interface subcommand causes the device to process received LLDP messages, while **no lldp receive** causes it to not process received LLDP messages.

For example, consider a switch that has no LLDP configuration commands at all. Example 9-20 adds a configuration that first enables LLDP for all interfaces (in both directions) with the **lldp run** global command. It then shows how to disable LLDP in both directions on Gi1/0/17 and how to disable LLDP in one direction on Gi1/0/18.

Example 9-20 *Enabling LLDP on All Ports, Disabling on a Few Ports*

```
lldp run
!
interface gigabitEthernet1/0/17
 no lldp transmit
 no lldp receive
!
interface gigabitEthernet1/0/18
 no lldp receive
```

Example 9-21 adds another example that again begins with a switch with all default settings. In this case, the configuration does not enable LLDP for all interfaces with the **lldp run** command, meaning that all interfaces default to not transmit and not receive LLDP

messages. The example does show how to then enable LLDP for both directions on one interface and in one direction for a second interface.

Example 9-21 *Enabling LLDP on Limited Ports, Leaving Disabled on Most*

```
interface gigabitEthernet1/0/19
 lldp transmit
 lldp receive
!
interface gigabitEthernet1/0/20
 lldp receive
```

Finally, checking LLDP status uses the exact same commands as CDP as listed in Table 9-4, other than the fact that you use the **lldp** keyword instead of **cdp**. For instance, **show lldp interface** lists the interfaces on which LLDP is enabled. Example 9-22 shows some examples from switch SW2 based on earlier Figure 9-8 (the same figure used in the CDP examples), with LLDP enabled in both directions on all interfaces with the **cdp run** global command.

Example 9-22 **show lldp** *Commands That Show LLDP Status*

```
SW2# show lldp
Global LLDP Information:
    Status: ACTIVE
    LLDP advertisements are sent every 30 seconds
    LLDP hold time advertised is 120 seconds
    LLDP interface reinitialisation delay is 2 seconds

SW2# show lldp interface g1/0/2

GigabitEthernet1/0/2:
    Tx: enabled
    Rx: enabled
    Tx state: IDLE
    Rx state: WAIT FOR FRAME

SW2# show lldp traffic

LLDP traffic statistics:
    Total frames out: 259
    Total entries aged: 0
    Total frames in: 257
    Total frames received in error: 0
    Total frames discarded: 0
    Total TLVs discarded: 0
    Total TLVs unrecognized: 0
```

Also, note that like CDP, LLDP uses a send timer and hold timer for the same purposes as CDP. The example shows the default settings of 30 seconds for the send timer and 120 seconds for the hold timer. You can override the defaults with the **lldp timer** *seconds* and **lldp holdtime** *seconds* global commands, respectively.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 9-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 9-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, app
Do labs		Blog
Review command references		Book

Review All the Key Topics

Key Topic

Table 9-6 Key Topics for Chapter 9

Key Topic Element	Description	Page Number
Figure 9-1	Logging to console and terminal	175
Figure 9-2	Logging to syslog and buffer	176
Figure 9-3	Log message levels	177
Table 9-2	Logging configuration commands	177
List	The **ntp master** and **ntp server** commands	183
List	Sequence for NTP client to choose a reference clock	188
List	Key facts about loopback interfaces	189
List	Information gathered by CDP	190
Table 9-3	Three CDP **show** commands that list information about neighbors	190
List	Differences between LLDP and CDP	195
List	LLDP configuration commands and logic	197

Key Terms You Should Know

log message, syslog server, Network Time Protocol (NTP), NTP client, NTP client/server mode, NTP server, NTP synchronization, CDP, LLDP

Command References

Tables 9-7 and 9-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 9-7 Configuration Command Reference

Command	Description
[no] logging console	Global command that enables (or disables with the **no** option) logging to the console device.
[no] logging monitor	Global command that enables (or disables with the **no** option) logging to users connected to the device with SSH or Telnet.
[no] logging buffered	Global command that enables (or disables with the **no** option) logging to an internal buffer.
logging [**host**] *ip-address* **|** *hostname*	Global command that enables logging to a syslog server.
logging console *level-name* **|** *level-number*	Global command that sets the log message level for console log messages.
logging monitor *level-name* **|** *level-number*	Global command that sets the log message level for log messages sent to SSH and Telnet users.
logging buffered *level-name | level-number*	Global command that sets the log message level for buffered log messages displayed later by the **show logging** command.
logging trap *level-name* **|** *level-number*	Global command that sets the log message level for messages sent to syslog servers.
[no] service sequence-numbers	Global command to enable or disable (with the **no** option) the use of sequence numbers in log messages.
clock timezone *name +–number*	Global command that names a timezone and defines the +/– offset versus UTC.
clock summertime *name* **recurring**	Global command that names a daylight savings time for a timezone and tells IOS to adjust the clock automatically.
ntp server *address | hostname*	Global command that configures the device as an NTP client by referring to the address or name of an NTP server.
ntp master *stratum-level*	Global command that configures the device as an NTP server and assigns its local clock stratum level.
ntp source *name/number*	Global command that tells NTP to use the listed interface (by name/number) for the source IP address for NTP messages.
interface loopback *number*	Global command that, at first use, creates a loopback interface. At all uses, it also moves the user into interface configuration mode for that interface.
[no] cdp run	Global command that enables and disables (with the **no** option) CDP for the entire switch or router.
[no] cdp enable	Interface subcommand to enable and disable (with the **no** option) CDP for a particular interface.
cdp timer *seconds*	Global command that changes the CDP send timer (the frequency at which CDP sends messages).
cdp holdtime *seconds*	Global command that changes how long CDP waits since the last received message from a neighbor before believing the neighbor has failed, removing the neighbor's information from the CDP table.

Command	Description
[no] lldp run	Global command to enable and disable (with the **no** option) LLDP for the entire switch or router.
[no] lldp transmit	Interface subcommand to enable and disable (with the **no** option) the transmission of LLDP messages on the interface.
[no] lldp receive	Interface subcommand to enable and disable (with the **no** option) the processing of received LLDP messages on the interface.
lldp timer *seconds*	Global command that changes the LLDP send timer (the frequency at which LLDP sends messages).
lldp holdtime *seconds*	Global command that changes how long LLDP waits since the last received message from a neighbor before believing the neighbor has failed, removing the neighbor's information from the LLDP table.

Table 9-8 Chapter 9 EXEC Command Reference

Command	Description
show logging	Lists the current logging configuration and lists buffered log messages at the end
terminal monitor **terminal no monitor**	For a user (SSH or Telnet) session, toggles on (**terminal monitor**) or off (**terminal no monitor**) the receipt of log messages, for that one session, if **logging monitor** is also configured
[no] debug {*various***}**	EXEC command to enable or disable (with the **no** option) one of a multitude of debug options
show clock	Lists the time-of-day and the date per the local device
show ntp associations	Shows all NTP clients and servers with which the local device is attempting to synchronize with NTP
show ntp status	Shows current NTP client status in detail
show interfaces loopback *number*	Shows the current status of the listed loopback interface
show cdp | lldp neighbors [*type number*]	Lists one summary line of information about each neighbor; optionally, lists neighbors off the listed interface
show cdp | lldp neighbors detail	Lists one large set of information (approximately 15 lines) for every neighbor
show cdp | lldp entry *name*	Displays the same information as **show cdp|lldp neighbors detail** but only for the named neighbor
show cdp | lldp	States whether CDP or LLDP is enabled globally and lists the default update and holdtime timers
show cdp | lldp interface [*type number*]	States whether CDP or LDP is enabled on each interface or a single interface if the interface is listed
show cdp | lldp traffic	Displays global statistics for the number of CDP or LDP advertisements sent and received

CHAPTER 10

Network Address Translation

This chapter covers the following exam topics:

4.0 IP Services

4.1 Configure and verify inside source NAT using static and pools

This chapter examines a very popular and very important part of both enterprise and small office/home office (SOHO) networks: Network Address Translation, or NAT. NAT helped solve a big problem with IPv4: the IPv4 address space would have been completely consumed by the mid-1990s. After it was consumed, the Internet could not continue to grow, which would have significantly slowed the development of the Internet.

This chapter breaks the topics into three major sections. The first section explains the challenges to the IPv4 address space caused by the Internet revolution of the 1990s. The second section explains the basic concept behind NAT, how several variations of NAT work, and how the Port Address Translation (PAT) option conserves the IPv4 address space. The final section shows how to configure NAT from the Cisco IOS Software command-line interface (CLI) and how to troubleshoot NAT.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 10-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Perspectives on IPv4 Address Scalability	1–2
Network Address Translation Concepts	3–4
NAT Configuration and Troubleshooting	5–7

1. Which of the following summarized subnets represent routes that could have been created for CIDR's goal to reduce the size of Internet routing tables?
   a. 10.0.0.0 255.255.255.0
   b. 10.1.0.0 255.255.0.0
   c. 200.1.1.0 255.255.255.0
   d. 200.1.0.0 255.255.0.0

2. Which of the following are not private addresses according to RFC 1918? (Choose two answers.)
   a. 172.31.1.1
   b. 172.33.1.1
   c. 10.255.1.1
   d. 10.1.255.1
   e. 191.168.1.1

3. With static NAT, performing translation for inside addresses only, what causes NAT table entries to be created?
   a. The first packet from the inside network to the outside network
   b. The first packet from the outside network to the inside network
   c. Configuration using the **ip nat inside source** command
   d. Configuration using the **ip nat outside source** command

4. With dynamic NAT, performing translation for inside addresses only, what causes NAT table entries to be created?
   a. The first packet from the inside network to the outside network
   b. The first packet from the outside network to the inside network
   c. Configuration using the **ip nat inside source** command
   d. Configuration using the **ip nat outside source** command

5. NAT has been configured to translate source addresses of packets for the inside part of the network, but only for some hosts as identified by an access control list. Which of the following commands indirectly identifies the hosts?
   a. **ip nat inside source list 1 pool barney**
   b. **ip nat pool barney 200.1.1.1 200.1.1.254 netmask 255.255.255.0**
   c. **ip nat inside**
   d. **ip nat inside 200.1.1.1 200.1.1.2**

6. Examine the following configuration commands:

```
interface Ethernet0/0
 ip address 10.1.1.1 255.255.255.0
 ip nat inside
interface Serial0/0
 ip address 200.1.1.249 255.255.255.252
ip nat inside source list 1 interface Serial0/0
access-list 1 permit 10.1.1.0 0.0.0.255
```

If the configuration is intended to enable source NAT overload, which of the following commands could be useful to complete the configuration? (Choose two answers.)

   a. The **ip nat outside** command
   b. The **ip nat pat** command
   c. The **overload** keyword
   d. The **ip nat pool** command

7. Examine the following **show** command output on a router configured for dynamic NAT:

```
-- Inside Source
access-list 1 pool fred refcount 2288
 pool fred: netmask 255.255.255.240
 start 200.1.1.1 end 200.1.1.7
 type generic, total addresses 7, allocated 7 (100%), misses 965
```

Users are complaining about not being able to reach the Internet. Which of the following is the most likely cause?

   a. The problem is not related to NAT, based on the information in the command output.
   b. The NAT pool does not have enough entries to satisfy all requests.
   c. Standard ACL 1 cannot be used; an extended ACL must be used.
   d. The command output does not supply enough information to identify the problem.

Foundation Topics

Perspectives on IPv4 Address Scalability

The original design for the Internet required every organization to ask for, and receive, one or more registered classful IPv4 network numbers. The people administering the program ensured that none of the IP networks were reused. As long as every organization used only IP addresses inside its own registered network numbers, IP addresses would never be duplicated, and IP routing could work well.

Connecting to the Internet using only a registered network number, or several registered network numbers, worked well for a while. In the early to mid-1990s, it became apparent that the Internet was growing so fast that all IP network numbers would be assigned by the mid-1990s! Concern arose that the available networks would be completely assigned, and some organizations would not be able to connect to the Internet.

The main long-term solution to the IPv4 address scalability problem was to increase the size of the IP address. This one fact was the most compelling reason for the advent of IP version 6 (IPv6). (Version 5 was defined much earlier but was never deployed, so the next attempt was labeled as version 6.) IPv6 uses a 128-bit address, instead of the 32-bit address in IPv4. With the same or improved process of assigning unique address ranges to every organization connected to the Internet, IPv6 can easily support every organization and individual on the planet, with the number of IPv6 addresses theoretically reaching above 10^{38}.

Many short-term solutions to the addressing problem were suggested, but three standards worked together to solve the problem. Two of the standards work closely together: Network Address Translation (NAT) and private addressing. These features together allow many organizations to use the same unregistered IPv4 network numbers internally—and still communicate well with the Internet. The third standard, classless interdomain routing (CIDR), allows ISPs to reduce the wasting of IPv4 addresses by assigning a company a subset of a network number rather than the entire network. CIDR also can allow Internet service providers (ISP) to summarize routes such that multiple Class A, B, or C networks match a single route, which helps reduce the size of Internet routing tables.

NOTE These tools have worked well. Estimates in the early 1990s predicted that the world would run out of IPv4 addresses by the mid-1990s, but IANA did not exhaust the IPv4 address space until February 2011, and ARIN (the RIR for North America) did not exhaust its supply of public IPv4 addresses until September 2015.

CIDR

CIDR is a global address assignment convention that defines how the Internet Assigned Numbers Authority (IANA), its member agencies, and ISPs should assign the globally unique IPv4 address space to individual organizations.

CIDR, defined in RFC 4632, has two main goals. First, CIDR defines a way to assign public IP addresses, worldwide, to allow route aggregation or route summarization. These route summaries greatly reduce the size of routing tables in Internet routers.

Figure 10-1 shows a typical case of CIDR route aggregation and how CIDR could be used to replace more than 65,000 routes with one route. First, imagine that ISP 1 owns Class C networks 198.0.0.0 through 198.255.255.0—not by accident, but by purposeful and thoughtful design to make this route aggregation example possible. In other words, IANA allocated all addresses that begin with 198 to one of the five Regional Internet Registries (RIR), and that RIR assigned this entire range to one big ISP in that part of the world.

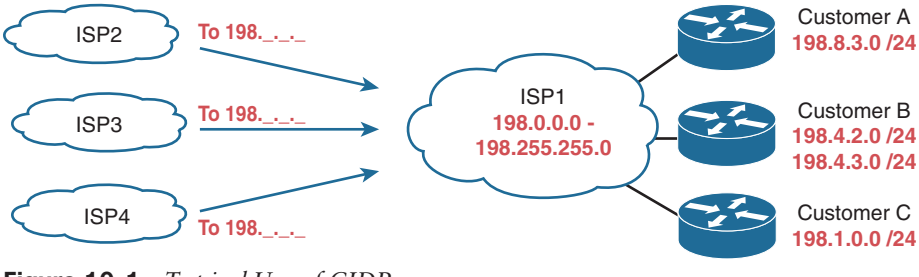


Figure 10-1 *Typical Use of CIDR*

The assignment of all addresses that begin with 198 to one ISP lets other ISPs use one route—a route for 198.0.0.0/8—to match all those addresses, forwarding packets for those addresses to ISP1. Figure 10-1 shows the ISPs on the left each with one route to 198.0.0.0/8—in other words, a route to all hosts whose IP address begins with 198. This one summary route will match packets sent to all addresses in the 65,536 Class C IP networks that begin with 198.

The second major CIDR feature allows RIRs and ISPs to reduce waste by assigning a subset of a classful network to a single customer. For example, imagine that ISP1's customer A needs only 10 IP addresses and that customer C needs 25 IP addresses. ISP1 does something like this:

- Assign customer A CIDR block 198.8.3.16/28, with 14 assignable addresses (198.8.3.17 to 198.8.3.30).
- Assign customer B CIDR block 198.8.3.32/27, with 30 assignable addresses (198.8.3.33 to 198.8.3.62).

These *CIDR blocks* act very much like a public IP network; in particular, they give each company a consecutive set of public IPv4 addresses to use. The public address assignment process has much less waste than before as well. In fact, most public address assignments for the last 20 years have been a CIDR block rather than an entire class A, B, or C network.

Private Addressing

Some computers might never be connected to the Internet. These computers' IP addresses could be duplicates of registered IP addresses in the Internet. When designing the IP addressing convention for such a network, an organization could pick and use any network number(s) it wanted, and all would be well. For example, you can buy a few routers, connect them in your office, and configure IP addresses in network 1.0.0.0, and it would work. The IP addresses you use might be duplicates of real IP addresses in the Internet, but if all you want to do is learn on the lab in your office, everything will be fine.

When building a private network that will have no Internet connectivity, you can use IP network numbers called *private internets*, as defined in RFC 1918, "Address Allocation for Private Internets." This RFC defines a set of networks that will never be assigned to any organization as a registered network number. Instead of using someone else's registered network numbers, you can use numbers in a range that are not used by anyone else in the public Internet. Table 10-2 shows the private address space defined by RFC 1918.

Key Topic

Table 10-2 RFC 1918 Private Address Space

Range of IP Addresses	Network(s)	Class of Networks	Number of Networks
10.0.0.0 to 10.255.255.255	10.0.0.0	A	1
172.16.0.0 to 172.31.255.255	172.16.0.0 – 172.31.0.0	B	16
192.168.0.0 to 192.168.255.255	192.168.0.0 – 192.168.255.0	C	256

In other words, any organization can use these network numbers. However, no organization is allowed to advertise these networks using a routing protocol on the Internet.

Answers to the "Do I Know This Already?" quiz:

1 D **2** B, E **3** C **4** A **5** A **6** A, C **7** B

Table 10-3 summarizes these important features that have helped extend the life of IPv4 by decades.

Table 10-3 Three Important Functions That Extended the Life of IPv4

Feature	RFC(s)	Main Benefits
CIDR*	4632	Assign more-specific public IPv4 address blocks to companies than Class A, B, and C networks. Aggregate routes to public IPv4 addresses based on worldwide address allocation plan.
NAT*	3022	Enable approximately 65,000 TCP/UDP sessions to be supported by a single public IPv4 address.
Private Networks	1918	Enable the use of NAT for enterprise Internet connections, with private addresses used inside the enterprise.

*CIDR and NAT may be better known for their original RFCs (1518, 1519 for CIDR; 1631 for NAT).

Network Address Translation Concepts

NAT, defined in RFC 3022, allows a host that does not have a valid, registered, globally unique IP address to communicate with other hosts through the Internet. The hosts might be using private addresses or addresses assigned to another organization. In either case, NAT allows these addresses that are not Internet ready to continue to be used and still allows communication with hosts across the Internet.

NAT achieves its goal by using a valid registered IP address to represent the private address to the rest of the Internet. The NAT function changes the private IP addresses to publicly registered IP addresses inside each IP packet, as shown in Figure 10-2.

Key Topic

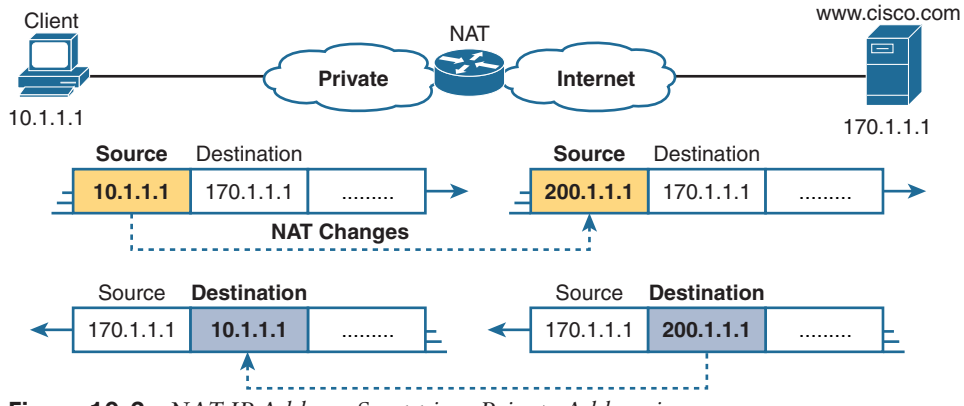


Figure 10-2 *NAT IP Address Swapping: Private Addressing*

Notice that the router, performing NAT, changes the packet's source IP address when the packet leaves the private organization. The router performing NAT also changes the destination address in each packet that is forwarded back into the private network. (Network 200.1.1.0 is a registered network in Figure 10-2.) The NAT feature, configured in the router labeled NAT, performs the translation.

This book discusses *source NAT*, which is the type of NAT that allows enterprises to use private addresses and still communicate with hosts in the Internet. Within source NAT, Cisco IOS supports several different ways to configure NAT. The next few pages cover the concepts behind several of these variations.

Static NAT

Static NAT works just like the example shown in Figure 10-2, but with the IP addresses statically mapped to each other. To help you understand the implications of static NAT and to explain several key terms, Figure 10-3 shows a similar example with more information.

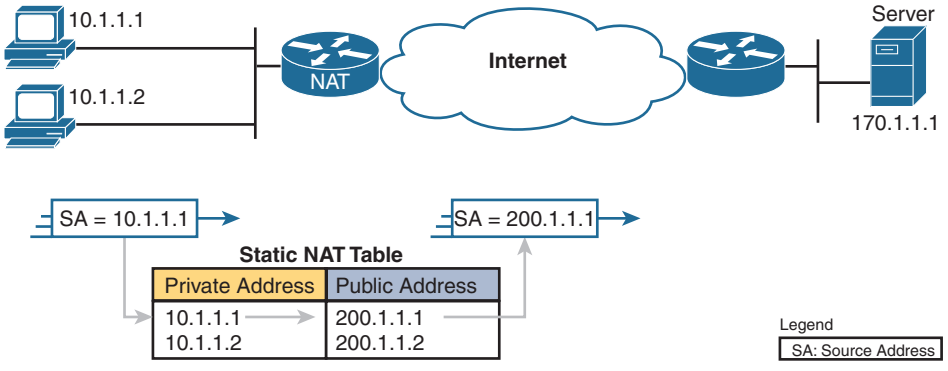


Figure 10-3 *Static NAT Showing Inside Local and Global Addresses*

First, the concepts: The company's ISP has assigned it registered network 200.1.1.0. Therefore, the NAT router must make the private IP addresses look like they are in network 200.1.1.0. To do so, the NAT router changes the source IP addresses in the packets going from left to right in the figure.

In this example, the NAT router changes the source address (SA in the figure) of 10.1.1.1 to 200.1.1.1. With static NAT, the NAT router simply configures a one-to-one mapping between the private address and the registered address that is used on its behalf. The NAT router has statically configured a mapping between private address 10.1.1.1 and public, registered address 200.1.1.1.

Supporting a second IP host with static NAT requires a second static one-to-one mapping using a second IP address in the public address range. For example, to support 10.1.1.2, the router statically maps 10.1.1.2 to 200.1.1.2. Because the enterprise has a single registered Class C network, it can support at most 254 private IP addresses with NAT, with the usual two reserved numbers (the network number and network broadcast address).

The terminology used with NAT, particularly with configuration, can be a little confusing. Notice in Figure 10-3 that the NAT table lists the private IP addresses as "private" and the public, registered addresses from network 200.1.1.0 as "public." Cisco uses the term *inside local* for the private IP addresses in this example and *inside global* for the public IP addresses.

Using NAT terminology, the enterprise network that uses private addresses, and therefore needs NAT, is the "inside" part of the network. The Internet side of the NAT function is the "outside" part of the network. A host that needs NAT (such as 10.1.1.1 in the example) has the IP address it uses inside the network, and it needs an IP address to represent it in the outside network. So, because the host essentially needs two different addresses to represent it, you need two terms. Cisco calls the private IP address used in the inside network the *inside local* address and the address used to represent the host to the rest of the Internet the *inside global* address. Figure 10-4 repeats the same example, with some of the terminology shown.

Key Topic

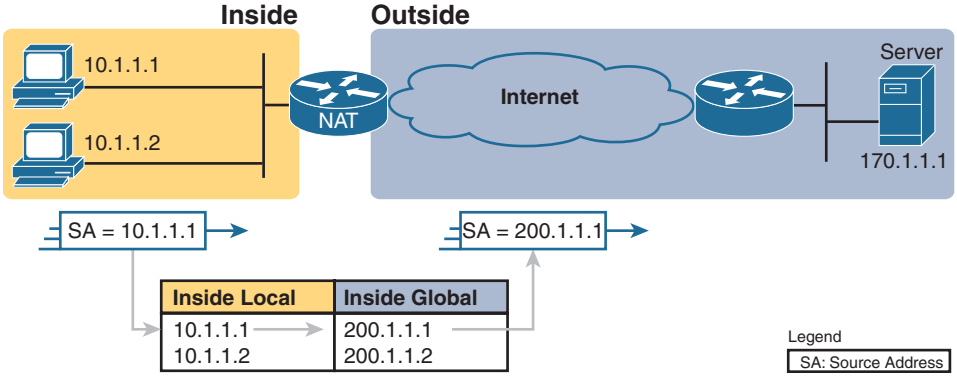


Figure 10-4 *Static NAT Terminology*

Source NAT changes only the IP address of inside hosts. Therefore, the current NAT table shown in Figure 10-4 shows the inside local and corresponding inside global registered addresses. The term *inside local* refers to the address used for the host inside the enterprise, the address used locally versus globally, which means in the enterprise instead of the global Internet. Conversely, the term *inside global* still refers to an address used for the host inside the enterprise, but it is the global address used while the packet flows through the Internet.

Note that the NAT feature called *destination NAT*, not covered in this book, uses similar terms *outside local* and *outside global*. However, with source NAT, one of the terms, *outside global*, is used. This term refers to the host that resides outside the enterprise. Because source NAT does not change that address, the term *outside global* applies at all times.

Table 10-4 summarizes these four similar terms and refers to the IPv4 addresses used as samples in the last three figures as examples.

Key Topic

Table 10-4 NAT Addressing Terms

Term	Values in Figures	Meaning
Inside local	10.1.1.1	**Inside:** Refers to the permanent location of the host, from the enterprise's perspective: it is inside the enterprise. **Local:** Means not global; that is, local. It is the address used for that host while the packet flows in the local enterprise rather than the global Internet. **Alternative:** Think of it as inside private, because this address is typically a private address.
Inside global	200.1.1.1	**Inside:** Refers to the permanent location of the host, from the enterprise's perspective. **Global:** Means global as in the global Internet. It is the address used for that host while the packet flows in the Internet. **Alternative:** Think of it as inside public, because the address is typically a public IPv4 address.
Outside global	170.1.1.1	With source NAT, the one address used by the host that resides outside the enterprise, which NAT does not change, so there is no need for a contrasting term. **Alternative:** Think of it as outside public, because the address is typically a public IPv4 address.
Outside local	—	This term is not used with source NAT. With destination NAT, the address would represent a host that resides outside the enterprise, but the address used to represent that host as packets pass through the local enterprise.

Dynamic NAT

Dynamic NAT has some similarities and differences compared to static NAT. Like static NAT, the NAT router creates a one-to-one mapping between an inside local and inside global address, and changes the IP addresses in packets as they exit and enter the inside network. However, the mapping of an inside local address to an inside global address happens dynamically.

Dynamic NAT sets up a pool of possible inside global addresses and defines matching criteria to determine which inside local IP addresses should be translated with NAT. For example, in Figure 10-5, a pool of five inside global IP addresses has been established: 200.1.1.1 through 200.1.1.5. NAT has also been configured to translate any inside local addresses that start with 10.1.1.

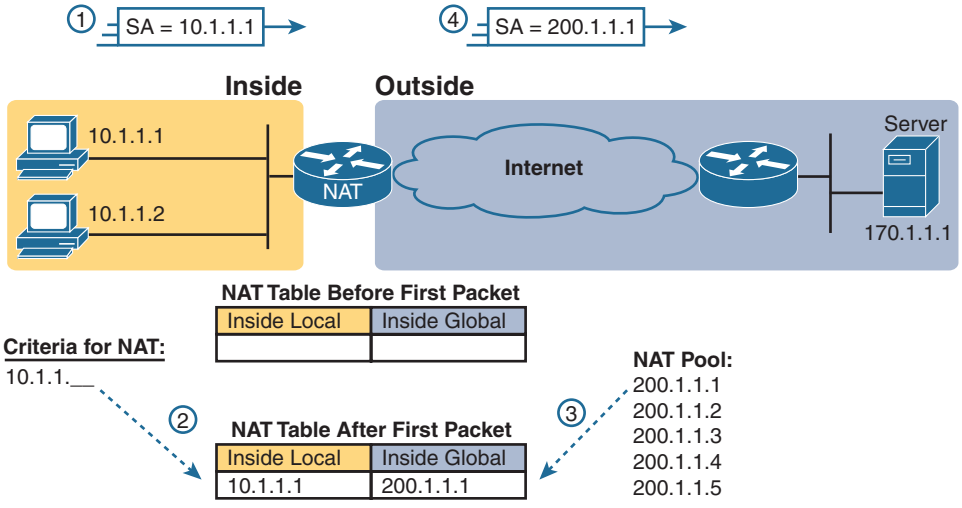


Figure 10-5 *Dynamic NAT*

The numbers 1, 2, 3, and 4 in the figure refer to the following sequence of events:

1. Host 10.1.1.1 sends its first packet to the server at 170.1.1.1.
2. As the packet enters the NAT router, the router applies some matching logic to decide whether the packet should have NAT applied. Because the logic has been configured to match source IP addresses that begin with 10.1.1, the router adds an entry in the NAT table for 10.1.1.1 as an inside local address.
3. The NAT router needs to allocate an IP address from the pool of valid inside global addresses. It picks the first one available (200.1.1.1, in this case) and adds it to the NAT table to complete the entry.
4. The NAT router translates the source IP address and forwards the packet.

The dynamic entry stays in the table as long as traffic flows occasionally. You can configure a timeout value that defines how long the router should wait, having not translated any packets with that address, before removing the dynamic entry. You can also manually clear the dynamic entries from the table using the **clear ip nat translation** * command.

NAT can be configured with more IP addresses in the inside local address list than in the inside global address pool. The router allocates addresses from the pool until all are allocated. If a new packet arrives from yet another inside host, and it needs a NAT entry, but all the pooled IP addresses are in use, the router simply discards the packet. The user must try again until a NAT entry times out, at which point the NAT function works for the next host that sends a packet. Essentially, the inside global pool of addresses needs to be as large as the maximum number of concurrent hosts that need to use the Internet at the same time—unless you use PAT, as is explained in the next section.

Overloading NAT with Port Address Translation

Some networks need to have most, if not all, IP hosts reach the Internet. If that network uses private IP addresses, the NAT router needs a very large set of registered IP addresses. With static NAT, for each private IP host that needs Internet access, you need a publicly registered IP address, completely defeating the goal of reducing the number of public IPv4

addresses needed for that organization. Dynamic NAT lessens the problem to some degree, because every single host in an internetwork should seldom need to communicate with the Internet at the same time. However, if a large percentage of the IP hosts in a network will need Internet access throughout that company's normal business hours, NAT still requires a large number of registered IP addresses, again failing to reduce IPv4 address consumption.

The NAT Overload feature, also called Port Address Translation (PAT), solves this problem. Overloading allows NAT to scale to support many clients with only a few public IP addresses.

The key to understanding how overloading works is to recall how hosts use TCP and User Datagram Protocol (UDP) ports. To see why, first consider the idea of three separate TCP connections to a web server, from three different hosts, as shown in Figure 10-6.

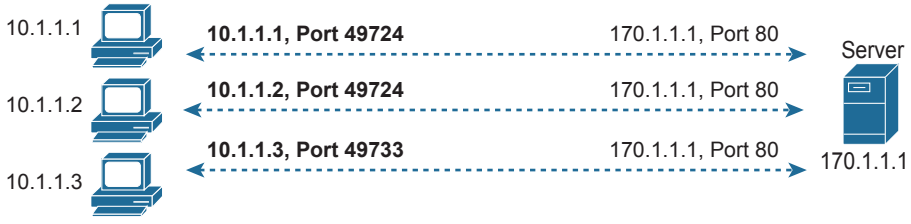


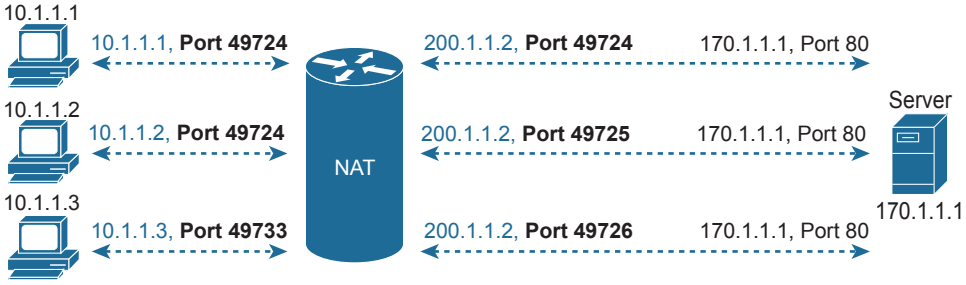
Figure 10-6 *Three TCP Connections from Three PCs*

Next, compare those three TCP connections in Figure 10-6 to three similar TCP connections, now with all three TCP connections from one client, as shown in Figure 10-7. The server does realize a difference because the server sees the IP address and TCP port number used by the clients in both figures. However, the server really does not care whether the TCP connections come from different hosts or the same host; the server just sends and receives data over each connection.

Figure 10-7 *Three TCP Connections from One PC*

NAT takes advantage of the fact that, from a transport layer perspective, the server doesn't care whether it has one connection each to three different hosts or three connections to a single host IP address. NAT overload (PAT) translates not only the address, but the port number when necessary, making what looks like many TCP or UDP flows from different hosts look like the same number of flows from one host. Figure 10-8 outlines the logic.

Key Topic



Inside Local	Inside Global
10.1.1.1: **49724**	200.1.1.2: **49724**
10.1.1.2: **49724**	200.1.1.2: **49725**
10.1.1.3: **49733**	200.1.1.2: **49726**

Dynamic NAT Table, With Overloading

Figure 10-8 *NAT Overload (PAT)*

When PAT creates the dynamic mapping, it selects not only an inside global IP address but also a unique port number to use with that address. The NAT router keeps a NAT table entry for every unique combination of inside local IP address and port, with translation to the inside global address and a unique port number associated with the inside global address. And because the port number field has 16 bits, NAT overload can use more than 65,000 port numbers, allowing it to scale well without needing many registered IP addresses—in many cases, needing only one inside global IP address.

Of the three types of NAT covered in this chapter so far, PAT is by far the most popular option. Static NAT and Dynamic NAT both require a one-to-one mapping from the inside local to the inside global address. PAT significantly reduces the number of required registered IP addresses compared to these other NAT alternatives.

NAT Configuration and Troubleshooting

The following sections describe how to configure the three most common variations of NAT: static NAT, dynamic NAT, and PAT, along with the **show** and **debug** commands used to troubleshoot NAT.

10

Static NAT Configuration

Static NAT configuration requires only a few configuration steps. Each static mapping between a local (private) address and a global (public) address must be configured. In addition, because NAT may be used on a subset of interfaces, the router must be told on which interfaces it should use NAT. Those same interface subcommands tell NAT whether the interface is inside or outside. The specific steps are as follows:

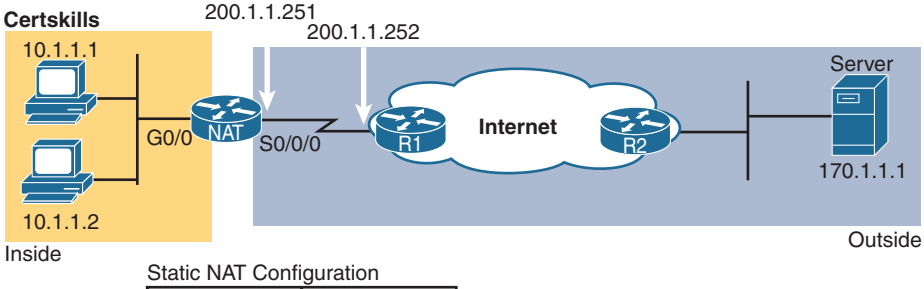
Config Checklist

Step 1. Use the **ip nat inside** command in interface configuration mode to configure interfaces to be in the inside part of the NAT design.

Step 2. Use the **ip nat outside** command in interface configuration mode to configure interfaces to be in the outside part of the NAT design.

Step 3. Use the **ip nat inside source static** *inside-local inside-global* command in global configuration mode to configure the static mappings.

Figure 10-9 shows the familiar network used in the description of static NAT earlier in this chapter, which is also used for the first several configuration examples. In Figure 10-9, you can see that Certskills has obtained Class C network 200.1.1.0 as a registered network number. That entire network, with mask 255.255.255.0, is configured on the serial link between Certskills and the Internet. With a point-to-point serial link, only two of the 254 valid IP addresses in that network are consumed, leaving 252 addresses.



Static NAT Configuration

Inside Local	Inside Global
10.1.1.1	200.1.1.1
10.1.1.2	200.1.1.2

Figure 10-9 *Sample Network for NAT Examples, with Public Class C 200.1.1.0/24*

When planning a NAT configuration, you must find some IP addresses to use as inside global IP addresses. Because these addresses must be part of some registered IP address range, it is common to use the extra addresses in the subnet connecting the enterprise to the Internet—for example, the extra 252 IP addresses in network 200.1.1.0 in this case. The router can also be configured with a loopback interface and assigned an IP address that is part of a globally unique range of registered IP addresses.

Example 10-1 lists the NAT configuration, using 200.1.1.1 and 200.1.1.2 for the two static NAT mappings.

Example 10-1 *Static NAT Configuration*

```
NAT# show running-config
!
! Lines omitted for brevity
!
interface GigabitEthernet0/0
 ip address 10.1.1.3 255.255.255.0
 ip nat inside
!
interface Serial0/0/0
 ip address 200.1.1.251 255.255.255.0
 ip nat outside
!
ip nat inside source static 10.1.1.2 200.1.1.2
ip nat inside source static 10.1.1.1 200.1.1.1

NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
```

```
--- 200.1.1.1           10.1.1.1           ---                ---
--- 200.1.1.2           10.1.1.2           ---                ---

NAT# show ip nat statistics
Total active translations: 2 (2 static, 0 dynamic; 0 extended)
Outside interfaces:
  Serial0/0/0
Inside interfaces:
  GigabitEthernet0/0
Hits: 100 Misses: 0
Expired translations: 0
Dynamic mappings:
```

The static mappings are created using the **ip nat inside source static** command. The **inside** keyword means that NAT translates addresses for hosts on the inside part of the network. The **source** keyword means that NAT translates the source IP address of packets coming into its inside interfaces. The **static** keyword means that the parameters define a static entry, which should never be removed from the NAT table because of timeout. Because the design calls for two hosts—10.1.1.1 and 10.1.1.2—to have Internet access, two **ip nat inside** commands are needed.

After creating the static NAT entries, the router needs to know which interfaces are "inside" and which are "outside." The **ip nat inside** and **ip nat outside** interface subcommands identify each interface appropriately.

A couple of **show** commands list the most important information about NAT. The **show ip nat translations** command lists the two static NAT entries created in the configuration. The **show ip nat statistics** command lists statistics, listing things such as the number of currently active translation table entries. The statistics also include the number of hits, which increments for every packet for which NAT must translate addresses.

Dynamic NAT Configuration

As you might imagine, dynamic NAT configuration differs in some ways from static NAT, but it has some similarities as well. Dynamic NAT still requires that each interface be identified as either an inside or outside interface, and of course static mapping is no longer required. Dynamic NAT uses an access control list (ACL) to identify which inside local (private) IP addresses need to have their addresses translated, and it defines a pool of registered public IP addresses to allocate. The specific steps are as follows:

Config Checklist

Step 1. Use the **ip nat inside** command in interface configuration mode to configure interfaces to be in the inside part of the NAT design (just like with static NAT).

Step 2. Use the **ip nat outside** command in interface configuration mode to configure interfaces to be in the outside part of the NAT design (just like with static NAT).

Step 3. Configure an ACL that matches the packets entering inside interfaces for which NAT should be performed.

Step 4. Use the **ip nat pool** *name first-address last-address* **netmask** *subnet-mask* command in global configuration mode to configure the pool of public registered IP addresses.

Step 5. Use the **ip nat inside source list** *acl-number* **pool** *pool-name* command in global configuration mode to enable dynamic NAT. Note the command references the ACL (step 3) and pool (step 4) per previous steps.

The next example shows a sample dynamic NAT configuration using the same network topology as the previous example (see Figure 10-9). In this case, the same two inside local addresses—10.1.1.1 and 10.1.1.2—need translation. However, unlike the previous static NAT example, the configuration in Example 10-2 places the public IP addresses (200.1.1.1 and 200.1.1.2) into a pool of dynamically assignable inside global addresses.

Example 10-2 *Dynamic NAT Configuration*

```
NAT# show running-config
!
! Lines omitted for brevity
!
interface GigabitEthernet0/0
 ip address 10.1.1.3 255.255.255.0
 ip nat inside
!
interface Serial0/0/0
 ip address 200.1.1.251 255.255.255.0
 ip nat outside
!
ip nat pool fred 200.1.1.1 200.1.1.2 netmask 255.255.255.252
ip nat inside source list 1 pool fred
!
access-list 1 permit 10.1.1.2
access-list 1 permit 10.1.1.1
```

Dynamic NAT configures the pool of public (global) addresses with the **ip nat pool** command listing the first and last numbers in an inclusive range of inside global addresses. For example, if the pool needed 10 addresses, the command might have listed 200.1.1.1 and 200.1.1.10, which means that NAT can use 200.1.1.1 through 200.1.1.10.

Dynamic NAT also performs a verification check on the **ip nat pool** command with the required **netmask** parameter. If the address range would not be in the same subnet, assuming the configured **netmask** was used on the addresses in the configured range, then IOS will reject the **ip nat pool** command. For example, as configured with the low end of 200.1.1.1, high end of 200.1.1.2, and a mask of 255.255.255.252, IOS would use the following checks, to ensure that both calculations put 200.1.1.1 and 200.1.1.2 in the same subnet:

- 200.1.1.1 with mask 255.255.255.252 implies subnet 200.1.1.0, broadcast address 200.1.1.3.
- 200.1.1.2 with mask 255.255.255.252 implies subnet 200.1.1.0, broadcast address 200.1.1.3.

If the command had instead showed low and high end values of 200.1.1.1 and 200.1.1.6, again with mask 255.255.255.252, IOS would reject the command. IOS would do the math spelled out in the following list, realizing that the numbers were in different subnets:

- 200.1.1.1 with mask 255.255.255.252 implies subnet 200.1.1.0, broadcast address 200.1.1.3.
- 200.1.1.6 with mask 255.255.255.252 implies subnet 200.1.1.4, broadcast address 200.1.1.7.

One other big difference between the dynamic NAT and static NAT configuration in Example 10-1 has to do with two options in the **ip nat inside source** command. The dynamic NAT version of this command refers to the name of the NAT pool it wants to use for inside global addresses—in this case, fred. It also refers to an IP ACL, which defines the matching logic for inside local IP addresses. So, the logic for the **ip nat inside source list 1 pool fred** command in this example is as follows:

> Create NAT table entries that map between hosts matched by ACL 1, for packets entering any inside interface, allocating an inside global address from the pool called fred.

Dynamic NAT Verification

Examples 10-3 and 10-4 show the evidence that dynamic NAT begins with no NAT table entries, but the router reacts after user traffic correctly drives the NAT function. Example 10-3 shows the output of the **show ip nat translations** and **show ip nat statistics** commands before any users generate traffic that makes NAT do some work. The **show ip nat translations** command, which lists the NAT table entries, lists a blank line; the **show ip nat statistics** command, which shows how many times NAT has created a NAT table entry, shows 0 active translations.

Example 10-3 *Dynamic NAT Verifications Before Generating Traffic*

```
! The next command lists one empty line because no entries have been dynamically
! created yet.
NAT# show ip nat translations

NAT# show ip nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)
Peak translations: 8, occurred 00:02:44 ago
Outside interfaces:
  Serial0/0/0
Inside interfaces:
  GigabitEthernet0/0
Hits: 0 Misses: 0
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[id 1] access-list 1 pool fred refcount 0
 pool fred: netmask 255.255.255.252
    start 200.1.1.1 end 200.1.1.2
    type generic, total addresses 2, allocated 0 (0%), misses 0

Total doors: 0
Appl doors: 0
Normal doors: 0
Queued Packets: 0
```

The **show ip nat statistics** command at the end of the example lists some particularly interesting troubleshooting information with two different counters labeled "misses," as

highlighted in the example. The first occurrence of this counter counts the number of times a new packet comes along, needing a NAT entry, and not finding one. At that point, dynamic NAT reacts and builds an entry. The second misses counter toward the end of the command output lists the number of misses in the pool. This counter increments only when dynamic NAT tries to allocate a new NAT table entry and finds no available addresses, so the packet cannot be translated—probably resulting in an end user not getting to the application.

Next, Example 10-4 updates the output of both commands after the user of the host at 10.1.1.1 telnets to host 170.1.1.1.

Example 10-4 *Dynamic NAT Verifications After Generating Traffic*

```
NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 200.1.1.1          10.1.1.1           ---                ---

NAT# show ip nat statistics
Total active translations: 1 (0 static, 1 dynamic; 0 extended)
Peak translations: 11, occurred 00:04:32 ago
Outside interfaces:
  Serial0/0/0
Inside interfaces:
  GigabitEthernet0/0
Hits: 69 Misses: 1
Expired translations: 0
Dynamic mappings:
-- Inside Source
access-list 1 pool fred refcount 1
[eml fred: netmask 255.255.255.252
    start 200.1.1.1 end 200.1.1.2
    type generic, total addresses 2, allocated 1 (50%), misses 0
```

The example begins with host 10.1.1.1 telnetting to 170.1.1.1 (not shown), with the NAT router creating a NAT entry. The NAT table shows a single entry, mapping 10.1.1.1 to 200.1.1.1. And, the first line in the output of the **show ip nat statistics** command lists a counter for 1 active translation, as shown in the NAT table at the top of the example.

Take an extra moment to consider the highlighted line, where the **show ip nat statistics** command lists 1 miss and 69 hits. The first miss counter, now at 1, means that one packet arrived that needed NAT, but there was no NAT table entry. NAT reacted and added a NAT table entry, so the hit counter of 69 means that the next 69 packets used the newly added NAT table entry. The second misses counter, still at 0, did not increment because the NAT pool had enough available inside global IP addresses to use to allocate the new NAT table entry. Also note that the last line lists statistics on the number of pool members allocated (1) and the percentage of the pool currently in use (50%).

The dynamic NAT table entries time out after a period of inactivity, putting those inside global addresses back in the pool for future use. Example 10-5 shows a sequence in which two different hosts make use of inside global address 200.1.1.1. Host 10.1.1.1 uses inside global address 200.1.1.1 at the beginning of the example. Then, instead of just waiting on

the NAT entry to time out, the example clears the NAT table entry with the **clear ip nat translation** * command. At that point, the user at 10.1.1.2 telnets to 170.1.1.1, and the new NAT table entry appears, using the same 200.1.1.1 inside global address.

Example 10-5 *Example of Reuse of a Dynamic Inside Global IP Address*

```
! Host 10.1.1.1 currently uses inside global 200.1.1.1
NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 200.1.1.1          10.1.1.1           ---                ---
NAT# clear ip nat translation *

!
! telnet from 10.1.1.2 to 170.1.1.1 happened next; not shown
!
! Now host 10.1.1.2 uses inside global 200.1.1.1

NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 200.1.1.1          10.1.1.2           ---                ---
!
! Telnet from 10.1.1.1 to 170.1.1.1 happened next; not shown
!
NAT# debug ip nat
IP NAT debugging is on

Oct 20 19:23:03.263: NAT*: s=10.1.1.1->200.1.1.2, d=170.1.1.1 [348]
Oct 20 19:23:03.267: NAT*: s=170.1.1.1, d=200.1.1.2->10.1.1.1 [348]
Oct 20 19:23:03.464: NAT*: s=10.1.1.1->200.1.1.2, d=170.1.1.1 [349]
Oct 20 19:23:03.568: NAT*: s=170.1.1.1, d=200.1.1.2->10.1.1.1 [349]
```

Finally, at the end of Example 10-5, you see that host 10.1.1.1 has telnetted to another host in the Internet, plus the output from the **debug ip nat** command. This **debug** command causes the router to issue a message every time a packet has its address translated for NAT. You generate the output results by entering a few lines from the Telnet connection from 10.1.1.1 to 170.1.1.1. The debug output tells you that host 10.1.1.1 now uses inside global address 200.1.1.2 for this new connection.

NAT Overload (PAT) Configuration

The static and dynamic NAT configurations matter, but the NAT overload (PAT) configuration in this section matters more. This is the feature that saves public IPv4 addresses and prolonged IPv4's life.

NAT overload, as mentioned earlier, allows NAT to support many inside local IP addresses with only one or a few inside global IP addresses. By essentially translating the private IP address and port number to a single inside global address, but with a unique port number, NAT can support many (more than 65,000) private hosts with only a single public, global address.

Two variations of PAT configuration exist in IOS. If PAT uses a pool of inside global addresses, the configuration looks exactly like dynamic NAT, except the **ip nat inside source list** global command has an **overload** keyword added to the end. If PAT just needs to use one inside global IP address, the router can use one of its interface IP addresses. Because NAT can support over 65,000 concurrent flows with a single inside global address, a single public IP address can support an entire organization's NAT needs.

The following statement details the configuration difference between NAT overload and 1:1 NAT when using a NAT pool:

Key Topic

Use the same steps for configuring dynamic NAT, as outlined in the previous section, but include the **overload** keyword at the end of the **ip nat inside source list** global command.

The following checklist details the configuration when using an interface IP address as the sole inside global IP address:

Config Checklist

Step 1. As with dynamic and static NAT, configure the **ip nat inside** interface subcommand to identify inside interfaces.

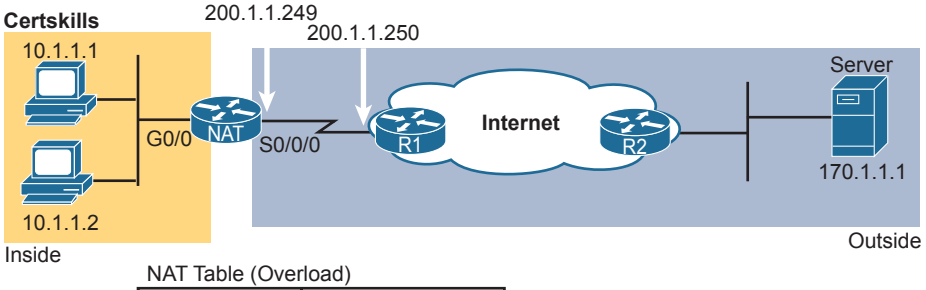
Step 2. As with dynamic and static NAT, configure the **ip nat outside** interface subcommand to identify outside interfaces.

Step 3. As with dynamic NAT, configure an ACL that matches the packets entering inside interfaces.

Step 4. Configure the **ip nat inside source list** *acl-number* **interface** *type/number* **overload** global configuration command, referring to the ACL created in step 3 and to the interface whose IP address will be used for translations.

Example 10-2 demonstrated a dynamic NAT configuration. To convert it to a PAT configuration, you would use the **ip nat inside source list 1 pool fred overload** command instead, simply adding the **overload** keyword.

The next example shows PAT configuration using a single interface IP address. Figure 10-10 shows the same familiar network, with a few changes. In this case, the ISP has given Certskills a subset of network 200.1.1.0: CIDR subnet 200.1.1.248/30. In other words, this subnet has two usable addresses: 200.1.1.249 and 200.1.1.250. These addresses are used on either end of the serial link between Certskills and its ISP. The NAT feature on the Certskills router translates all NAT addresses to its serial IP address, 200.1.1.249.



NAT Table (Overload)

Inside Local	Inside Global
10.1.1.1: 49712	200.1.1.249: 49712
10.1.1.2: 49713	200.1.1.249: 49713
10.1.1.2: 49913	200.1.1.249: 49913

Figure 10-10 *NAT Overload and PAT*

In Example 10-6, which shows the NAT overload configuration, NAT translates using inside global address 200.1.1.249 only, so the NAT pool is not required. In the example, host 10.1.1.2 creates two Telnet connections, and host 10.1.1.1 creates one Telnet connection, causing three dynamic NAT entries, each using inside global address 200.1.1.249, but each with a unique port number.

Example 10-6 *NAT Overload Configuration*

```
NAT# show running-config
!
! Lines Omitted for Brevity
!
interface GigabitEthernet0/0
 ip address 10.1.1.3 255.255.255.0
 ip nat inside
!
interface Serial0/0/0
 ip address 200.1.1.249 255.255.255.252
 ip nat outside
!
ip nat inside source list 1 interface Serial0/0/0 overload
!
access-list 1 permit 10.1.1.2
access-list 1 permit 10.1.1.1
!

NAT# show ip nat translations
Pro  Inside global      Inside local       Outside local      Outside global
tcp  200.1.1.249:49712  10.1.1.1:49712     170.1.1.1:23       170.1.1.1:23
tcp  200.1.1.249:49713  10.1.1.2:49713     170.1.1.1:23       170.1.1.1:23
tcp  200.1.1.249:49913  10.1.1.2:49913     170.1.1.1:23       170.1.1.1:23
NAT# show ip nat statistics
Total active translations: 3 (0 static, 3 dynamic; 3 extended)
Peak translations: 12, occurred 00:01:11 ago
Outside interfaces:
  Serial0/0/0
Inside interfaces:
  GigabitEthernet0/0
Hits: 103 Misses: 3
Expired translations: 0
Dynamic mappings:
-- Inside Source
access-list 1 interface Serial0/0/0 refcount 3
```

The **ip nat inside source list 1 interface serial 0/0/0 overload** command has several parameters, but if you understand the dynamic NAT configuration, the new parameters shouldn't be too hard to grasp. The **list 1** parameter means the same thing as it does for

dynamic NAT: inside local IP addresses matching ACL 1 have their addresses translated. The **interface serial 0/0/0** parameter means that the only inside global IP address available is the IP address of the NAT router's interface serial 0/0/0. Finally, the **overload** parameter means that overload is enabled. Without this parameter, the router does not perform overload, just dynamic NAT.

As you can see in the output of the **show ip nat translations** command, three translations have been added to the NAT table. Before this command, host 10.1.1.1 creates one Telnet connection to 170.1.1.1, and host 10.1.1.2 creates two Telnet connections. The router creates one NAT table entry for each unique combination of inside local IP address and port.

NAT Troubleshooting

The majority of NAT troubleshooting issues relate to getting the configuration correct. Source NAT has several configuration options—static, dynamic, PAT—with several configuration commands for each. You should work hard at building skills with the configuration so that you can quickly recognize configuration mistakes. The following troubleshooting checklist summarizes the most common source NAT issues, most of which relate to incorrect configuration.

- **Reversed inside and outside:** Ensure that the configuration includes the **ip nat inside** and **ip nat outside** interface subcommands and that the commands are not reversed (the **ip nat inside** command on outside interfaces, and vice versa). With source NAT, only the inside interface triggers IOS to add new translations, so designating the correct inside interfaces is particularly important.
- **Static NAT:** Check the **ip nat inside source static** command to ensure it lists the inside local address first and the inside global IP address second.
- **Dynamic NAT (ACL):** Ensure that the ACL configured to match packets sent by the inside hosts match that host's packets before any NAT translation has occurred. For example, if an inside local address of 10.1.1.1 should be translated to 200.1.1.1, ensure that the ACL matches source address 10.1.1.1, not 200.1.1.1.
- **Dynamic NAT (pool):** For dynamic NAT without PAT, ensure that the pool has enough IP addresses. When not using PAT, each inside host consumes one IP address from the pool. A large or growing value in the second misses counter in the **show ip nat statistics** command output can indicate this problem. Also, compare the configured pool to the list of addresses in the NAT translation table (**show ip nat translations**). Finally, if the pool is small, the problem may be that the configuration intended to use PAT and is missing the **overload** keyword (see the next item).
- **PAT:** It is easy to forget to add the **overload** option on the end of the **ip nat inside source list** command. PAT configuration is identical to a valid dynamic NAT configuration except that PAT requires the **overload** keyword. Without it, dynamic NAT works, but the pool of addresses is typically consumed very quickly. The NAT router will not translate nor forward traffic for hosts if there is not an available pool IP address for their traffic, so some hosts experience an outage.
- **ACL:** As mentioned in Chapter 3, "Advanced IPv4 Access Control Lists," you can always add a check for ACLs that cause a problem. Perhaps NAT has been configured correctly, but an ACL exists on one of the interfaces, discarding the packets. Note that the order of operations inside the router matters in this case. For packets entering an interface, IOS processes ACLs before NAT. For packets exiting an interface, IOS processes any outbound ACL after translating the addresses with NAT.

- **User traffic required:** NAT reacts to user traffic. If you configure NAT in a lab, NAT does not act to create translations (**show ip nat translations**) until some user traffic enters the NAT router on an inside interface, triggering NAT to do a translation. The NAT configuration can be perfect, but if no inbound traffic occurs that matches the NAT configuration, NAT does nothing.
- **IPv4 routing:** IPv4 routing could prevent packets from arriving on either side of the NAT router. Note that the routing must work for the destination IP addresses used in the packets.

With source NAT, the user sits at some user device like a PC. She attempts to connect to some server, using that server's DNS name. After DNS resolution, the client (the inside host) sends an IP packet with a destination address of the server. For instance, as shown in Figure 10-11, PC1 sends an IP packet with destination IP address 170.1.1.1, some server in the Internet. PC1 is an inside host, the server is an outside host, and 170.1.1.1 is the outside global address. (Note that these addresses match the previous example, which referenced Figure 10-10.)

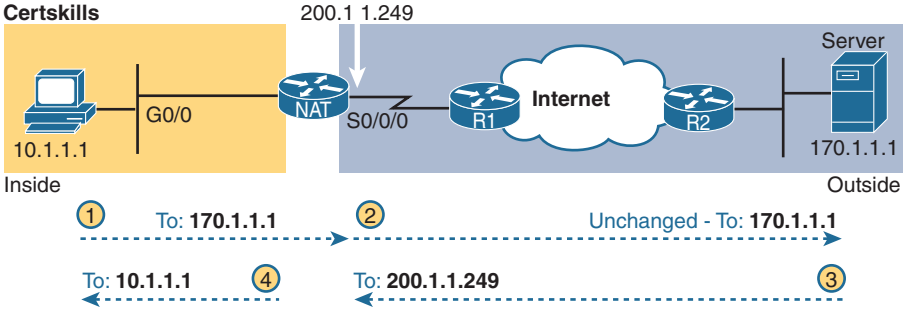


Figure 10-11 *Destination Address Changes on Outside to Inside (Only) with Source NAT*

Note that with source NAT in what should be a familiar design, the destination IP address of the packet does not change during the entire trip. So, troubleshooting of IPv4 routing toward the outside network will be based on the same IP address throughout.

Now look at steps 3 and 4 in the figure, which reminds you that the return packet will first flow to the NAT inside global address (200.1.1.249 in this case) as shown at step 3. Then NAT converts the destination address to 10.1.1.1 in this case. So, to troubleshoot packets flowing right to left in this case, you have to troubleshoot based on two different destination IP addresses.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 10-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 10-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review memory tables		Book, website
Review command tables		Book
Do labs		Blog

Review All the Key Topics

Key Topic

Table 10-6 Key Topics for Chapter 10

Key Topic Element	Description	Page Number
Table 10-2	List of private IP network numbers	206
Figure 10-2	Main concept of NAT translating private IP addresses into publicly unique global addresses	207
Figure 10-4	Typical NAT network diagram with key NAT terms listed	209
Table 10-4	List of four key NAT terms and their meanings	210
Figure 10-8	Concepts behind address conservation achieved by NAT overload (PAT)	213
Paragraph	Summary of differences between dynamic NAT configuration and PAT using a pool	220

Key Terms You Should Know

CIDR, inside global, inside local, NAT overload, outside global, Port Address Translation, private IP network, source NAT

Command References

Tables 10-7 and 10-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 10-7 Chapter 10 Configuration Command Reference

Command	Description
ip nat {**inside** | **outside**}	Interface subcommand to enable NAT and identify whether the interface is in the inside or outside of the network
ip nat inside source {**list** {*access-list-number* | *access-list-name*}} {**interface** *type number* | **pool** *pool-name*} [**overload**]	Global command that enables NAT globally, referencing the ACL that defines which source addresses to NAT, and the interface or pool from which to find global addresses
ip nat pool *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}	Global command to define a pool of NAT addresses
ip nat inside source *inside-local inside-global*	Global command that lists the inside and outside address (or, an outside interface whose IP address should be used) to be paired and added to the NAT translation table

Table 10-8 Chapter 10 EXEC Command Reference

Command	Description
show ip nat statistics	Lists counters for packets and NAT table entries, as well as basic configuration information
show ip nat translations [**verbose**]	Displays the NAT table
clear ip nat translation {* | [**inside** *global-ip local-ip*] [**outside** *local-ip global-ip*]}	Clears all or some of the dynamic entries in the NAT table, depending on which parameters are used
clear ip nat translation *protocol* **inside** *global-ip global-port local-ip local-port* [**outside** *local-ip global-ip*]	Clears some of the dynamic entries in the NAT table, depending on which parameters are used
debug ip nat	Issues a log message describing each packet whose IP address is translated with NAT

CHAPTER 11

Quality of Service (QoS)

This chapter covers the following exam topics:

4.0 IP Services

4.7 Explain the forwarding per-hop behavior (PHB) for QoS such as classification, marking, queuing, congestion, policing, shaping

Quality of Service (QoS) refers to tools that network devices can use to manage several related characteristics of what happens to a packet while it flows through a network. Specifically, these tools manage the bandwidth made available to that type of packet, the delay the packet experiences, the jitter (variation in delay) between successive packets in the same flow, and the percentage of packet loss for packets of each class. These tools balance the trade-offs of which types of traffic receive network resources and when, giving more preference to some traffic and less preference to others.

QoS tools define actions a device can apply to a message between the time it enters the device until it exits the device. QoS defines these actions as *per-hop behaviors* (PHBs), which is a formal term to refer to actions other than storing and forwarding a message. These actions can delay the message, discard it, or even change header fields. The device can choose different PHBs for different kinds of messages, improving the QoS behavior for some messages, while worsening the QoS behavior for others.

This chapter works through the QoS tools listed in the single QoS exam topic: "Explain the forwarding per-hop behavior (PHB) for QoS such as classification, marking, queuing, congestion, policing, shaping." Each topic emphasizes the problems each tool solves and how each tool manages bandwidth, delay, jitter, and loss.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 11-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Introduction to QoS	1
Classification and Marking	2, 3
Queuing	4
Shaping and Policing	5
Congestion Avoidance	6

1. Which of the following attributes do QoS tools manage? (Choose three answers.)
   a. Bandwidth
   b. Delay
   c. Load
   d. MTU
   e. Loss
2. Which of the following QoS marking fields could remain with a packet while being sent through four different routers, over different LAN and WAN links? (Choose two answers.)
   a. CoS
   b. IPP
   c. DSCP
   d. MPLS EXP
3. Which of the following are available methods of classifying packets in DiffServ on Cisco routers? (Choose three answers.)
   a. Matching the IP DSCP field
   b. Matching the 802.1p CoS field
   c. Matching fields with an extended IP ACL
   d. Matching the SNMP Location variable
4. Which of the following behaviors are applied to a low latency queue in a Cisco router or switch? (Choose two answers.)
   a. Shaping
   b. Policing
   c. Priority scheduling
   d. Round-robin scheduling
5. Think about a policing function that is currently working, and also think about a shaping function that is also currently working. That is, the current bit rate of traffic exceeds the respective policing and shaping rates. Which statements are true about these features? (Choose two answers.)
   a. The policer may or may not be discarding packets.
   b. The policer is definitely discarding packets.
   c. The shaper may or may not be queuing packets to slow down the sending rate.
   d. The shaper is definitely queuing packets to slow down the sending rate.

6. A queuing system has three queues serviced with round-robin scheduling and one low latency queue that holds all voice traffic. Round-robin queue 1 holds predominantly UDP traffic, while round-robin queues 2 and 3 hold predominantly TCP traffic. The packets in each queue happen to have a variety of DSCP markings per the QoS design. In which queues would it make sense to use a congestion avoidance (drop management) tool? (Choose two answers.)

   a. The LLQ
   b. Queue 1
   c. Queue 2
   d. Queue 3

Foundation Topics

Introduction to QoS

Routers typically sit at the WAN edge, with both WAN interfaces and LAN interfaces. Those LAN interfaces typically run at much faster speeds, while the WAN interfaces run at slower speeds. While that slower WAN interface is busy sending the packets waiting in the router, hundreds or even thousands more IP packets could arrive in the LAN interfaces, all needing to be forwarded out that same WAN interface. What should the router do? Send them all, in the same order in which they arrived? Prioritize the packets, to send some earlier than others, preferring one type of traffic over another? Discard some of the packets when the number of packets waiting to exit the router gets too large?

That first paragraph described some of the many classic Quality of Service (QoS) questions in networking. QoS refers to the tools that networking devices use to apply some different treatment to packets in the network as they pass through the device. For instance, the WAN edge router would queue packets waiting for the WAN interface to be available. The router could also use a queue scheduling algorithm to determine which packets should be sent next, using some other order than the arrival order—giving some packets better service and some worse service.

QoS: Managing Bandwidth, Delay, Jitter, and Loss

Cisco offers a wide range of QoS tools on both routers and switches. All these tools give you the means to manage four characteristics of network traffic:

- Bandwidth
- Delay
- Jitter
- Loss

Bandwidth refers to the speed of a link, in bits per second (bps). But while we think of bandwidth as speed, it helps to also think of bandwidth as the capacity of the link, in terms of how many bits can be sent over the link per second. The networking device's QoS tools determine what packet is sent over the link next, so the networking device is in control of which messages get access to the bandwidth next and how much of that bandwidth (capacity) each type of traffic gets over time.

For example, consider that typical WAN edge router that has hundreds of packets waiting to exit the WAN link. An engineer might configure a queuing tool to reserve 10 percent of the bandwidth for voice traffic, 50 percent for mission-critical data applications, and leave the rest of the bandwidth for all other types of traffic. The queuing tool could then use those settings to make the choice about which packets to send next.

Delay can be described as one-way delay or round-trip delay. *One-way delay* refers to the time between sending one packet and that same packet arriving at the destination host. *Round-trip delay* counts the one-way delay plus the time for the receiver of the first packet to send back a packet—in other words, the time it takes to send one packet between two hosts and receive one back. Many different individual actions impact delay; this chapter will discuss a few of those, including queuing and shaping delay.

Jitter refers to the variation in one-way delay between consecutive packets sent by the same application. For example, imagine an application sends a few hundred packets to one particular host. The first packet's one-way delay is 300 milliseconds (300 ms, or .3 seconds). The next packet's one-way delay is 300 ms; so is the third's; and so on. In that case, there is no jitter. However, if instead the first packet has a one-way delay of 300 ms, the next has a one-way delay of 310 ms, and the next has 325 ms, then there is some variation in the delay; 10 ms between packets 1 and 2, and another 15 ms between packets 2 and 3. That difference is called jitter.

Finally, *loss* refers to the number of lost messages, usually as a percentage of packets sent. The comparison is simple: if the sender for some application sends 100 packets, and only 98 arrive at the destination, that particular application flow experienced 2 percent loss. Loss can be caused by many factors, but often, people think of loss as something caused by faulty cabling or poor WAN services. That is one cause. However, more loss happens because of the normal operation of the networking devices, in which the devices' queues get too full, so the device has nowhere to put new packets, and it discards the packet. Several QoS tools manage queuing systems to help control and avoid loss.

Types of Traffic

With QoS, a network engineer sets about to prefer one type of traffic over another in regard to bandwidth, delay, jitter, and loss. Sometimes, that choice relates to the specific business. For example, if all the mission-critical applications sit on servers in three known subnets, then the QoS plan could be set up to match packets going to/from that subnet and give that traffic better treatment compared to other traffic. However, in other cases, the choice of how to apply QoS tools relates to the nature of different kinds of applications. Some applications have different QoS needs than others. This next topic compares the basic differences in QoS needs based on the type of traffic.

Data Applications

First, consider a basic web application, with a user at a PC or tablet. The user types in a URI to request a web page. That request may require a single packet going to the web server, but it may result in hundreds or thousands of packets coming back to the web client, as shown in Figure 11-1.

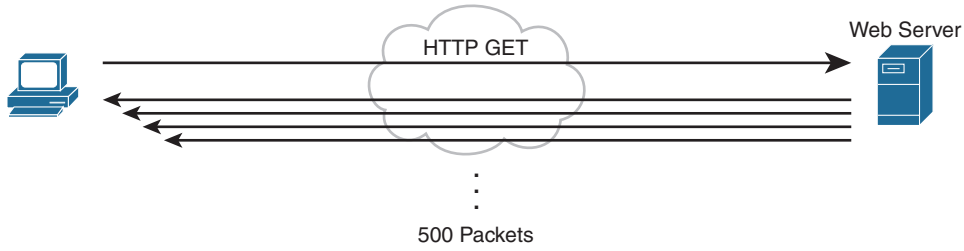


Figure 11-1 *Concept of Disproportionate Packet/Byte Volumes with HTTP Traffic*

NOTE If you wonder how one web page might require thousands of packets, consider this math: with a 1500-byte IP maximum transmission unit (MTU), the data part of a TCP segment could be at most 1460 bytes (1500 bytes minus 20 bytes each for the IP and TCP header). In this example, 1000 such packets total to 1,460,000 bytes, or about 1.5 MB. It is easy to imagine a web page with just a few graphics that totals more than 1.5 MB in size.

So, what is the impact of bandwidth, delay, jitter, and loss on an interactive web-based application? First, the packets require a certain amount of bandwidth capacity. As for delay, each of those packets from the server to the client takes some amount of one-way delay, with some jitter as well. Of the 500 packets shown in Figure 11-1, if some are lost (transmission errors, discarded by devices, or other reasons), then the server's TCP logic will retransmit, but parts of the web page may not show up right away.

While QoS tools focus on managing bandwidth, delay, jitter, and loss, the user mainly cares about the quality of the overall experience. For instance, with a web application, how long after clicking do you see something useful in your web browser? So, as a user, you care about the *Quality of Experience* (QoE), which is a term referring to users' perception of their use of the application on the network. QoS tools directly impact bandwidth, delay, jitter, and loss, which then should have some overall good effect to influence the users' QoE. And you can use QoS tools to create a better QoE for more important traffic; for instance, you might give certain business-critical applications better QoS treatment, which improves QoE for users of those apps.

In contrast, a noninteractive data application (historically called *batch* traffic)—for instance, data backup or file transfers—has different QoS requirements than interactive data applications. Batch applications typically send more data than interactive applications, but because no one is sitting there waiting to see something pop on the screen, the delay and jitter do not matter much. Much more important for these applications is meeting the need to complete the larger task (transferring files) within a larger time window. QoS tools can be used to provide enough bandwidth to meet the capacity needs of these applications and manage loss to reduce the number of retransmissions.

Voice and Video Applications

Voice and video applications each have a similar breakdown of interactive and noninteractive flows. To make the main points about both voice and video, this section looks more deeply at voice traffic.

Answers to the "Do I Know This Already?" quiz:

1 A, B, E **2** B, C **3** A, B, C **4** B, C **5** A, D **6** C, D

Before looking at voice, though, first think about the use of the term *flow* in networking. A flow is all the data moving from one application to another over the network, with one flow for each direction. For example, if you open a website and connect to a web server, the web page content that moves from the server to the client is one flow. Listen to some music with a music app on your phone, and that creates a flow from your app to the music app's server and a flow from the server back to your phone. From a voice perspective, a phone call between two IP phones would create a flow for each direction. For video, it could be the traffic from one video surveillance camera collected by security software.

Now on to voice, specifically Voice over IP (VoIP). VoIP defines the means to take the sound made at one telephone and send it inside IP packets over an IP network, playing the sound back on the other telephone. Figure 11-2 shows the general idea. The steps in the figure include

Step 1. The phone user makes a phone call and begins speaking.

Step 2. A chip called a *codec* processes (digitizes) the sound to create a binary code (160 bytes with the G.711 codec, for example) for a certain time period (usually 20 ms).

Step 3. The phone places the data into an IP packet.

Step 4. The phone sends the packet to the destination IP phone.

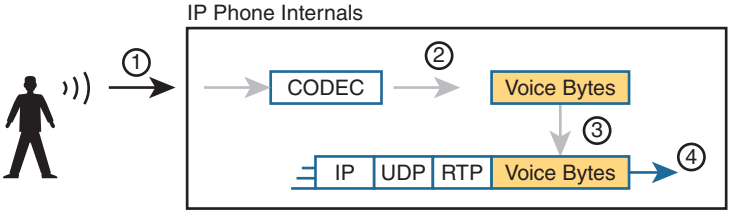


Figure 11-2 *Creating VoIP Packets with an IP Phone and a G.711 Codec*

If you work through the math a bit, this single call, with the G.711 codec, requires about 80 Kbps of bandwidth (ignoring the data-link header and trailer overhead). Counting the headers and VoIP payload as shown in the figure, each of the IP packets has 200 bytes. Each holds 20 ms of digitized voice, so the phone sends 50 packets per second. These 50 packets at 200 bytes each equal 10,000 bytes per second, or 80,000 bits per second, which is 80 Kbps. Other voice codecs require even less bandwidth, with the commonly used G.729 taking about 24 Kbps (again ignoring data-link overhead).

At first, it may look like VoIP calls require little in regard to QoS. For bandwidth, a single voice call or flow requires only a little bandwidth in comparison to many data applications. However, interactive voice does require a much better level of quality for delay, jitter, and loss.

For instance, think about making a phone call with high one-way delay. You finish speaking and pause for the other person to respond. And he does not, so you speak again—and hear the other person's voice overlaid on your own. The problem: too much delay. Or, consider calls for which the sound breaks up. The problem? It could have been packet loss, or it could have been jitter.

You can achieve good-quality voice traffic over an IP network, but you must implement QoS to do so. QoS tools set about to give different types of traffic the QoS behavior they need. Cisco's *Enterprise QoS Solution Reference Network Design Guide*, which itself quotes other sources in addition to relying on Cisco's long experience in implementing QoS, suggests the following guidelines for interactive voice:

Key Topic

- **Delay (one-way):** 150 ms or less
- **Jitter:** 30 ms or less
- **Loss:** 1% or less

In comparison, interactive voice requires more attention than interactive data applications for QoS features. Data applications generally tolerate more delay, jitter, and loss than voice (and video). A single voice call does generally take less bandwidth than a typical data application, but that bandwidth requirement is consistent. Data applications tend to be bursty, with data bursts in reaction to the user doing something with the application.

Video has a much more varied set of QoS requirements. Generally, think of video like voice, but with a much higher bandwidth requirement than voice (per flow) and similar requirements for low delay, jitter, and loss. As for bandwidth, video can use a variety of codecs that impact the amount of data sent, but many other technical features impact the amount of bandwidth required for a single video flow. (For instance, a sporting event with lots of movement on screen takes more bandwidth than a news anchor reading the news in front of a solid background with little movement.) This time quoting from *End-to-End QoS Network Design*, Second Edition (Cisco Press, 2013), some requirements for video include

Key Topic

- **Bandwidth:** 384 Kbps to 20+ Mbps
- **Delay (one-way):** 200–400 ms
- **Jitter:** 30–50 ms
- **Loss:** 0.1%–1%

NOTE *End-to-End QoS Network Design* is written by some of the same people who created the Cisco *Enterprise QoS Solution Reference Network Design Guide* (available at Cisco.com). If you are looking for a book to dig into more depth on QoS, this book is an excellent reference for Cisco QoS.

QoS as Mentioned in This Book

QoS tools change the QoS characteristics of certain flows in the network. The rest of the chapter focuses on the specific tools mentioned in the lone CCNA 200-301 exam topic about QoS, presented in the following major sections:

- "Classification and Marking" is about the marking of packets and the definition of trust boundaries.
- "Queuing" describes the scheduling of packets to give one type of packet priority over another.
- "Shaping and Policing" explains these two tools together because they are often used on opposite ends of a link.
- "Congestion Avoidance" addresses how to manage the packet loss that occurs when network devices get too busy.

QoS on Switches and Routers

Before moving on to several sections of the chapter about specific QoS tools, let me make a point about the terms *packet* and *frame* as used in this chapter.

The QoS tools discussed in this chapter can be used on both switches and routers. There are some differences in the features and differences in implementation, due to the differences of internal architecture between routers and switches. However, to the depth discussed here, the descriptions apply equally to both LAN switches and IP routers.

This chapter uses the word *packet* in a general way, to refer to any message being processed by a networking device, just for convenience. Normally, the term *packet* refers to the IP header and encapsulated headers and data, but without the data-link header and trailer. The term *frame* refers to the data-link header/trailer with its encapsulated headers and data. For this chapter, those differences do not matter to the discussion, but at the same time, the discussion often shows a message that sometimes is literally a packet (without the data-link header/trailer) and sometimes a frame.

Throughout the chapter, the text uses *packet* for all messages, because the fact of whether or not the message happens to have a data-link header/trailer at that point is immaterial to the basic discussion of features.

Additionally, note that all the examples in the chapter refer to routers, just to be consistent.

Classification and Marking

The first QoS tool discussed in this chapter, classification and marking, or simply marking, refers to a type of QoS tool that classifies packets based on their header contents, and then marks the message by changing some bits in specific header fields. This section looks first at the role of classification across all QoS tools, and then it examines the marking feature.

Classification Basics

QoS tools sit in the path that packets take when being forwarded through a router or switch, much like the positioning of ACLs. Like ACLs, QoS tools are enabled on an interface. Also like ACLs, QoS tools are enabled for a direction: packets entering the interface (before the forwarding decision) or for messages exiting the interface (after the forwarding decision).

The term *classification* refers to the process of matching the fields in a message to make a choice to take some QoS action. So, again comparing QoS tools to ACLs, ACLs perform classification and filtering; that is, ACLs match (classify) packet headers. ACLs can have the purpose (action) of choosing which packets to discard. QoS tools perform classification (matching of header fields) to decide which packets to take certain QoS actions against. Those actions include the other types of QoS tools discussed in this chapter, such as queuing, shaping, policing, and so on.

For example, consider the internal processing done by a router as shown in Figure 11-3. In this case, an output queuing tool has been enabled on an interface. Routers use queuing tools to place some packets in one output queue, other packets in another, and so on, when the outgoing interface happens to be busy. Then, when the outgoing interface becomes available to send another message, the queuing tool's scheduler algorithm can pick the next message from any one of the queues, prioritizing traffic based on the rules configured by the network engineer.

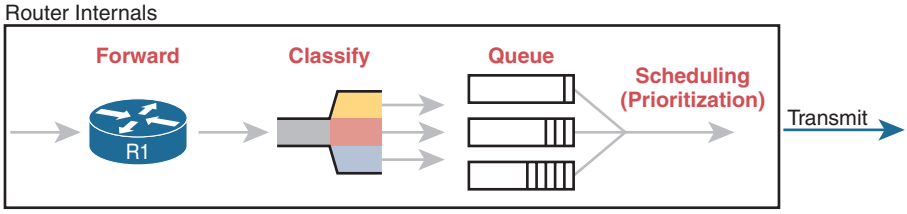


Figure 11-3 *Big Idea: Classification for Queuing in a Router*

The figure shows the internals of a router and what happens to the packet during part of that internal processing, moving left to right inside the router, as follows:

Step 1. The router makes a forwarding (routing) decision.

Step 2. The output queuing tool uses classification logic to determine which packets go into which output queue.

Step 3. The router holds the packets in the output queue waiting for the outgoing interface to be available to send the next message.

Step 4. The queuing tool's scheduling logic chooses the next packet, effectively prioritizing one packet over another.

While the example shows a queuing tool, note that the queuing tool requires the ability to classify messages by comparing the messages to the configuration, much like ACLs.

Matching (Classification) Basics

Now think about classification from an enterprise-wide perspective, which helps us appreciate the need for marking. Every QoS tool can examine various headers to make comparisons to classify packets. However, you might apply QoS tools on most every device in the network, sometimes at both ingress and egress on most of the interfaces. Using complex matching of many header fields in every device and on most interfaces requires lots of configuration. The work to match packets can even degrade device performance of some devices. So, while you could have every device use complex packet matching, doing so is a poor strategy.

A better strategy, one recommended both by Cisco and by RFCs, suggests doing complex matching early in the life of a packet and then marking the packet. *Marking* means that the QoS tool changes one or more header fields, setting a value in the header. Several header fields have been designed for the purpose of marking the packets for QoS processing. Then, devices that process the packet later in its life can use much simpler classification logic.

Figure 11-4 shows an example, with a PC on the left sending an IP packet to some host off the right side of the figure (not shown). Switch SW1, the first networking device to forward the packet, does some complex comparisons and marks the packet's Differentiated Services Code Point (DSCP) field, a 6-bit field in the IP header meant for QoS marking. The next three devices that process this message—SW2, R1, and R2—then use simpler matching to classify the packet by comparing the packet's DSCP value, placing packets with one DSCP value in class 1, and packets with another DSCP value in class 2.

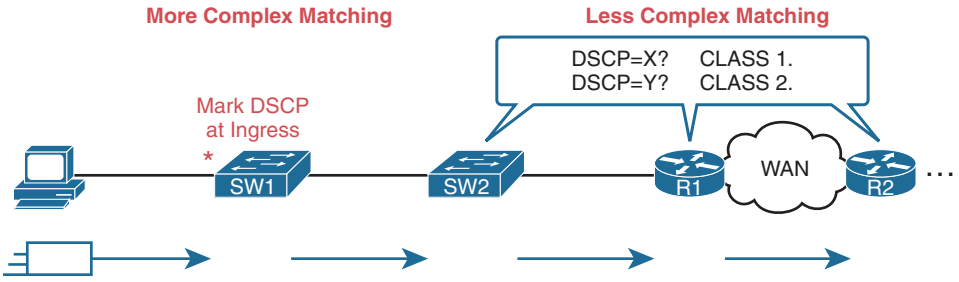


Figure 11-4 *Systematic Classification and Marking for the Enterprise*

Classification on Routers with ACLs and NBAR

Now that you know the basics of what classification and marking do together, this section takes the discussion a little deeper with a closer look at classification on routers, which is followed by a closer look at the marking function.

First, QoS classification sounds a lot like what ACLs do, and it should. In fact, many QoS tools support the ability to simply refer to an IP ACL, with this kind of logic:

> For any packet matched by the ACL with a permit action, consider that packet a match for QoS, so do a particular QoS action.

As a reminder, Figure 11-5 shows the IP and TCP header. All these fields are matchable for QoS classification.

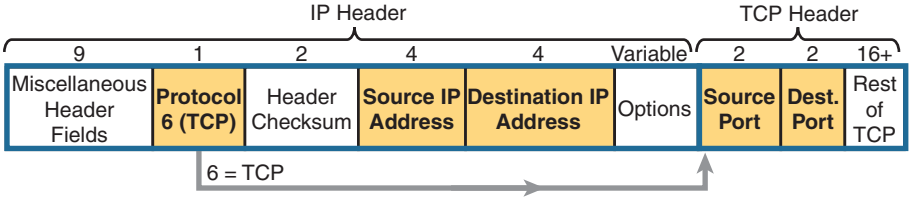


Figure 11-5 *Classification with Five Fields Used by Extended ACLs*

Now think about the enterprise's QoS plan for a moment. That plan should list details such as which types of traffic should be classified as being in the same class for queuing purposes, for shaping, and for any other QoS tool. That plan should detail the fields in the header that can be matched. For instance, if all the IP phones sit in subnets within the range of addresses 10.3.0.0/16, then the QoS plan should state that. Then the network engineer could configure an extended ACL to match all packets to/from IP addresses inside 10.3.0.0/16 and apply appropriate QoS actions to that voice traffic.

However, not every classification can be easily made by matching with an ACL. In more challenging cases, Cisco Network Based Application Recognition (NBAR) can be used. NBAR is basically in its second major version, called NBAR2, or next-generation NBAR. In short, NBAR2 matches packets for classification in a large variety of ways that are very useful for QoS.

NBAR2 looks far beyond what an ACL can examine in a message. Many applications cannot be identified based on well-known port alone. NBAR solves those problems.

Cisco also organizes what NBAR can match in ways that make it easy to separate the traffic into different classes. For instance, the Cisco WebEx application provides audio and video conferencing on the web. In a QoS plan, you might want to classify WebEx differently than other video traffic and classify it differently than voice calls between IP phones. That is, you might classify WebEx traffic and give it a unique DSCP marking. NBAR provides easy built-in matching ability for WebEx, plus more than 1000 different subcategories of applications.

Just to drive the point home with NBAR, Example 11-1 lists four lines of help output for one of many NBAR configuration commands. I chose a variety of items that might be more memorable. With the use of the keywords on the left in the correct configuration command, you could match the following: entertainment video from Amazon, video from Cisco's video surveillance camera products, voice from Cisco IP Phones, and video from sports channel ESPN. (NBAR refers to this idea of defining the characteristics of different applications as *application signatures*.)

Example 11-1 *Example of the Many NBAR2 Matchable Applications*

```
R1#(config)# class-map matchingexample
R1(config-cmap)# match protocol attribute category voice-and-video ?
! output heavily edited for length
  amazon-instant-video      VOD service by Amazon
  cisco-ip-camera           Cisco video surveillance camera
  cisco-phone               Cisco IP Phones and PC-based Unified Communicators
  espn-video                ESPN related websites and mobile applications video
  facetime                  Facetime video calling software
! Output snipped.
```

To wrap up the discussion of NBAR for classification, compare the first two highlighted entries in the output. Without NBAR, it would be difficult to classify an entertainment video from Amazon versus the video from a security camera, but those two highlighted entries show how you easily have classified that traffic differently. The third highlighted item shows how to match traffic for Cisco IP Phones (and PC-based equivalents), again making for an easier match of packets of a particular type.

Marking IP DSCP and Ethernet CoS

The QoS plan for an enterprise centers on creating classes of traffic that should receive certain types of QoS treatment. That plan would note how to classify packets into each classification and the values that should be marked on the packets, basically labeling each packet with a number to associate it with that class. For example, that plan might state the following:

- Classify all voice payload traffic that is used for business purposes as IP DSCP EF and CoS 5.
- Classify all video conferencing and other interactive video for business purposes as IP DSCP AF41 and CoS 4.
- Classify all business-critical data application traffic as IP DSCP AF21 and CoS 2.

This next topic takes a closer look at the specific fields that can be marked, defining the DSCP and CoS marking fields.

Marking the IP Header

Marking a QoS field in the IP header works well with QoS because the IP header exists for the entire trip from the source host to the destination host. When a host sends data, the host sends a data-link frame that encapsulates an IP packet. Each router that forwards the IP packet discards the old data-link header and adds a new header. Because the routers do not discard and reinsert IP headers, marking fields in the IP header stay with the data from the first place it is marked until it reaches the destination host.

IPv4 defines a Type of Service (ToS) byte in the IPv4 header, as shown in Figure 11-6. The original RFC defined a 3-bit IP Precedence (IPP) field for QoS marking. That field gave us eight separate values—binary 000, 001, 010, and so on, through 111—which when converted to decimal are decimals 0 through 7.

Key Topic

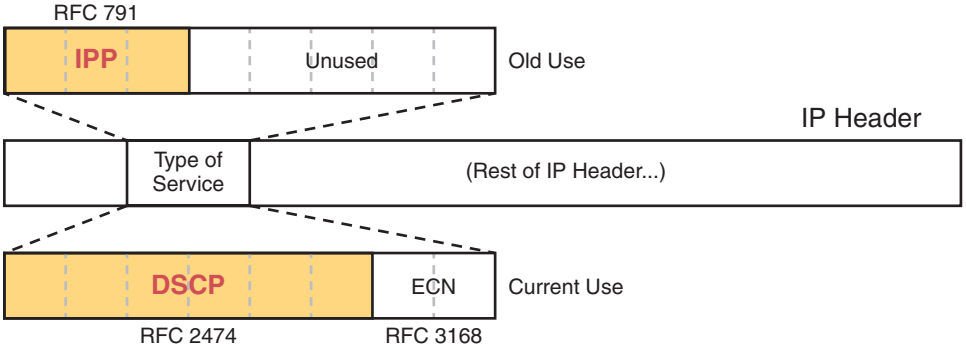


Figure 11-6 *IP Precedence and Differentiated Services Code Point Fields*

NOTE Those last 5 bits of the ToS byte per RFC 791 were mostly defined for some purpose but were not used in practice to any significant extent.

While a great idea, IPP gave us only eight different values to mark, so later RFCs redefined the ToS byte with the DSCP field. DSCP increased the number of marking bits to 6 bits, allowing for 64 unique values that can be marked. The DiffServ RFCs, which became RFCs back in the late 1990s, have become accepted as the most common method to use when doing QoS, and using the DSCP field for marking has become quite common.

IPv6 has a similar field to mark as well. The 6-bit field also goes by the name DSCP, with the byte in the IPv6 header being called the IPv6 *Traffic Class* byte. Otherwise, think of IPv4 and IPv6 being equivalent in terms of marking.

IPP and DSCP fields can be referenced by their decimal values as well as some convenient text names. The later section titled "DiffServ Suggested Marking Values" details some of the names.

Marking the Ethernet 802.1Q Header

Another useful marking field exists in the 802.1Q header, in a field originally defined by the IEEE 802.1p standard. This field sits in the third byte of the 4-byte 802.1Q header, as a 3-bit field, supplying eight possible values to mark (see Figure 11-7). It goes by two different names: *Class of Service*, or CoS, and *Priority Code Point*, or PCP.

Key Topic

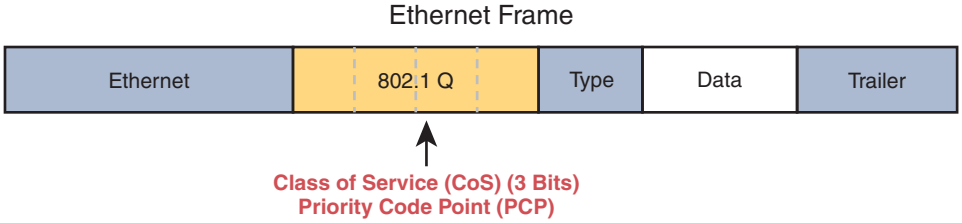


Figure 11-7 *Class of Service Field in 802.1Q/p Header*

The figure uses two slightly different shades of gray (in print) for the Ethernet header and trailer fields versus the 802.1Q header, as a reminder: the 802.1Q header is not included in all Ethernet frames. The 802.1Q header only exists when 802.1Q trunking is used on a link. As a result, QoS tools can make use of the CoS field only for QoS features enabled on interfaces that use trunking, as shown in Figure 11-8.

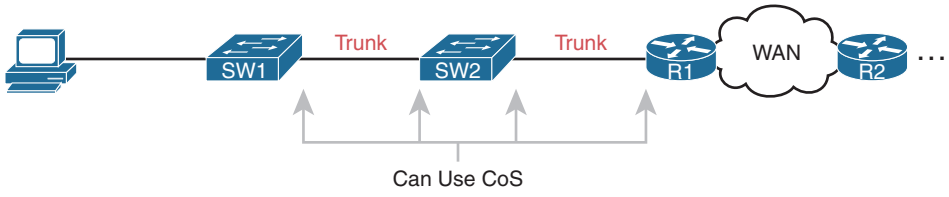


Figure 11-8 *Useful Life of CoS Marking*

For instance, if the PC on the left were to send data to a server somewhere off the figure to the right, the DSCP field would exist for that entire trip. However, the CoS field would exist over the two trunks only and would be useful mainly on the four interfaces noted with the arrow lines.

Other Marking Fields

Other marking fields also exist in other headers. Table 11-2 lists those fields for reference.

Table 11-2 Marking Fields

Field Name	Header(s)	Length (bits)	Where Used
DSCP	IPv4, IPv6	6	End-to-end packet
IPP	IPv4, IPv6	3	End-to-end packet
CoS	802.1Q	3	Over VLAN trunk
TID	802.11	3	Over Wi-Fi
EXP	MPLS Label	3	Over MPLS WAN

Defining Trust Boundaries

The end-user device can mark the DSCP field—and even the CoS field if trunking is used on the link. Would you, as the network engineer, trust those settings and let your networking devices trust and react to those markings for their various QoS actions?

Most of us would not, because anything the end user controls might be used inappropriately at times. For instance, a PC user could know enough about DiffServ and DSCPs to know that most voice traffic is marked with a DSCP called Expedited Forwarding (EF), which has a decimal value of 46. Voice traffic gets great QoS treatment, so PC users could mark all their traffic as DSCP 46, hoping to get great QoS treatment.

The people creating a QoS plan for an enterprise have to choose where to place the trust boundary for the network. The *trust boundary* refers to the point in the path of a packet flowing through the network at which the networking devices can trust the current QoS markings. That boundary typically sits in a device under the control of the IT staff.

For instance, a typical trust boundary could be set in the middle of the first ingress switch in the network, as shown in Figure 11-9. The markings on the message as sent by the PC cannot be trusted. However, because SW1 performed classification and marking as the packets entered the switch, the markings can be trusted at that point.

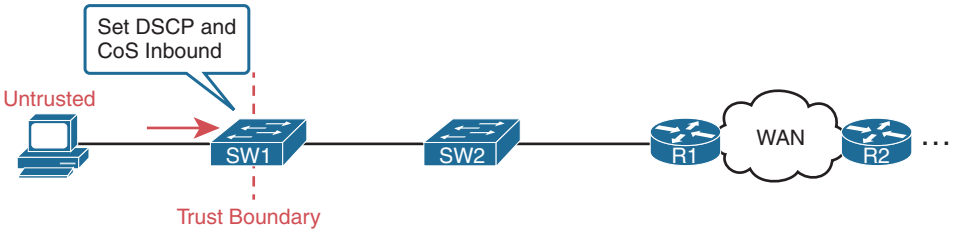


Figure 11-9 *Trusting Devices—PC*

Interestingly, when the access layer includes an IP Phone, the phone is typically the trust boundary, instead of the access layer switch. IP Phones can set the CoS and DSCP fields of the messages created by the phone, as well as those forwarded from the PC through the phone. The specific marking values are actually configured on the attached access switch. Figure 11-10 shows the typical trust boundary in this case, with notation of what the phone's marking logic usually is: mark all of the PC's traffic with a particular DSCP and/or CoS, and the phone's traffic with different values.

Key Topic

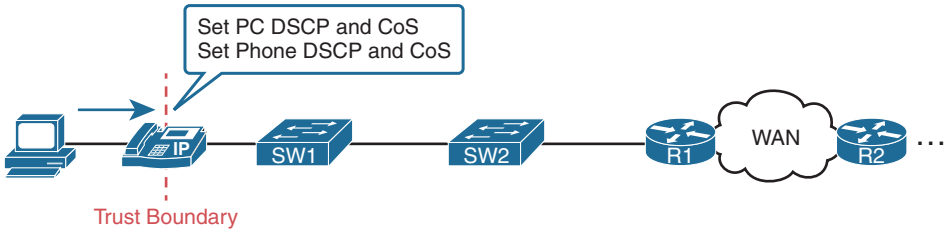


Figure 11-10 *Trusting Devices—IP Phone*

DiffServ Suggested Marking Values

Everything in this chapter follows the DiffServ architecture as defined originally by RFC 2475, plus many other DiffServ RFCs. In particular, DiffServ goes beyond theory in several areas, including making suggestions about the specific DSCP values to use when marking IP packets. By suggesting specific markings for specific types of traffic, DiffServ hoped to create a consistent use of DSCP values in all networks. By doing so, product vendors could provide good default settings for their QoS features, QoS could work better between an enterprise and service provider, and many other benefits could be realized.

The next two topics outline three sets of DSCP values as used in DiffServ.

Expedited Forwarding (EF)

DiffServ defines the *Expedited Forwarding* (EF) DSCP value—a single value—as suggested for use for packets that need low latency (delay), low jitter, and low loss. The Expedited Forwarding RFC (RFC 3246) defines the specific DSCP value (decimal 46) and an equivalent text name (Expedited Forwarding). QoS configuration commands allow the use of the decimal value or text name, but one purpose of having a text acronym to use is to make the value more memorable, so many QoS configurations refer to the text names.

Most often QoS plans use EF to mark voice payload packets. With voice calls, some packets carry voice payload, and other packets carry call signaling messages. Call signaling messages set up (create) the voice call between two devices, and they do not require low delay, jitter, and loss. Voice payload packets carry the digitized voice, as shown back in Figure 11-2, and these packets do need better QoS. By default, Cisco IP Phones mark voice payload with EF, and mark voice signaling packets sent by the phone with another value called CS3.

Assured Forwarding (AF)

The Assured Forwarding (AF) DiffServ RFC (2597) defines a set of 12 DSCP values meant to be used in concert with each other. First, it defines the concept of four separate queues in a queuing system. Additionally, it defines three levels of drop priority within each queue for use with congestion avoidance tools. With four queues, and three drop priority classes per queue, you need 12 different DSCP markings, one for each combination of queue and drop priority. (Queuing and congestion avoidance mechanisms are discussed later in this chapter.)

Assured Forwarding defines the specific AF DSCP text names and equivalent decimal values as listed in Figure 11-11. The text names follow a format of AFXY, with X referring to the queue (1 through 4) and Y referring to the drop priority (1 through 3).

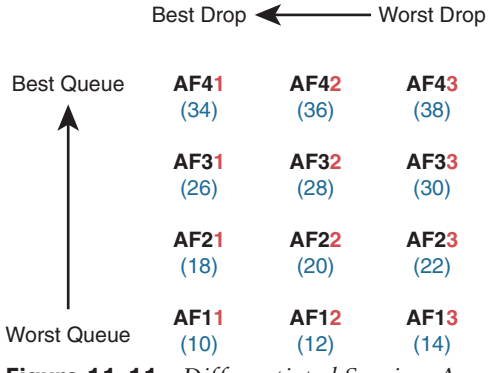


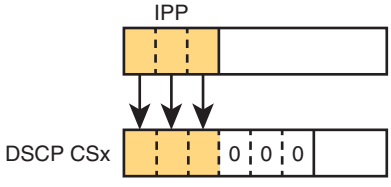
Figure 11-11 *Differentiated Services Assured Forwarding Values and Meaning*

For example, if you marked packets with all 12 values, those with AF11, AF12, and AF13 would all go into one queue; those with AF21, AF22, and AF23 would go into another queue; and so on. Inside the queue with all the AF2y traffic, you would treat the AF21, AF22, and AF23 each differently in regard to drop actions (congestion avoidance), with AF21 getting the preferred treatment and AF23 the worst treatment.

Class Selector (CS)

Originally, the ToS byte was defined with a 3-bit IP Precedence (IPP) field. When DiffServ redefined the ToS byte, it made sense to create eight DSCP values for backward compatibility with IPP values. The Class Selector (CS) DSCP values are those settings.

Figure 11-12 shows the main idea along with the eight CS values, both in name and in decimal value. Basically, the DSCP values have the same first 3 bits as the IPP field, and with binary 0s for the last 3 bits, as shown on the left side of the figure. CSx represents the text names, where x is the matching IPP value (0 through 7).



IPP	CS	Decimal DSCP
0	CS0	0
1	CS1	8
2	CS2	16
3	CS3	24
4	CS4	32
5	CS5	40
6	CS6	48
7	CS7	56

Figure 11-12 *Class Selector*

This section on classification and marking has provided a solid foundation for understanding the tools explored in the next three major sections of this chapter: queuing, shaping/policing, and congestion avoidance.

Guidelines for DSCP Marking Values

Even with this introduction to the various DSCP marking values, you could imagine that an enterprise needs to follow a convention for how to use the markings. With so many different values, having different uses of different DSCP values by different devices in the same enterprise would make deploying QoS quite difficult at best.

Among its many efforts to standardize QoS, Cisco helped to develop RFC 4954, an RFC that defines several conventions for how to use the DSCP field. The RFC provides alternative plans with different levels of detail. Each plan defines a type of traffic and the DSCP value to use when marking data. Without getting into the depth of any one plan, the plans all specify some variation for how all devices should mark data as follows:

- DSCP EF: Voice payload
- AF4x: Interactive video (for example, videoconferencing)
- AF3x: Streaming video
- AF2x: High priority (low latency) data
- CS0: Standard data

Cisco not only worked to develop the RFC standards but also uses those standards. Cisco uses default marking conventions based on the marking data in RFC 4594, with some small exceptions. If you want to read more about these QoS marking plans, refer to a couple of sources. First, look for the Cisco QoS Design Guides at Cisco.com. Also refer to RFC 4594.

Queuing

All networking devices use queues. Network devices receive messages, make a forwarding decision, and then send the message—but sometimes the outgoing interface is busy. So, the device keeps the outgoing message in a queue, waiting for the outgoing interface to be available—simple enough.

The term *queuing* refers to the QoS toolset for managing the queues that hold packets while they wait their turn to exit an interface (and in other cases in which a router holds packets waiting for some resource). But queuing refers to more than one idea, so you have to look inside devices to think about how they work. For instance, consider Figure 11-13, which shows the internals of a router. The router, of course, makes a forwarding decision, and it needs to be ready to queue packets for transmission once the outgoing interface is available. At the same time, the router may take a variety of other actions as well—ingress ACL, ingress NAT (on the inside interface), egress ACLs after the forwarding decision is made, and so on.

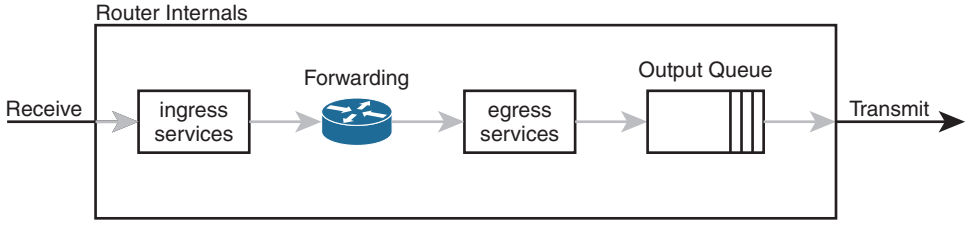


Figure 11-13 *Output Queuing in a Router: Last Output Action Before Transmission*

The figure shows *output queuing* in which the device holds messages until the output interface is available. The queuing system may use a single output queue, with a first-in, first-out (FIFO) scheduler. (In other words, it's like ordering lunch at the sandwich shop that has a single ordering line.)

Next, think a little more deeply about the queuing system. Most networking devices can have a queuing system with multiple queues. To use multiple queues, the queuing system needs a classifier function to choose which packets are placed into which queue. (The classifier can react to previously marked values or do a more extensive match.) The queuing system needs a scheduler as well, to decide which message to take next when the interface becomes available, as shown in Figure 11-14.

Key Topic

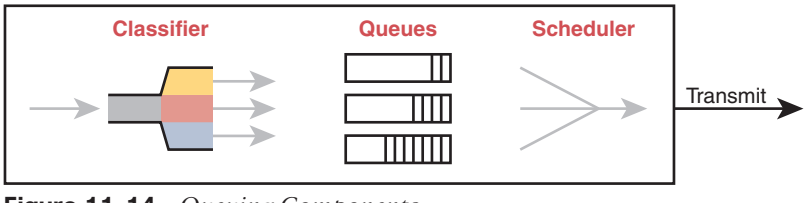


Figure 11-14 *Queuing Components*

Of all these components of the queuing system, the scheduler can be the most interesting part because it can perform prioritization. *Prioritization* refers to the concept of giving priority to one queue over another in some way.

Round-Robin Scheduling (Prioritization)

One scheduling algorithm used by Cisco routers and switches uses round-robin logic. In its most basic form, round robin cycles through the queues in order, taking turns with each queue. In each cycle, the scheduler either takes one message or takes a number of bytes from each queue by taking enough messages to total that number of bytes. Take some messages from queue 1, move on and take some from queue 2, then take some from queue 3, and so on, starting back at queue 1 after finishing a complete pass through the queues.

Round-robin scheduling also includes the concept of *weighting* (generally called *weighted round robin*). Basically, the scheduler takes a different number of packets (or bytes) from each queue, giving more preference to one queue over another.

For example, routers use a popular tool called *Class-Based Weighted Fair Queuing* (CBWFQ) to guarantee a minimum amount of bandwidth to each class. That is, each class receives at least the amount of bandwidth configured during times of congestion, but maybe more. Internally, CBWFQ uses a weighted round-robin scheduling algorithm, while letting the network engineer define the weightings as a percentage of link bandwidth. Figure 11-15 shows an example in which the three queues in the system have been given 20, 30, and 50 percent of the bandwidth each, respectively.

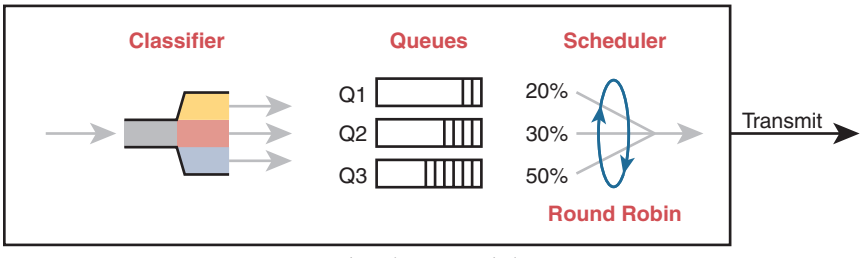


Figure 11-15 *CBWFQ Round-Robin Scheduling*

With the queuing system shown in the figure, if the outgoing link is congested, the scheduler guarantees the percentage bandwidth shown in the figure to each queue. That is, queue 1 gets 20 percent of the link even during busy times.

Low Latency Queuing

Earlier in the chapter, the section titled "Voice and Video Applications" discussed the reasons why voice and video, particularly interactive voice and video like phone calls and videoconferencing, need low latency (low delay), low jitter, and low loss. Unfortunately, a round-robin scheduler does not provide low enough delay, jitter, or loss. The solution: add Low Latency Queuing (LLQ) to the scheduler.

First, for a quick review, Table 11-3 lists the QoS requirements for a voice call. The numbers come from the *Enterprise QoS Solution Reference Network Design Guide*, referenced earlier in the chapter. The amount of bandwidth required per call varies based on the codec used by the call. However, the delay, jitter, and loss requirements remain the same for all voice calls. (Interactive video has similar requirements for delay, jitter, and loss.)

Table 11-3 QoS Requirements for a VoIP Call per Cisco Voice Design Guide

Bandwidth/call	One-way Delay (max)	Jitter (max)	Loss (max)
30–320 Kbps	150 ms	30 ms	<1%

A round-robin queuing system adds too much delay for these voice and video packets. To see why, imagine a voice packet arrives and is routed to be sent out some interface with the queuing system shown in Figure 11-16. However, that next voice packet arrives just as the round-robin scheduler moves on to service the queue labeled "data 1." Even though the voice queue has been given 50 percent of the link bandwidth, the scheduler does not send that voice message until it sends some messages from the other three queues—adding delay and jitter.

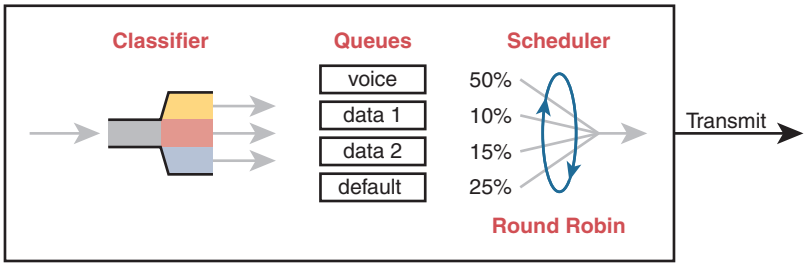


Figure 11-16 *Round Robin Not Good for Voice Delay (Latency) and Jitter*

The solution, LLQ, tells the scheduler to treat one or more queues as special *priority queues*. The LLQ scheduler always takes the next message from one of these special priority queues. Problem solved: very little delay for packets in that queue, resulting in very little jitter as well. Plus the queue never has time to fill up, so there are no drops due to the queue filling up. Figure 11-17 shows the addition of the LLQ logic for the voice queue.

Key Topic

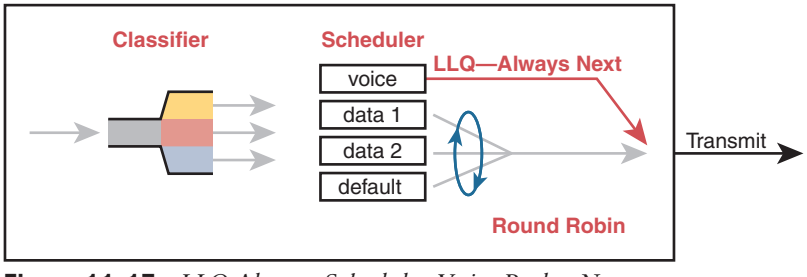


Figure 11-17 *LLQ Always Schedules Voice Packet Next*

Using LLQ, or a priority queue, provides the needed low delay, jitter, and loss for the traffic in that queue. However, think about those other queues. Do you see the problem? What happens if the speed of the interface is X bits per second, but more than X bits per second come into the voice queue? The scheduler never services the other queues (called *queue starvation*).

As you might guess, there is a solution: limit the amount of traffic placed into the priority queue, using a feature called policing. The next section talks about policers in more detail, but for now, think of it as a cap on the bandwidth used by the priority queue. For instance, you could reserve 20 percent of the link's bandwidth for the voice queue and make it a priority queue. However, in this case, instead of 20 percent being the minimum bandwidth, it is the maximum for that queue. If more than 20 percent of the link's worth of bits shows up in that queue, the router will discard the excess.

Limiting the amount of bandwidth in the priority queue protects the other queues, but it causes yet another problem. Voice and video need low loss, and with LLQ, we put the voice

and video into a priority queue that will discard the excess messages beyond the bandwidth limit. The solution? Find a way to limit the amount of voice and video that the network routes out this link, so that the policer never discards any of the traffic. There are QoS tools to help you do just that, called Call Admission Control (CAC) tools. However, CAC tools did not get a mention in the exam topics, so this chapter leaves those tools at a brief mention.

A Prioritization Strategy for Data, Voice, and Video

This section about queuing introduces several connected ideas, so before leaving the discussion of queuing, think about this strategy for how most enterprises approach queuing in their QoS plans:

Key Topic

1. Use a round-robin queuing method like CBWFQ for data classes and for noninteractive voice and video.
2. If faced with too little bandwidth compared to the typical amount of traffic, give data classes that support business-critical applications much more guaranteed bandwidth than is given to less important data classes.
3. Use a priority queue with LLQ scheduling for interactive voice and video, to achieve low delay, jitter, and loss.
4. Put voice in a separate queue from video so that the policing function applies separately to each.
5. Define enough bandwidth for each priority queue so that the built-in policer should not discard any messages from the priority queues.
6. Use Call Admission Control (CAC) tools to avoid adding too much voice or video to the network, which would trigger the policer function.

Shaping and Policing

This section introduces two related QoS tools—shaping and policing. These tools have a more specialized use and are not found in as many locations in a typical enterprise. These tools are most often used at the WAN edge in an enterprise network design.

Both policing and shaping monitor the bit rate of the combined messages that flow through a device. Once enabled, the policer or shaper notes each packet that passes and measures the number of bits per second over time. Both attempt to keep the bit rate at or below the configured speed, but by using two different actions: policers discard packets, and shapers hold packets in queues to delay the packets.

Shapers and policers monitor the traffic rate (the bits per second that move through the shaper or policer) versus a configured shaping rate or policing rate, respectively. The basic question that both ask is listed below, with the actions based on the answers:

Key Topic

1. Does this next packet push the measured rate past the configured shaping rate or policing rate?
2. If no:
   a. Let the packet keep moving through the normal path and do nothing extra to the packet.
3. If yes:
   a. If shaping, delay the message by queuing it.
   b. If policing, either discard the message or mark it differently.

This section first explains policing, which discards or re-marks messages that exceed the policing rate, followed by shaping, which slows down messages that exceed the shaping rate.

Policing

Focus on the traffic rate versus the configured policing rate for a moment, and the policing action of discarding messages. Those concepts sit at the core of what the policing function does.

Traffic arrives at networking devices at a varying rate, with valleys and spikes. That is, if you graph the bit rate of the collective bits that enter or exit any interface, the graph would look something like the one on the left side of Figure 11-18. The policer would measure that rate and make a similar measurement. Still on the left side of the figure, the horizontal dashed line represents the policing rate, which is the rate configured for the policer. So, the policer has some awareness of the measured bit rate over time, which can be compared to the configured rate.

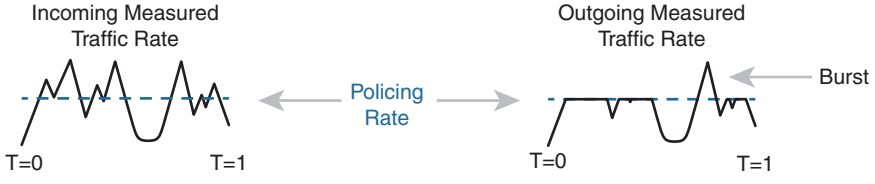


Figure 11-18 *Effect of a Policer and Shaper on an Offered Traffic Load*

The right side of the figure shows a graph of what happens to the traffic when a policer discards any messages that would have otherwise pushed the rate over the configured policing rate. In effect, the policer chops off the top of the graph at the policing rate.

The graph on the right also shows one example of a policer allowing a burst of traffic. Policers allow for a burst beyond the policing rate for a short time, after a period of low activity. So, that one peak that exceeds the policing rate on the graph on the right side allows for the nature of bursty data applications.

Where to Use Policing

Now that you understand the basics of policing, take a moment to ponder. Policers monitor messages, measure a rate, and discard some messages. How does that help a network in regard to QoS? At first glance, it seems to hurt the network, discarding messages, many of which the transport or application layer will have to resend. How does that improve bandwidth, delay, jitter, or loss?

Policing makes sense only in certain cases, and as a general tool, it can be best used at the edge between two networks. For instance, consider a typical point-to-point metro Ethernet WAN connection between two enterprise routers, R1 and R2. Usually, the enterprise network engineers just view the WAN as a cloud, with Ethernet interfaces on the routers, as shown at the top of Figure 11-19.

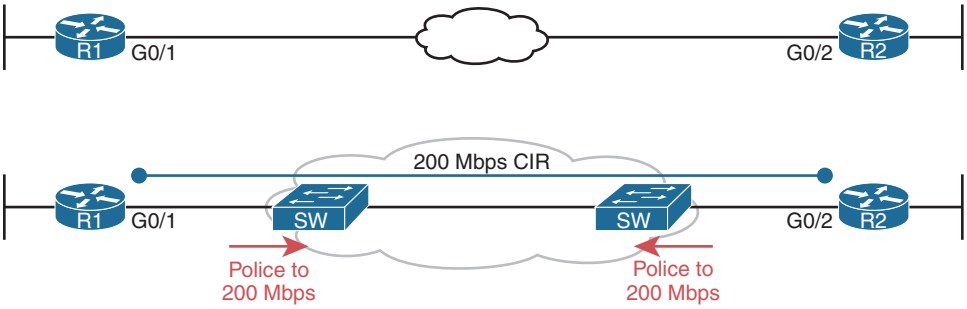


Figure 11-19 *Ethernet WAN: Link Speed Versus CIR*

Now think about the contract for this MetroE connection, as shown at the bottom of Figure 11-19. In this case, this connection uses Gigabit Ethernet for the access links, and a 200-Mbps *committed information rate* (CIR). That is, the SP providing the WAN service agrees to allow the enterprise to send 200 Mbps of traffic in each direction. However, remember that the enterprise routers transmit the data at the speed of the access link, or 1 Gbps in this case.

Think like the SP for a moment, and think about supporting tens of thousands of Gigabit Ethernet links into your WAN service, all with 200-Mbps CIRs. What would happen if you just let all those customers send data that, over time, averaged close to 1000 Mbps (1 Gbps)? That is, if all customers kept sending data far beyond their contracted CIR, that much traffic could cause congestion in the WAN service. Also, those customers might choose to pay for a lower CIR, knowing that the SP would send the data anyway. And customers who were well behaved and did not send more data than their CIR might suffer from the congestion just as much as the customers who send far too much data.

Figure 11-19 also notes the solution to the problem: The SP can police incoming packets, setting the policing rate to match the CIR that the customer chooses for that link. By doing so, the SP protects all customers from the negative effects of the customers who send too much traffic. Customers receive what they paid for. And the SP can provide reports of actual traffic rates, so the enterprise knows when to buy a faster CIR for each link.

Policers can discard excess traffic, but they can also re-mark packets. Think again about what an SP does with an ingress policer, as shown in Figure 11-19: they are discarding their customers' messages. So, the SP might want to make a compromise that works better for its customers, while still protecting the SP's network. The SP could mark the messages with a new marking value, with this strategy:

1. Re-mark packets that exceed the policing rate, but let them into the SP's network.
2. If other SP network devices are experiencing congestion when they process the packet, the different marking means that device can discard the packet. However…
3. …if no other SP network devices are experiencing congestion when forwarding that re-marked packet, it gets through the SP network anyway.

With this strategy, the SP can treat their customers a little better by discarding less traffic, while still protecting the SP's network during times of stress.

Summarizing the key features of policing:

Key Topic

- It measures the traffic rate over time for comparison to the configured policing rate.
- It allows for a burst of data after a period of inactivity.
- It is enabled on an interface, in either direction, but typically at ingress.
- It can discard excess messages but can also re-mark the message so that it is a candidate for more aggressive discard later in its journey.

Shaping

You have a 1-Gbps link from a router into a SP, but a 200-Mbps CIR for traffic to another site, as seen in Figure 11-19. The SP has told you that it always discards incoming traffic that exceeds the CIR. The solution? Use a shaper to slow down the traffic—in this case to a 200-Mbps shaping rate.

That scenario—shaping before sending data to an SP that is policing—is one of the typical uses of a shaper. Shapers can be useful in other cases as well, but generally speaking, shapers make sense when a device can send at a certain speed, but there is a benefit to slowing down the rate.

The shaper slows messages down by queuing the messages. The shaper then services the shaping queues, but not based on when the physical interface is available. Instead, the shaper schedules messages from the shaping queues based on the shaping rate, as shown in Figure 11-20. Following the left-to-right flow in the figure, for a router, the packet is routed out an interface; the shaper queues packets so that the sending rate through the shaper does not exceed the shaping rate; and then output queuing works as normal, if needed.

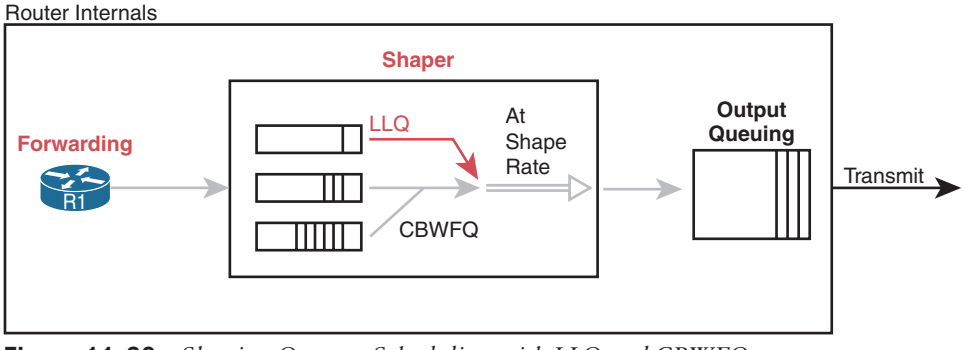


Figure 11-20 *Shaping Queues: Scheduling with LLQ and CBWFQ*

Note that in some cases, the output queuing function has little to do. For instance, in the earlier example shown in Figure 11-19, the SP is policing incoming messages at 200 Mbps. If the router (R1, for instance) were to shape all traffic exiting toward the SP to 200 Mbps as well, with that 1-Gbps interface, the output queue would seldom if ever be congested.

Because shapers create queues where messages wait, you should apply a queuing tool to those queues. It is perfectly normal to apply the round-robin and priority queuing features of CBWFQ and LLQ, respectively, to the shaping queues, as noted in the figure.

Setting a Good Shaping Time Interval for Voice and Video

Once again, a QoS tool has attempted to solve one QoS problem but introduces another. The unfortunate side effect of a shaper is that it slows down packets, which then creates more delay and probably more jitter. The delay occurs in part because of the message simply waiting in a queue, but partly because of the mechanisms used by a shaper. Thankfully, you can (and should) configure a shaper's setting that changes the internal operation of the shaper, which then reduces the delay and jitter caused to voice and video traffic.

A shaper's *time interval* refers to its internal logic and how a shaper averages, over time, sending at a particular rate. A shaper basically sends as fast as it can and then waits; sends and waits; sends and waits. For instance, the policing and shaping example in this section suggests shaping at 200 Mbps on a router that has a 1000-Mbps (1-Gbps) outgoing interface. In that case, the shaper would result in the interface sending data 20 percent of the time and being silent 80 percent of the time.

Figure 11-21 shows a graph of the shaping time interval concept, assuming a time interval of 1 second. To average 200 million bits per second, the shaper would allow 200 million bits to exit its shaping queues and exit the interface each second. Because the interface transmits bits at 1 Gbps, it takes just .2 seconds, or 200 ms, to send all 200 million bits. Then the shaper must wait for the rest of the time interval, another 800 ms, before beginning the next time interval.

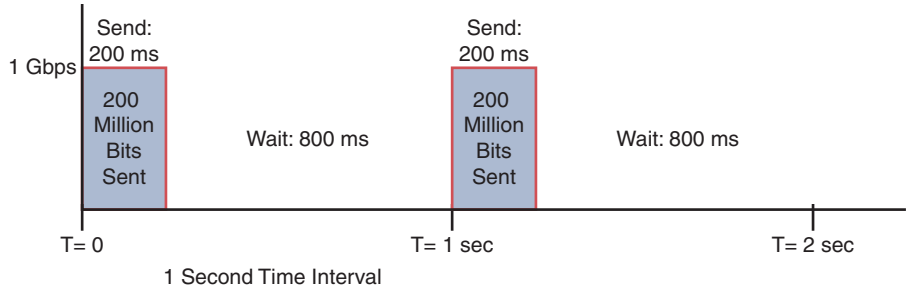


Figure 11-21 *One Second (1000 ms) Shaping Time Interval, Shaping at 20 Percent of Line Rate*

Now think about a voice or video packet that needs very low delay and jitter—and unfortunately, it arrives just as the shaper finishes sending data for a time interval. Even if that voice or video packet is in a priority shaping queue, the packet will wait 800 ms before the shaper schedules the next packet—far too long compared to the 150-ms one-way delay goal for voice.

The solution to this problem: configure a short time interval. For example, consider the following time intervals (abbreviated Tc), and their effects, for this same example (1-Gbps link, shaping to 200 Mbps), but with shorter and shorter time intervals:

Tc = 1 second (1000 ms): Send at 1 Gbps for 200 ms, rest for 800 ms

Tc = .1 second (100 ms): Send at 1 Gbps for 20 ms, rest for 80 ms

Tc = .01 second (10 ms): Send at 1 Gbps for 2 ms, rest for 8 ms

When shaping, use a short time interval. By recommendation, use a 10-ms time interval to support voice and video. With that setting, a voice or video packet should wait no more than 10 ms while waiting for the next shaping time interval, at which point the priority queue scheduling should take all the voice and video messages next.

Summarizing the key features of shapers:

Key Topic

- Shapers measure the traffic rate over time for comparison to the configured shaping rate.
- Shapers allow for bursting after a period of inactivity.
- Shapers are enabled on an interface for egress (outgoing packets).
- Shapers slow down packets by queuing them and over time releasing them from the queue at the shaping rate.
- Shapers use queuing tools to create and schedule the shaping queues, which is very important for the same reasons discussed for output queuing.

Congestion Avoidance

The QoS feature called congestion avoidance attempts to reduce overall packet loss by preemptively discarding some packets used in TCP connections. To see how it works, you first need to look at how TCP works in regard to windowing and then look at how congestion avoidance features work.

TCP Windowing Basics

TCP uses a flow control mechanism called *windowing*. Each TCP receiver grants a window to the sender. The window, which is a number, defines the number of bytes the sender can send over the TCP connection before receiving a TCP acknowledgment for at least some of those bytes. More exactly, the window size is the number of unacknowledged bytes that the sender can send before the sender must simply stop and wait.

The TCP window mechanism gives the receiver control of the sender's rate of sending data. Each new segment sent by the receiver back to the sender grants a new window, which can be smaller or larger than the previous window. By raising and lowering the window, the receiver can make the sender wait more or wait less.

NOTE Each TCP connection has two senders and two receivers; that is, each host sends and receives data. For this discussion, focus on one direction, with one host as the sender and the other as the receiver. If calling one host the "sender" and one the "receiver," note that the receiver then acknowledges data in TCP segments sent back to the sender by the receiver.

By choice, when all is well, the receiver keeps increasing the granted window, doubling it every time the receiver acknowledges data. Eventually, the window grows to the point that the sender never has to stop sending: the sender keeps receiving TCP acknowledgments before sending all the data in the previous window. Each new acknowledgment (as listed in a TCP segment and TCP header) grants a new window to the sender.

Also by choice, when a TCP receiver senses the loss of a TCP segment, he shrinks the window with the next window size listed in the next TCP segment the receiver sends back to the sender. For each TCP segment lost, the window can shrink by one-half, with multiple segment losses causing the window to shrink by half multiple times, slowing down the sender's rate significantly.

Now think about router queues for a moment. Without a congestion avoidance tool, an event called a *tail drop* causes the most drops in a network. Figure 11-22 shows the idea,

showing the same queuing system, but in three separate conditions—little congestion, medium congestion, and much congestion. On the left, with little congestion, the output queues on an interface have not yet filled. In the middle, the queues have started to fill, with one queue being totally full. Any new packets that arrive for that queue right now will be dropped because there is no room at the tail of the queue (tail drop).

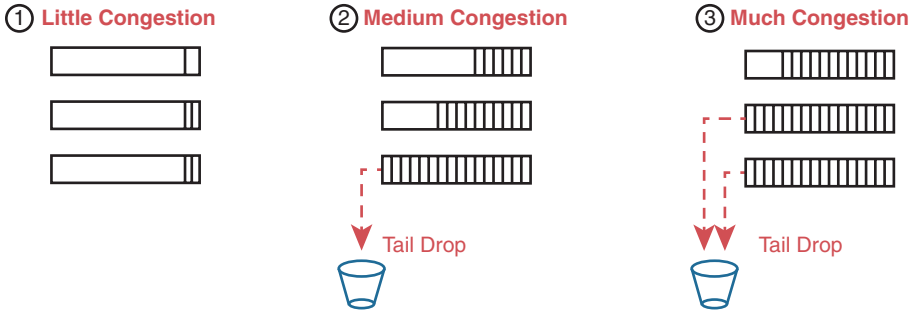


Figure 11-22 *Tail Drop Concepts with Three Different Scenarios*

The worse the congestion in the queues, the more likely tail drop will occur, as shown with the most congested case on the right side of the figure. The more congestion, the bigger the negative impact on traffic—both in terms of loss and in terms of increasing delay in TCP connections.

Congestion Avoidance Tools

Congestion avoidance tools attempt to avoid the congestion, primarily through using TCP's own windowing mechanisms. These tools discard some TCP segments before the queues fill, hoping that enough TCP connections will slow down, reducing congestion, and avoiding a much worse problem: the effects of many more packets being dropped due to tail drop. The strategy is simple: discard some now in hopes that the device discards far fewer in the long term.

Congestion avoidance tools monitor the average queue depth over time, triggering more severe actions the deeper the queue, as shown in Figure 11-23. The height of the box represents the queue depth, or the number of packets in the queue. When the queue depth is low, below the minimum threshold values, the congestion avoidance tool does nothing. When the queue depth is between the minimum and maximum thresholds, the congestion avoidance tool discards a percentage of the packets—usually a small percentage, like 5, 10, or 20 percent. If the queue depth passes the maximum threshold, the tool drops all packets, in an action called *full drop*.

Of course, like all the QoS tools mentioned in this chapter, congestion avoidance tools can classify messages to treat some packets better than others. In the same queue, packets with one marking might be dropped more aggressively, and those with better DSCP markings dropped less aggressively.

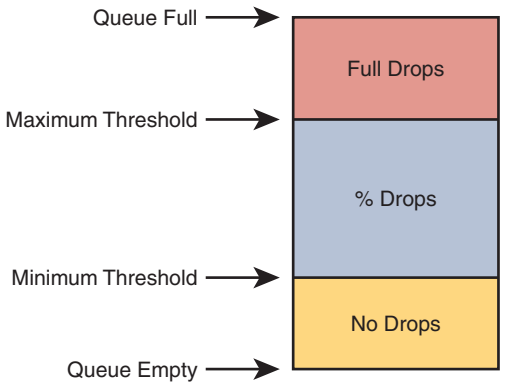


Figure 11-23 *Mechanisms of Congestion Avoidance*

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 11-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 11-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website
Watch video		website

Review All the Key Topics

Key Topic

Table 11-5 Key Topics for Chapter 11

Key Topic Element	Description	Page Number
List	Four QoS characteristics	228
List	Voice call QoS requirements	232
List	Video QoS requirements	232
Figure 11-6	IP Precedence and IP DSCP marking fields	237
Figure 11-7	802.1Q CoS marking field	238
Figure 11-10	Trust boundary with IP Phones	239
Figure 11-14	Queuing components	242
Figure 11-17	LLQ scheduling logic with a priority queue	244

Key Topic Element	Description	Page Number
List	A strategy for using queuing (congestion management) to prioritize traffic	245
List	Logic steps for shapers and policers	245
List	Key features of policers	248
List	Key features of shapers	250

Key Terms You Should Know

per-hop behavior (PHB), marking, classification, Quality of Service (QoS), IP Precedence (IPP), Differentiated Services Code Point (DSCP), Class of Service (CoS), bandwidth, delay, jitter, loss, queuing, priority queue, round robin, policing, shaping, Differentiated Services (DiffServ), policing rate, shaping rate

CHAPTER 12

Miscellaneous IP Services

This chapter covers the following exam topics:

3.0 IP Connectivity

3.5 Describe the purpose of First Hop Redundancy Protocol

4.0 Infrastructure Services

4.4 Explain the function of SNMP in network operations

4.9 Describe the capabilities and function of TFTP/FTP in the network

When reading this chapter, think of it as three separate small topics rather than one large topic. The content just happens to include a few IP-based services that have little to do with each other, but the length of coverage of each topic is too short to justify a separate chapter. The result: Chapter 12, "Miscellaneous IP Services." So when reading, feel free to treat each of the three major headings as a separate study event.

First Hop Redundancy Protocols (FHRPs), which provides redundancy for the function of the default router in any subnet, begins the chapter. The term *FHRP* refers to a class of solutions, with three options, and with the examples showing the most popular option, Hot Standby Router Protocol (HSRP).

Simple Network Management Protocol (SNMP) follows in the second major section. As per the associated exam topic, this section focuses on SNMP concepts rather than configuration, including how managed devices—SNMP agents—can be interrogated by network management systems—SNMP clients—to find the current status of each device.

File Transfer Protocol (FTP) and Trivial File Transfer Protocol (TFTP) star in the third major section. The first branch of this section focuses on a few practical uses of TFTP and FTP, specifically how to use these protocols on Cisco routers to upgrade the IOS. Armed with that practical knowledge, you then look at the protocol details of both FTP and TFTP in the rest of the section.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 12-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
First Hop Redundancy Protocol	1–3
Simple Network Management Protocol	4, 5
FTP and TFTP	6, 7

1. R1 and R2 attach to the same Ethernet VLAN, with subnet 10.1.19.0/25, with addresses 10.1.19.1 and 10.1.19.2, respectively, configured with the **ip address** interface subcommand. Host A refers to 10.1.19.1 as its default router, and host B refers to 10.1.19.2 as its default router. The routers do not use an FHRP. Which of the following is a problem for this LAN?
   a. The design breaks IPv4 addressing rules because two routers cannot connect to the same LAN subnet.
   b. If one router fails, neither host can send packets off-subnet.
   c. If one router fails, both hosts will use the one remaining router as a default router.
   d. If one router fails, the host that uses that router as a default router cannot send packets off-subnet.
2. R1 and R2 attach to the same Ethernet VLAN, with subnet 10.1.19.0/25, with addresses 10.1.19.1 and 10.1.19.2, respectively, configured with the **ip address** interface subcommand. The routers use an FHRP. Host A and host B attach to the same LAN and have correct default router settings per the FHRP configuration. Which of the following statements is true for this LAN?
   a. The design breaks IPv4 addressing rules because two routers cannot connect to the same LAN subnet.
   b. If one router fails, neither host can send packets off-subnet.
   c. If one router fails, both hosts will use the one remaining router as a default router.
   d. If one router fails, only one of the two hosts will still be able to send packets off-subnet.
3. R1 and R2 attach to the same Ethernet VLAN, with subnet 10.1.19.0/25, with addresses 10.1.19.1 and 10.1.19.2, respectively, configured with the **ip address** interface subcommand. The routers use HSRP. The network engineer prefers to have R1 be the default router when both R1 and R2 are up. Which of the following is the likely default router setting for hosts in this subnet?
   a. 10.1.19.1
   b. 10.1.19.2
   c. Another IP address in subnet 10.1.19.0/25 other than 10.1.19.1 and 10.1.19.2
   d. A host name that the FHRP mini-DNS will initially point to 10.1.19.1

4. A Network Management Station (NMS) is using SNMP to manage some Cisco routers and switches with SNMPv2c. Which of the following answers most accurately describes how the SNMP agent on a router authenticates any SNMP Get requests received from the NMS?
   a. Using a username and hashed version of a password
   b. Using either the read-write or read-only community string
   c. Using only the read-write community string
   d. Using only the read-only community string
5. Which of the following SNMP messages are typically sent by an SNMP agent?
   a. Trap
   b. Get Request
   c. Inform
   d. Set Request
6. An FTP client connects to an FTP server using active mode and retrieves a copy of a file from the server. Which of the answers describes a TCP connection initiated by the FTP client?
   a. The FTP control connection
   b. The FTP data connection
   c. The FTP TLS connection
   d. None of the other answers are correct.
7. Which of the following functions are supported by FTP but not by TFTP? (Choose two answers.)
   a. Transferring files from client to server
   b. Changing the current directory on the server
   c. Transferring files from server to client
   d. Listing directory contents of a server's directory

Foundation Topics

First Hop Redundancy Protocol

When networks use a design that includes redundant routers, switches, LAN links, and WAN links, in some cases other protocols are required to take advantage of that redundancy and to prevent problems caused by it.

For instance, imagine a WAN with many remote branch offices. If each remote branch has two WAN links connecting it to the rest of the network, those routers can use an IP routing protocol to pick the best routes. The routing protocol learns routes over both WAN links, adding the best route into the routing table. When the better WAN link fails, the routing protocol adds the alternate route to the IP routing table, taking advantage of the redundant link.

As another example, consider a LAN with redundant links and switches. Those LANs have problems unless the switches use Spanning Tree Protocol (STP) or Rapid STP (RSTP). STP/RSTP prevents the problems created by frames that loop through those extra redundant paths in the LAN.

This section examines yet another type of protocol that helps when a network uses some redundancy, this time with redundant default routers. When two or more routers connect to the same LAN subnet, all those routers could be used as the default router for the hosts in the subnet. However, to make the best use of the redundant default routers, another protocol is needed. The term *First Hop Redundancy Protocol* (FHRP) refers to the category of protocols that can be used so that the hosts take advantage of redundant routers in a subnet.

This first major section of the chapter discusses the major concepts behind how different FHRPs work. This section begins by discussing a network's need for redundancy in general and the need for redundant default routers. It then shows how the three available FHRP options can each solve the problems that occur when using redundant default routers.

The Need for Redundancy in Networks

Networks need redundant links to improve the availability of those networks. Eventually, something in a network will fail. A router power supply might fail, or a cable might break, or a switch might lose power. And those WAN links, shown as simple lines in most drawings in this book, are actually the most complicated physical parts of the network, with many individual parts that can fail as well.

Depending on the design of the network, the failure of a single component might mean an outage that affects at least some part of the user population. Network engineers refer to any one component that, if it fails, brings down that part of the network as a *single point of failure*. For instance, in Figure 12-1, the LANs appear to have some redundancy, whereas the WAN does not. If most of the traffic flows between sites, many single points of failure exist, as shown in the figure.

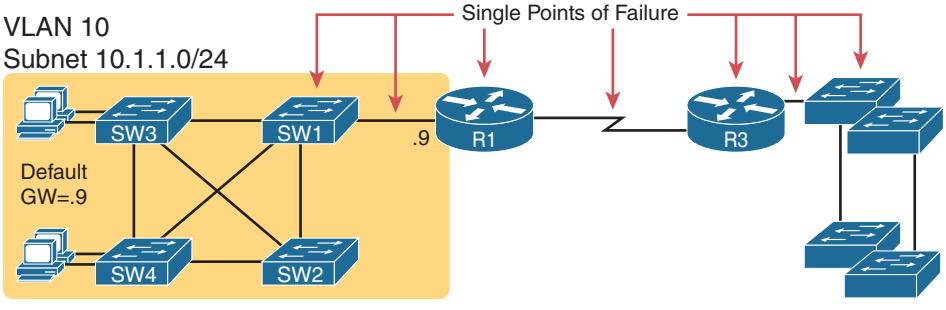


Figure 12-1 *R1 and the One WAN Link as Single Points of Failure*

The figure notes several components as a single point of failure. If any one of the noted parts of the network fails, packets cannot flow from the left side of the network to the right.

Generally speaking, to improve availability, the network engineer first looks at a design and finds the single points of failure. Then the engineer chooses where to add to the network so

that one (or more) single point of failure now has redundant options, increasing availability. In particular, the engineer

- Adds redundant devices and links
- Implements any necessary functions that take advantage of the redundant device or link

For instance, of all the single points of failure in Figure 12-1, the most expensive over the long term would likely be the WAN link because of the ongoing monthly charge. However, statistically, the WAN links are the most likely component to fail. So, a reasonable upgrade from the network in Figure 12-1 would be to add a WAN link and possibly even connect to another router on the right side of the network, as shown in Figure 12-2.

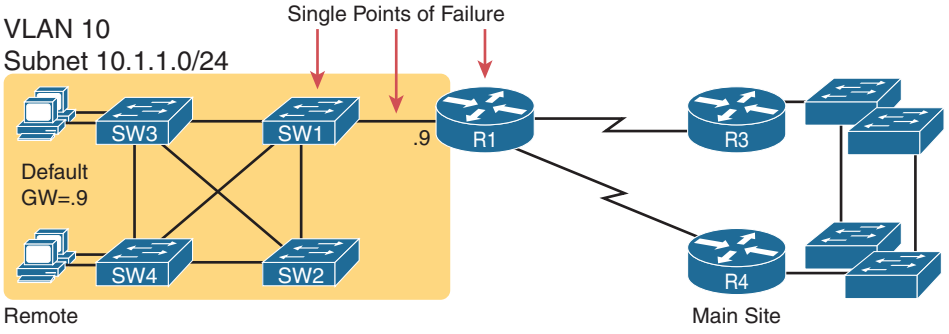


Figure 12-2 *Higher Availability but with R1 Still as a Single Point of Failure*

Many real enterprise networks follow designs like Figure 12-2, with one router at each remote site, two WAN links connecting back to the main site, and redundant routers at the main site (on the right side of the figure). Compared to Figure 12-1, the design in Figure 12-2 has fewer single points of failure. Of the remaining single points of failure, a risk remains, but it is a calculated risk. For many outages, a reload of the router solves the problem, and the outage is short. But the risk still exists that the switch or router hardware fails completely and requires time to deliver a replacement device on-site before that site can work again.

For enterprises that can justify more expense, the next step in higher availability for that remote site is to protect against those catastrophic router and switch failures. In this particular design, adding one router on the left side of the network in Figure 12-2 removes all the single points of failure that had been noted earlier. Figure 12-3 shows the design with a second router, which connects to a different LAN switch so that SW1 is also no longer a single point of failure.

NOTE Medium to large enterprise networks work hard at striking a balance of high-availability features versus the available budget dollars. Cisco.com has many design documents that discuss trade-offs in high-availability design. If interested in learning more, search Cisco.com for "high availability campus network design."

Answers to the "Do I Know This Already?" quiz:

1 D **2** C **3** C **4** B **5** A, C **6** A **7** B, D

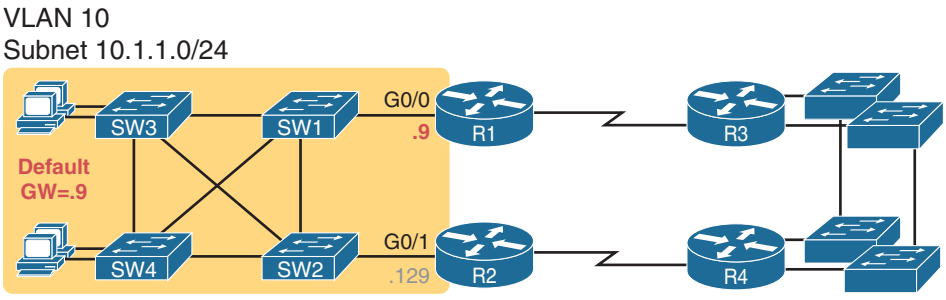


Figure 12-3 *Removing All Single Points of Failure from the Network Design*

The Need for a First Hop Redundancy Protocol

Of the designs shown so far in this chapter, only Figure 12-3's design has two routers to support the LAN on the left side of the figure, specifically the same VLAN and subnet. While having the redundant routers on the same subnet helps, the network needs to use an FHRP when these redundant routers exist.

To see the need and benefit of using an FHRP, first think about how these redundant routers could be used as default routers by the hosts in VLAN 10/subnet 10.1.1.0/24, as shown in Figure 12-4. The host logic will remain unchanged, so each host has a single default router setting. So, some design options for default router settings include the following:

- All hosts in the subnet use R1 (10.1.1.9) as their default router, and they statically reconfigure their default router setting to R2's 10.1.1.129 if R1 fails.
- All hosts in the subnet use R2 (10.1.1.129) as their default router, and they statically reconfigure their default router setting to R1's 10.1.1.9 if R2 fails.
- Half the hosts use R1, and half use R2, as their default router, and if either router fails, that half of the users statically reconfigure their default router setting.

To make sure the concept is clear, Figure 12-4 shows this third option, with half the hosts using R1 and the other half using R2. The figure removes all the LAN switches just to unclutter the figure. Hosts A and B use R1 as their default router, and hosts C and D use R2 as their default router.

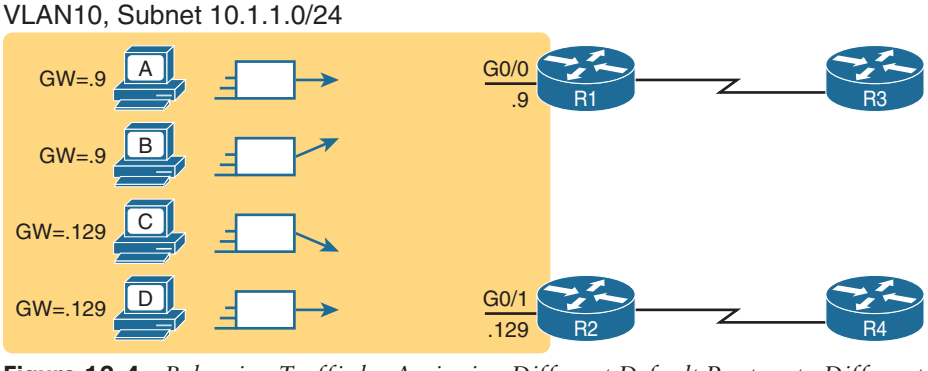


Figure 12-4 *Balancing Traffic by Assigning Different Default Routers to Different Clients*

All of these options have a problem: the users have to take action. They have to know an outage occurred. They have to know how to reconfigure their default router setting. And they have to know when to change it back to the original setting.

FHRPs make this design work better. The two routers appear to be a single default router. The users never have to do anything: their default router setting remains the same, and their ARP table even remains the same.

To allow the hosts to remain unchanged, the routers have to do some more work, as defined by one of the FHRP protocols. Generically, each FHRP makes the following happen:

Key Topic

1. All hosts act like they always have, with one default router setting that never has to change.
2. The default routers share a virtual IP address in the subnet, defined by the FHRP.
3. Hosts use the FHRP virtual IP address as their default router address.
4. The routers exchange FHRP protocol messages so that both agree as to which router does what work at any point in time.
5. When a router fails or has some other problem, the routers use the FHRP to choose which router takes over responsibilities from the failed router.

The Three Solutions for First-Hop Redundancy

The term *First Hop Redundancy Protocol* does not name any one protocol. Instead, it names a family of protocols that fill the same role. For a given network, like the left side of Figure 12-4, the engineer would pick one of the protocols from the FHRP family.

NOTE *First Hop* is a reference to the default router being the first router, or first router hop, through which a packet must pass.

Table 12-2 lists the three FHRP protocols in chronological order, based on when these were first used. Cisco first introduced the proprietary Hot Standby Router Protocol (HSRP), and it worked well for many of its customers. Later, the IETF developed an RFC for a similar protocol, Virtual Router Redundancy Protocol (VRRP). Finally, Cisco developed a more robust option, Gateway Load Balancing Protocol (GLBP).

Key Topic

Table 12-2 Three FHRP Options

Acronym	Full Name	Origin	Redundancy Approach	Load Balancing Per...
HSRP	Hot Standby Router Protocol	Cisco	active/standby	subnet
VRRP	Virtual Router Redundancy Protocol	RFC 5798	active/standby	subnet
GLBP	Gateway Load Balancing Protocol	Cisco	active/active	host

This chapter focuses on HSRP and does not discuss VRRP and GLBP other than this brief mention. HSRP, the first of the three FHRP protocols to enter the market, remains a popular option in many networks. The current CCNA 200-301 exam requires you to know the functions of an FHRP, so the example of HSRP meets that need, with the next few pages walking through the concepts of how HSRP works. (Note that Appendix D, "Topics from

Previous Editions," contains a section with more depth about GLBP, copied from an earlier edition of the book, as well as a section on HSRP configuration if you are interested in reading more that goes beyond the current exam's topics.)

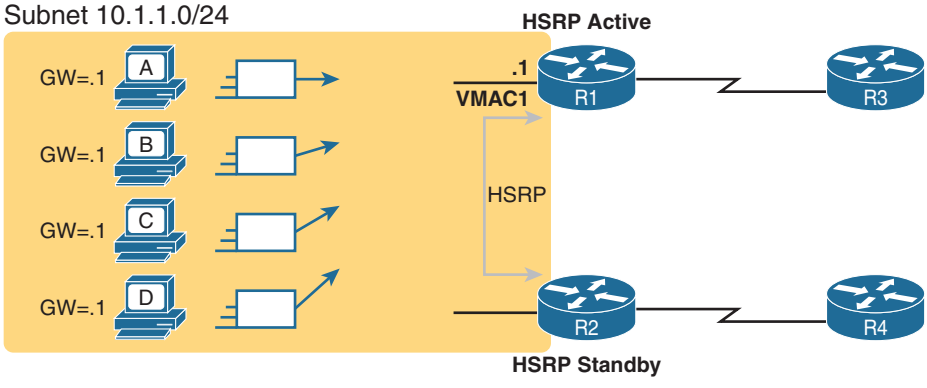
HSRP Concepts

HSRP operates with an active/standby model (also more generally called *active/passive*). HSRP allows two (or more) routers to cooperate, all being willing to act as the default router. However, at any one time, only one router actively supports the end-user traffic. The packets sent by hosts to their default router flow to that one active router. Then the other routers, with an HSRP standby state, sit there patiently waiting to take over should the active HSRP router have a problem.

The HSRP active router implements a virtual IP address and matching virtual MAC address. This virtual IP address exists as part of the HSRP configuration, which is an additional configuration item compared to the usual **ip address** interface subcommand. This virtual IP address is in the same subnet as the interface IP address, but it is a different IP address. The router then automatically creates the virtual MAC address. All the cooperating HSRP routers know these virtual addresses, but only the HSRP active router uses these addresses at any one point in time.

Hosts refer to the virtual IP address as their default router address, instead of any one router's interface IP address. For instance, in Figure 12-5, R1 and R2 use HSRP. The HSRP virtual IP address is 10.1.1.1, with the virtual MAC address referenced as VMAC1 for simplicity's sake.

Key Topic



Host ARP Table

IP	MAC
10.1.1.1	**VMAC1**

Figure 12-5 *All Traffic Goes to .1 (R1, Which Is Active); R2 Is Standby*

HSRP Failover

HSRP on each router has some work to do to make the network function as shown in Figure 12-5. The two routers need HSRP configuration, including the virtual IP address. The two routers send HSRP messages to each other to negotiate and decide which router should currently be active and which should be standby. Then the two routers continue to send messages to each other so that the standby router knows when the active router fails so that it can take over as the new active router.

Figure 12-6 shows the result when R1, the HSRP active router in Figure 12-5, fails. R1 quits using the virtual IP and MAC address, while R2, the new active router, starts using these addresses. The hosts do not need to change their default router settings at all, with traffic now flowing to R2 instead of R1.

Key Topic

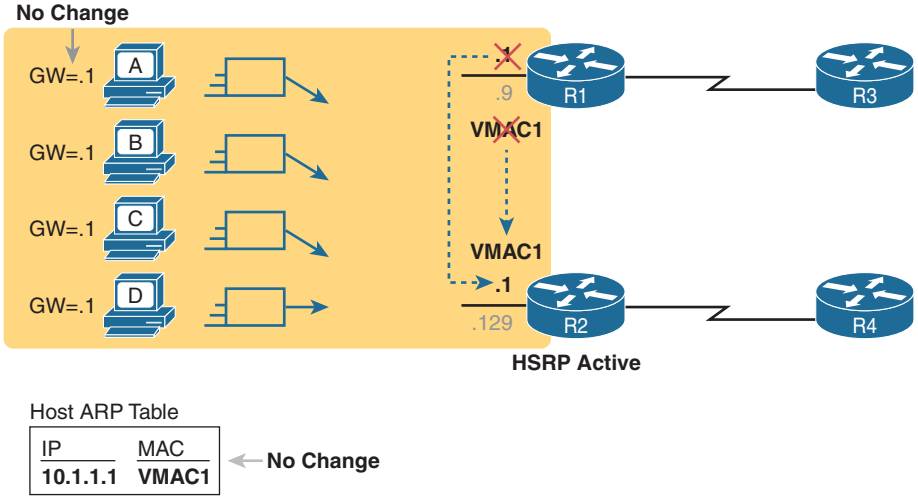


Figure 12-6 *Packets Sent Through R2 (New Active) Once It Takes Over for Failed R1*

When the failover happens, some changes do happen, but none of those changes happen on the hosts. The host keeps the same default router setting, set to the virtual IP address (10.1.1.1 in this case). The host's ARP table does not have to change either, with the HSRP virtual MAC being listed as the MAC address of the virtual router.

When the failover occurs, changes happen on both the routers and the LAN switches. Clearly, the new active router has to be ready to receive packets (encapsulated inside frames) using the virtual IP and MAC addresses. However, the LAN switches, hidden in the last few figures, formerly sent frames destined for VMAC1 to router R1. Now the switches must know to send the frames to the new active router, R2.

To make the switches change their MAC address table entries for VMAC1, R2 sends an Ethernet frame with VMAC1 as the source MAC address. The switches, as normal, learn the source MAC address (VMAC1), but with new ports that point toward R2. The frame is also a LAN broadcast, so all the switches learn a MAC table entry for VMAC1 that leads toward R2. (By the way, this Ethernet frame holds an ARP Reply message, called a gratuitous ARP, because the router sends it without first receiving an ARP Request.)

HSRP Load Balancing

The active/standby model of HSRP means that in one subnet all hosts send their off-subnet packets through only one router. In other words, the routers do not share the workload, with one router handling all the packets. For instance, back in Figure 12-5, R1 was the active router, so all hosts in the subnet sent their packets through R1, and none of the hosts in the subnet sent their packets through R2.

HSRP does support load balancing by preferring different routers to be the active router in different subnets. Most sites that require a second router for redundancy are also big

enough to use several VLANs and subnets at the site. The two routers will likely connect to all the VLANs, acting as the default router in each VLAN. HSRP then can be configured to prefer one router as active in one VLAN and another router as active in another VLAN, balancing the traffic. Or you can configure multiple instances of HSRP in the same subnet (called multiple HSRP groups), preferring one router to be active in one group and the other router to be preferred as active in another.

For instance, Figure 12-7 shows a redesigned LAN, now with two hosts in VLAN 1 and two hosts in VLAN 2. Both R1 and R2 connect to the LAN, and both use a VLAN trunking and router-on-a-stick (ROAS) configuration. Both routers use HSRP in each of the two subnets, supporting each other. However, on purpose, R1 has been configured so that it wins the negotiation to become HSRP active in VLAN 1, and R2 has been configured to win in VLAN 2.

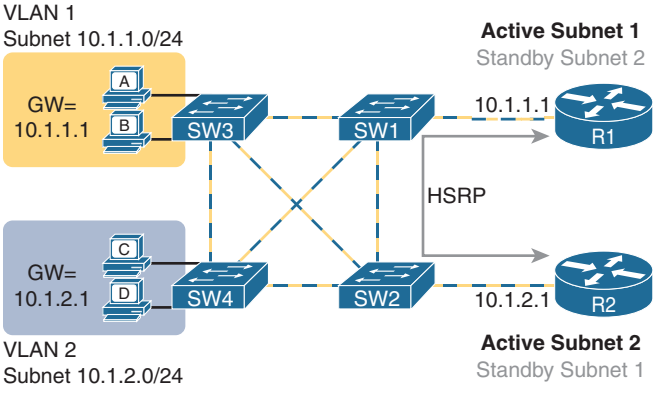


Figure 12-7 *Load Balancing with HSRP by Using Different Active Routers per Subnet*

Note that by having each router act as the HSRP active router in some subnets, the design makes use of both routers and both WAN links.

FHRPs are needed on any device that acts as a default router, which of course includes both traditional routers and Layer 3 switches. HSRP can be configured on routers and Layer 3 switches on interfaces that have IP addresses configured. However, in most cases, HSRP is used on interfaces to subnets that have hosts that need to use a default router. Those interfaces include router physical interfaces, router trunk subinterfaces, and Layer 3 switched virtual interfaces (SVI).

Simple Network Management Protocol

In 1988, RFC 1065, "Structure and Identification of Management Information for TCP/IP-based Internets," was published. The idea behind this document was the fact that information about devices on a TCP/IP-based network—configuration settings, status information, counters, and so on—could be broken down into a database of variables. Those variables could then be collected by management software to monitor and manage the IP-based network. After all, the elements of any IP-based machines would have commonalities. For example, a PC, a network printer, and a router would all have commonalities such as interfaces, IP addresses, and buffers. Why not create a standardized database of these variables and a simple system for monitoring and managing them? This idea was brilliant, caught on, and became what we know today as *Simple Network Management Protocol* (SNMP).

This second of three major sections of the chapter now turns our attention to SNMP by looking at the major concepts along with the two common versions used today: SNMPv2c and SNMPv3.

SNMP is an application layer protocol that provides a message format for communication between what are termed *managers* and *agents*. An SNMP manager is a network management application running on a PC or server, with that host typically being called a Network Management Station (NMS). Many SNMP agents exist in the network, one per device that is managed. The SNMP agent is software running inside each device (router, switch, and so on), with knowledge of all the variables on that device that describe the device's configuration, status, and counters. The SNMP manager uses SNMP protocols to communicate with each SNMP agent.

Each agent keeps a database of variables that make up the parameters, status, and counters for the operations of the device. This database, called the Management Information Base (MIB), has some core elements in common across most networking devices. It also has a large number of variables unique to that type of device—for instance, router MIBs will include variables not needed on switch MIBs, and vice versa. (For perspective, I did a quick check on a router when writing this section and found a little over 7000 MIB variables on a router.)

Figure 12-8 connects a few of these ideas and terms together. First, many companies sell SNMP management products—for example, the Cisco Prime series of management products (www.cisco.com/go/prime) use SNMP (and other protocols) to manage networks. IOS on routers and switches include an SNMP agent, with built-in MIB, that can be enabled with the configuration shown later in this chapter.

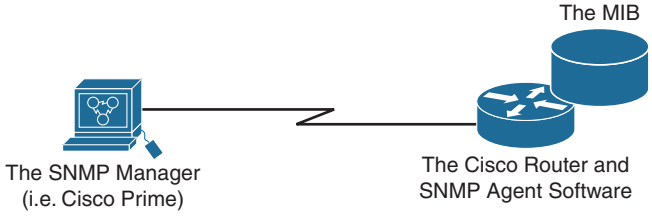


Figure 12-8 *Elements of Simple Network Management Protocol*

SNMP Variable Reading and Writing: SNMP Get and Set

The NMS typically polls the SNMP agent on each device. The NMS can notify the human user in front of the PC or send emails, texts, and so on to notify the network operations staff of any issues identified by the data found by polling the devices. You can even reconfigure the device through these SNMP variables in the MIB if you permit this level of control.

Specifically, the NMS uses the SNMP Get, GetNext, and GetBulk messages (together referenced simply as Get messages) to ask for information from an agent. The NMS sends an SNMP Set message to write variables on the SNMP agent as a means to change the configuration of the device. These messages come in pairs, with, for instance, a Get Request asking the agent for the contents of a variable, and the Get Response supplying that information. Figure 12-9 shows an example of a typical flow, with the NMS using an SNMP Get to ask for the MIB variable that describes the status of a particular router interface.

Key Topic

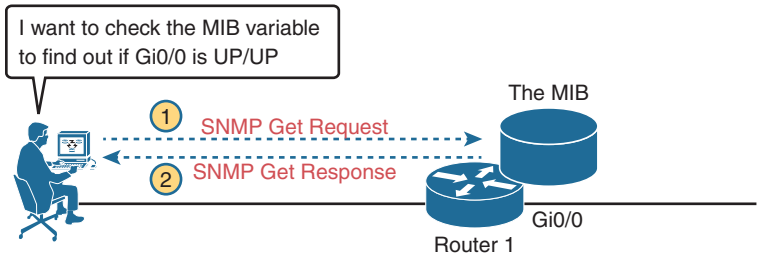


Figure 12-9 *SNMP Get Request and Get Response Message Flow*

SNMP permits much flexibility in how you monitor variables in the MIB. Most commonly, a network administrator gathers and stores statistics over time using the NMS. The NMS, with the stored data, can then analyze various statistical facts such as averages, minimums, and maximums. To be proactive, administrators can set thresholds for certain key variables, telling the NMS to send a notification (email, text, and so on) when a threshold is passed.

SNMP Notifications: Traps and Informs

In addition to asking for information with Get commands and setting variables on agents with the Set command, SNMP agents can initiate communications to the NMS. These messages, generally called notifications, use two specific SNMP messages: Trap and Inform. SNMP agents send a Trap or Inform SNMP message to the NMS to list the state of certain MIB variables when those variables reach a certain state.

As an example of a Trap, suppose that Router 1's G0/0 interface fails, as shown at step 1 of Figure 12-10. With Traps configured, the router would send an SNMP Trap message to the NMS, with that Trap message noting the down state of the G0/0 interface. Then, the NMS software can send a text message to the network support staff, pop up a window on the NMS screen, change the color of the correct router icon to red on the graphical interface, and so on.

Key Topic

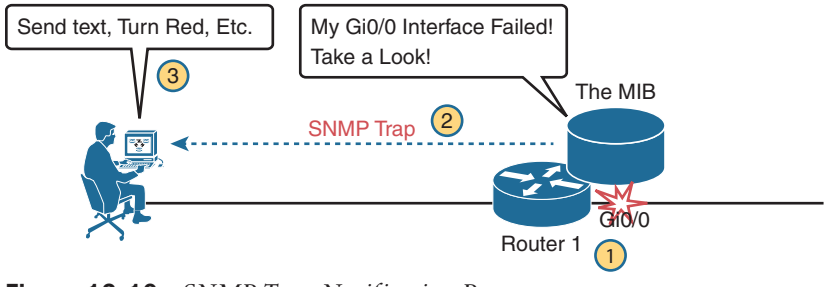


Figure 12-10 *SNMP Trap Notification Process*

SNMP Traps and Inform messages have the exact same purpose but differ in the protocol mechanisms. SNMP Traps, available since the first version of SNMP from the late 1980s (SNMP Version 1, or SNMPv1), use a fire-and-forget process. The SNMP agent sends the Trap to the IP address of the NMS, with UDP as the transport protocol as with all SNMP messages, and with no application layer error recovery. If the Trap arrives, great; if it is lost in transit, it is lost.

Inform messages are like Trap messages but with reliability added. Added to the protocol with SNMP Version 2 (SNMPv2), Informs still use UDP but add application layer reliability.

The NMS must acknowledge receipt of the Inform with an SNMP Response message, or the SNMP agent will time out and resend the Inform.

Note that Traps and Informs both have a useful role today, and Traps are still frequently used. Both inform the NMS. Traps use less overhead on the agent, while Informs improve reliability of the messages but require a little more overhead effort.

The Management Information Base

Every SNMP agent has its own Management Information Base. The MIB defines variables whose values are set and updated by the agent. The MIB variables on the devices in the network enable the management software to monitor/control the network device.

More formally, the MIB defines each variable as an *object ID* (OID). On most devices, the MIB then organizes the OIDs based in part on RFC standards, and in part with vendor-proprietary variables. The MIB organizes all the variables into a hierarchy of OIDs, usually shown as a tree. Each node in the tree can be described based on the tree structure sequence, either by name or by number. Figure 12-11 shows a small part of the tree structure of an MIB that happens to be part of the Cisco-proprietary part of the MIB.

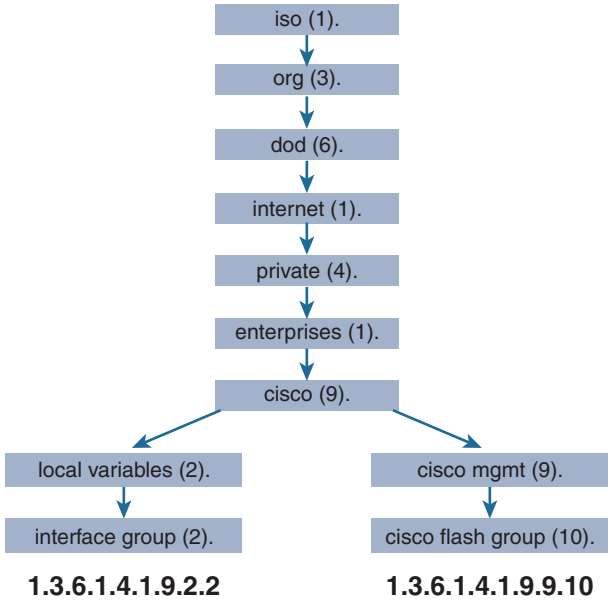


Figure 12-11 *Management Information Base (MIB)*

Working directly with an MIB, with long variable names and numbers, can be a bit of a challenge, so NMS software typically hides the complexity of the MIB variable numbering and names. However, to get a sense for the variable names, Figure 12-11 shows the tree structure for two variables, with the variable names being the long string of numbers shown at the bottom of the figure. Working with those numbers and the tree structure can be difficult at best. As a result, most people manage their networks using an NMS such as Cisco Prime. For perspective, you could use an SNMP manager and type MIB variable 1.3.6.1.4.1.9.2.1.58.0 and click a button to get that variable, to see the current CPU usage percentage from a Cisco router. However, most users of an NMS would much prefer to

ignore those details and have a simple graphical interface to ask for the same information, never having to know that 1.3.6.1.4.9.2.1.58.0 represents the router CPU utilization MIB variable.

Securing SNMP

SNMP supports a few security mechanisms, depending in part on the particular version. This section works through the options.

First, one strong method to secure SNMP is to use ACLs to limit SNMP messages to those from known servers only. SNMP agents on Cisco routers and switches support SNMP messages that flow in both IPv4 and IPv6 packets. The SNMP agent can configure an IPv4 ACL to filter incoming SNMP messages that arrive in IPv4 packets and an IPv6 ACL to filter SNMP messages that arrive in IPv6 packets.

Using an IPv4 and IPv6 ACL to secure an agent makes good sense. The only hosts that should be sending SNMP messages to the SNMP agent in a router or switch are the NMS hosts. Those NMS hosts seldom move and their IP addresses should be well known to the networking staff. It makes good sense to configure an ACL that permits packets sourced from the IP addresses of all NMS hosts, but no others.

As for the SNMP protocol messages, all versions of SNMP support a basic clear-text password mechanism, although none of those versions refer to the mechanism as using a password. SNMP Version 3 (SNMPv3) adds more modern security as well.

SNMPv1 defined clear-text passwords called SNMP *communities*. Basically, both the SNMP agent and the SNMP manager need prior knowledge of the same SNMP community value (called a *community string*). The SNMP Get messages and the Set message include the appropriate community string value, in clear text. If the NMS sends a Get or Set with the correct community string, as configured on the SNMP agent, the agent processes the message.

SNMPv1 defines both a read-only community and a read-write community. The *read-only (RO) community* allows Get messages, and the *read-write (RW) community* allows both reads and writes (Gets and Sets). Figure 12-12 shows the concepts. At steps 1 and 2, the agent is configured with particular RO and RW community strings, and the NMS configures the matching values. At step 3, the SNMP Get can flow with either community, but at Step 4, the Set Request must use the RW community.

Key Topic

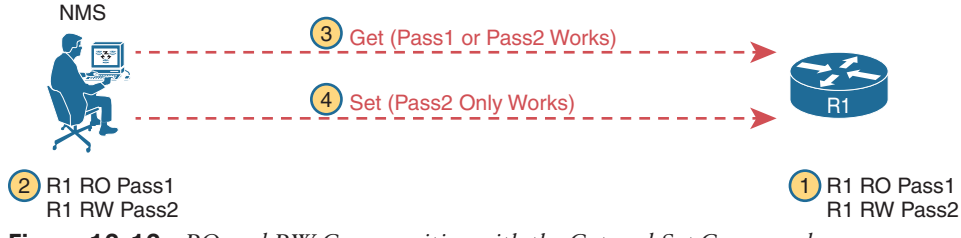


Figure 12-12 *RO and RW Communities with the Get and Set Commands*

SNMPv2, and the related Community-based SNMP Version 2 (SNMPv2c), added a wrinkle in naming but basically kept the same community security feature as SNMPv1 once the standards process completed. The original specifications for SNMPv2 did not include SNMPv1 communities; however, the marketplace still wanted communities, so an additional

RFC added the SNMPv1 communities mechanism back to SNMPv2. This updated RFC, "Community-based SNMPv2," came to be known simply as SNMPv2c. Vendors (including Cisco) implemented SNMPv2c; however, security was still relatively weak.

SNMPv3 arrived with much celebration among network administrators. Finally, security had arrived with the powerful network management protocol. SNMPv3 does away with communities and replaces them with the following features:

- **Message integrity:** This mechanism, applied to all SNMPv3 messages, confirms whether or not each message has been changed during transit.
- **Authentication:** This optional feature adds authentication with both a username and password, with the password never sent as clear text. Instead, it uses a hashing method like many other modern authentication processes.
- **Encryption (privacy):** This optional feature encrypts the contents of SNMPv3 messages so that attackers who intercept the messages cannot read their contents.

NOTE The CCNA 200-301 exam blueprint lists SNMP in one exam topic, with that exam topic reduced to "explain SNMP," with no requirement for configuration or verification skills. However, the previous version of the CCNA R&S certification did include SNMP configuration. Refer to Appendix D if interested in learning about SNMP configuration and verification.

FTP and TFTP

This final of three major sections of the chapter focuses on two topics: File Transfer Protocol (FTP) and Trivial File Transfer Protocol (TFTP). Both exist as TCP/IP protocols defined in RFCs. Both use a client and server model, in which the client connects to a server and then the client can copy files to the server or from the server. Both exist as a myriad of implementations of both client and server code, from command-line clients to apps with graphical interfaces, using the respective FTP or TFTP protocols behind the scenes.

This section discusses FTP and TFTP with two branches. The first section takes a practical view of the most common use of TFTP and FTP by network engineers while on the job: the job of updating IOS images. The process can make use of TFTP and FTP, so this section provides the basics. The second branch of this final major section then moves on to talk about FTP and TFTP in a much broader sense, with details about each protocol, their capabilities, and what capabilities each provides to any user.

Managing Cisco IOS Images with FTP/TFTP

IOS exists as a file—a single file—that the router then loads into RAM to use as its operating system. To better understand the process, you must understand a few more details about how IOS works. In particular, you need to understand the IO file system (IFS), which defines how IOS stores files (including the IOS file). The IOS image upgrade process occurs by copying new IOS files into the router and then booting the router with that new IOS.

The IOS File System

Every OS creates file systems to store files. A computer needs some type of permanent storage, but it needs more than just a place to store bytes. The OS organizes the storage into a file system, which includes directories, structure, and filenames, with the associated rules. By using a file system, the OS can keep data organized so the user and the applications can find the data later.

Every OS defines its own file system conventions. Windows OSs, for instance, use a left-leaning slash (\) in directory structures, like \Desktop\Applications. Linux and macOS use a right-leaning slash, for example, /Desktop. Each OS refers to physical disks slightly differently as well, and IOS is no different.

As for the physical storage, Cisco routers typically use flash memory, with no hard disk drive. Flash memory is rewriteable, permanent storage, which is ideal for storing files that need to be retained when the router loses power. Cisco purposefully uses flash memory rather than hard disk drives in its products because there are no moving parts in flash memory, so there is a smaller chance of failure as compared with disk drives. Some routers have flash memory on the motherboard. Others have flash memory slots that allow easy removal and replacement of the flash card, but with the intent that the card remain in the device most of the time. Also, many devices have USB slots that support USB flash drives.

For each physical memory device in the router, IOS creates a simple IOS file system and gives that device a name. Example 12-1 lists the surprisingly long list of IOS file systems. Note that entries of type *disk* and *usbflash* are the physical storage devices in that router. In this case, the router has one of two of the 2901's compact flash slots populated with a 256-MB flash card and one of the two USB flash slots populated with an 8-GB USB flash drive. Look at the size column and prefixes column in the output to find these devices, based on their types as *disk* and *usbflash*.

Example 12-1 *Cisco IOS File Systems on a Router*

```
R2# show file systems
File Systems:

       Size(b)       Free(b)      Type   Flags   Prefixes
             -             -    opaque      rw   archive:
             -             -    opaque      rw   system:
             -             -    opaque      rw   tmpsys:
             -             -    opaque      rw   null:
             -             -   network      rw   tftp:
*    256487424      49238016      disk      rw   flash0: flash:#
             -             -      disk      rw   flash1:
        262136        253220     nvram      rw   nvram:
             -             -    opaque      wo   syslog:
             -             -    opaque      rw   xmodem:
             -             -    opaque      rw   ymodem:
             -             -   network      rw   rcp:
             -             -   network      rw   pram:
             -             -   network      rw   http:
             -             -   network      rw   ftp:
             -             -   network      rw   scp:
             -             -    opaque      ro   tar:
             -             -   network      rw   https:
             -             -    opaque      ro   cns:
    7794737152    7483719680  usbflash      rw   usbflash0:
74503236 bytes copied in 187.876 secs (396555 bytes/sec)
```

The example lists 20 different IOS file systems in this case, but the router does not have 20 different physical storage devices. Instead, IOS uses these file systems for other purposes as well, with these types:

- **Opaque:** To represent logical internal file systems for the convenience of internal functions and commands
- **Network:** To represent external file systems found on different types of servers for the convenience of reference in different IOS commands
- **Disk:** For flash
- **Usbflash:** For USB flash
- **NVRAM:** A special type for NVRAM memory, the default location of the startup-config file

Many IOS commands refer to files in an IFS, but only some commands refer directly to the files by their formal names. The formal names use the prefix as seen in the far right column of Example 12-1. For instance, the command **more flash0:/wotemp/fred** would display the contents of file *fred* in directory */wotemp* in the first flash memory slot in the router. (The **more** command itself displays the contents of a file.) However, many commands use a keyword that indirectly refers to a formal filename, to reduce typing. For example:

- **show running-config** command: Refers to file system:running-config
- **show startup-config** command: Refers to file nvram:startup-config
- **show flash** command: Refers to default flash IFS (usually flash0:)

Upgrading IOS Images

One of the first steps to upgrade a router's IOS to a new version is to obtain the new IOS image and put it in the right location. Typically, Cisco routers have their IOS in one of the local physical file systems, most often in permanent flash. The only requirement is that the IOS be in some reachable file system—even if the file sits on an external server and the device loads the OS over the network. However, the best practice is to store each device's IOS file in flash that will remain with the device permanently.

Figure 12-13 illustrates the process to upgrade an IOS image into flash memory, using the following steps:

Step 1. Obtain the IOS image from Cisco, usually by downloading the IOS image from Cisco.com using HTTP or FTP.

Step 2. Place the IOS image someplace that the router can reach. Locations include TFTP or FTP servers in the network or a USB flash drive that is then inserted into the router.

Step 3. Issue the **copy** command from the router, copying the file into the flash memory that usually remains with the router on a permanent basis. (Routers usually cannot boot from the IOS image in a USB flash drive.)

Key Topic

Figure 12-13 *Copying an IOS Image as Part of the Cisco IOS Software Upgrade Process*

Copying a New IOS Image to a Local IOS File System Using TFTP

Example 12-2 provides an example of step 3 from Figure 12-13, copying the IOS image into flash memory. In this case, router R2, a 2901, copies an IOS image from a TFTP server at IP address 2.2.2.1.

Key Topic

Example 12-2 **copy tftp flash** *Command Copies the IOS Image to Flash Memory*

```
R2# copy tftp flash
Address or name of remote host []? 2.2.2.1
Source filename []? c2900-universalk9-mz.SPA.152-4.M1.bin
Destination filename [c2900-universalk9-mz.SPA.152-4.M1.bin ]?
Accessing tftp://2.2.2.1/c2900-universalk9-mz.SPA.152-4.M1.bin ...
Loading c2900-universalk9-mz.SPA.152-4.M1.bin from 2.2.2.1 (via GigabitEthernet0/1):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 97794040 bytes]


97794040 bytes copied in 187.876 secs (396555 bytes/sec)
R2#
```

The **copy** command does a simple task—copy a file—but the command also has several small items to check. It needs a few pieces of information from the user, so the command prompts the user for that information by showing the user some text and waiting for the user's input. The bold items in the example show the user's input. The router then has to check to make sure the copy will work. The command works through these kinds of questions:

1. What is the IP address or host name of the TFTP server?
2. What is the name of the file?
3. Ask the server to learn the size of the file, and then check the local router's flash to ask whether enough space is available for this file in flash memory.
4. Does the server actually have a file by that name?
5. Do you want the router to erase any old files in flash?

The router prompts you for answers to some of these questions, as necessary. For each question, you should either type an answer or press **Enter** if the default answer (shown in square brackets at the end of the question) is acceptable. Afterward, the router erases flash memory if directed, copies the file, and then verifies that the checksum for the file shows that no errors occurred in transmission.

NOTE Most people use the IOS filenames that Cisco supplies because these names embed information about the IOS image, like the version. Also, if you want to use the same destination filename as the source, avoid the mistake of typing "y" or "yes" to confirm the selection; instead, you would be setting the destination filename to "y" or "yes." Simply press **Enter** to confirm the selection listed in brackets.

You can view the contents of the flash file system to see the IOS file that was just copied by using a couple of commands. The **show flash** command shows the files in the default flash file system (flash0:), as seen at the top of Example 12-3. Below it, the more general **dir flash0:** command lists the contents of that same file system, with similar information. (You can use the **dir** command to display the contents of any local IFS.)

Example 12-3 *Command Copies the IOS Image to Flash Memory*

```
R4# show flash
-#- --length-- -----date/time------ path
1    104193476 Jul 21 2015 13:38:06 +00:00  c2900-universalk9-mz.SPA.154-3.M3.bin
3      3000320 Jul 10 2012 00:05:44 +00:00  cpexpress.tar
4         1038 Jul 10 2012 00:05:52 +00:00  home.shtml
5       122880 Jul 10 2012 00:06:02 +00:00  home.tar
6      1697952 Jul 10 2012 00:06:16 +00:00  securedesktop-ios-3.1.1.45-k9.pkg
7       415956 Jul 10 2012 00:06:28 +00:00  sslclient-win-1.1.4.176.pkg
8         1153 Aug 16 2012 18:20:56 +00:00  wo-lic-1
9     97794040 Oct 10 2014 21:06:38 +00:00  c2900-universalk9-mz.SPA.152-4.M1.bin

49238016 bytes available (207249408 bytes used)

R4# dir flash0:
Directory of flash0:/

   1  -rw-    104193476  Jul 21 2015 13:38:06 +00:00  c2900-universalk9-mz.SPA.154-3.
M3.bin
   3  -rw-     3000320  Jul 10 2012 00:05:44 +00:00  cpexpress.tar
   4  -rw-        1038  Jul 10 2012 00:05:52 +00:00  home.shtml
   5 -rw-       122880  Jul 10 2012 00:06:02 +00:00  home.tar
   6 -rw-      1697952  Jul 10 2012 00:06:16 +00:00  securedesktop-ios-3.1.1.45-k9.
pkg
   7 -rw-       415956  Jul 10 2012 00:06:28 +00:00  sslclient-win-1.1.4.176.pkg
   8 -rw-         1153  Aug 16 2012 18:20:56 +00:00  wo-lic-1
   9 -rw-     97794040  Oct 10 2014 21:06:38 +00:00  c2900-universalk9-mz.SPA.152-4.
M1.bin

256487424 bytes total (49238016 bytes free)
```

Pay close attention to the memory usage per file and for the IFS as shown in the example. The output lists the size in bytes for each file. Note that the IOS file is about 104 MB. Note that the size of the IOS file matches the size shown earlier in the TFTP transfer in Example 12-2. The end of each of the commands then lists the amount of space available for new files to be added to flash (one lists it as "bytes available"; the other as "bytes free"). However, that same ending line of each command shows slightly different information about usage: **show flash** lists the bytes used, whereas the **dir** command lists the total bytes (bytes used plus bytes free). Play around with the numbers in this example to make sure you know which command lists which particular total.

Verifying IOS Code Integrity with MD5

You download the IOS from Cisco, copy it to your router, and run it. Is it really the code from Cisco? Or did some nefarious attacker somehow get you to download a fake IOS that has a virus?

Cisco provides a means to check the integrity of the IOS file to prevent this type of problem. Figure 12-14 shows the basic mechanics of the process. First, when Cisco builds a new IOS image, it calculates and publishes an MD5 hash value for that specific IOS file. That is, Cisco uses as input the IOS file itself, runs the MD5 math algorithm against that file, producing a hex code. Cisco places that code at the download site for all to see. Then, you run that same MD5 math on your router against the IOS file on the router, using the IOS **verify** command. That command will list the MD5 hash as recalculated on your router. If both MD5 hashes are equal, the file has not changed.

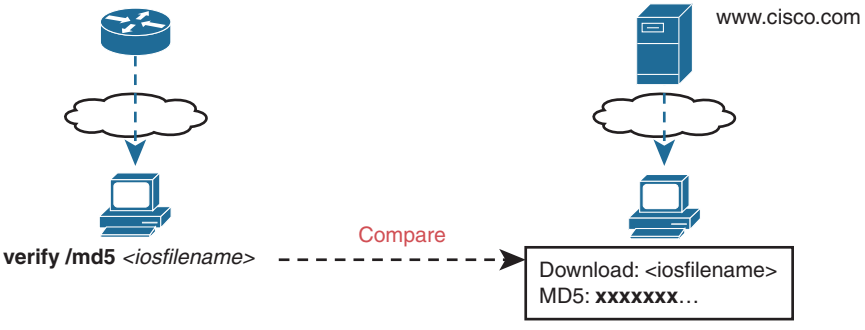


Figure 12-14 *MD5 Verification of IOS Images—Concepts*

The **verify /md5** command generates the MD5 hash on your router, as shown in Example 12-4. Note that you can include the hash value computed by Cisco as the last parameter (as shown in the example), or leave it off. If you include it, IOS will tell you if the locally computed value matches what you copied into the command. If you leave it out, the **verify** command lists the locally computed MD5 hash, and you have to do the picky character-by-character check of the values yourself.

Example 12-4 *Verifying Flash Memory Contents with the* **show flash** *Command*

```
R2# verify /md5 flash0:c2900-universalk9-mz.SPA.154-3.M3.bin a79e325e6c498b70829d4d
b0afba5041
.....................................................................................
.....................................................................................
…..MD5 of flash0:c2900-universalk9-mz.SPA.154-3.M3.bin Done!
Verified (flash0:c2900-universalk9-mz.SPA.154-3.M3.bin) = a79e325e6c498b70829d4d
b0afba5041
```

Copying Images with FTP

The networking world has many options for file transfer, several of which IOS supports for the transfer of files into and out of the IOS file systems that reside on the router. TFTP and FTP have been supported for the longest time, with more recent support added for protocols like Secure Copy Protocol (SCP), which uses the SSH File Transfer Protocol (SFTP). Table 12-3 lists some of the names of file transfer protocols that you might come across when working with routers.

Table 12-3 Common Methods to Copy Files Outside a Router

Method	Method (Full Name)	Encrypted?
TFTP	Trivial File Transfer Protocol	No
FTP	File Transfer Protocol	No
SCP	Secure Copy Protocol	Yes

To copy files with FTP, you follow the same kind of process you use with TFTP (see Example 12-5). You can follow the interactive prompts after using an EXEC command like **copy ftp flash**. However, the **copy** command allows you to use a URI for the source and/or destination, which lets you put most or all of the information in the command line itself. Each URI refers to the formal name of a file in the IFS.

Key Topic

Example 12-5 *Installing a New IOS with FTP*

```
R1# copy ftp://wendell:odom@192.168.1.170/c2900-universalk9-mz.SPA.155-2.T1.bin flash
Destination filename [c2900-universalk9-mz.SPA.155-2.T1.bin]?
Accessing ftp://192.168.1.170/c2900-universalk9-mz.SPA.155-2.T1.bin...
Loading c2900-universalk9-mz.SPA.155-2.T1.bin !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 107410736/4096 bytes]

107410736 bytes copied in 119.604 secs (898053 bytes/sec)
```

First, take a close look at the long URI in the command that begins with "ftp." The "ftp" part identifies the protocol, of course. After the //, the text references the username (wendell) and password (odom), as well as the FTP server's IP address. After the single / comes the filename on the server.

Although the command is long, it has only two parameters, with the long first parameter and the short keyword **flash** as the second parameter. The **copy** command lists the source location as the first parameter and the destination as the second. The destination in this case, **flash**, is a keyword that refers to the default flash, typically flash0:, but it does not identify a specific filename. As a result, IOS prompts the user for a specific destination filename, with a default (in brackets) to keep the source filename. In this case, the user just pressed Enter to accept the default. To avoid being prompted at all, the command could have listed **flash:c2900-universalk9-mz.SPA.155-2.T1.bin** as that second parameter, fully defining the destination file.

Finally, with another twist, you can configure the FTP username and password on the router so that you do not have to include them in the **copy** command. For instance, the global configuration commands **ip ftp username wendell** and **ip ftp password odom** would have configured those values. Then the **copy** command would have begun with **copy ftp://192.168.1.170/...**, omitting the username:password in the command, without needing to then prompt the user for the username and password.

That completes the examples of showing how to copy IOS files into a router using TFTP and FTP. The exam topics happen to mention TFTP and FTP, but not the IOS upgrade

process, so the text now turns away from the IOS upgrade process to focus more on TFTP and FTP. However, there are a few more steps to complete to upgrade IOS, such as configuring the boot system command and reloading the router. If you want to read about the rest of the IOS upgrade process or other related tasks like managing configuration files and performing password recovery, refer to this book's Appendix F, "Previous Edition ICND1 Chapter 35: Managing IOS Files."

However, to complete the IOS upgrade process, you need to finish a few more required steps.

The FTP and TFTP Protocols

The IOS **copy** command, when using the **tftp** or **ftp** keyword, makes the command act as a client. The client connects to a TFTP or FTP server and then attempts to transfer the file. In the examples from the IOS, that **copy** command copied the file from the server into the client device (a router).

The rest of this section examines what happens behind the scenes in that process, with a closer look at both FTP and TFTP as protocols and tools.

FTP Protocol Basics

FTP has long been a core Internet protocol, serving as the primary file transfer protocol for several decades. RFC 959, which standardizes FTP, dates back to 1985. FTP uses TCP as its transport protocol, relying on TCP to provide an error-free in-order deliver of data so that the FTP application knows that each file transfer creates an exact copy of the file with no omissions. FTP uses well-known TCP port 21 and in some cases also well-known port 20.

As for normal operation, FTP uses a client/server model for file transfer, as shown in the example in Figure 12-15. The figure shows the major steps but not every message. For instance, step 1 shows host A creating a TCP connection to the server (which takes the usual three TCP messages). Step 2 represents the exchange that allows the server to authenticate the client. Step 3 shows the idea that, once authenticated, the client and server can send FTP commands over the connection to tell the other device what to do.

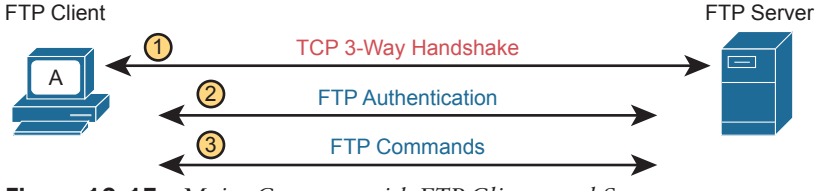


Figure 12-15 *Major Concepts with FTP Clients and Servers*

The commands that flow over this initial TCP connection—called the FTP control connection—define the kinds of functions supported by FTP. Those commands allow the client to navigate around the directory structures of the server, list files, and then transfer files from the server (FTP GET) or to the server (FTP PUT). Following is a summary of some of the FTP actions:

Key Topic

- **Navigate directories:** List the current directory, change the current directory to a new directory, go back to the home directory, all on both the server and client side of the connection.
- **Add/remove directories:** Create new directories and remove existing directories on both the client and server.
- **List files:** List files on both the client and server.
- **File transfer:** Get (client gets a copy of the file from the server), Put (client takes a file that exists on the client and puts a copy of the FTP server).

While many OSs support command-line FTP clients, which require you to learn the various FTP commands and use those from the command line, most users instead use an FTP client app that issues the FTP commands behind the scenes. Clients typically display files on the local system as well as the server with a user interface similar to a typical file browser on a desktop OS (for instance, Windows Explorer, macOS Finder). Figure 12-16 shows a sample user interface from the Filezilla FTP client (Filezilla-project.org).

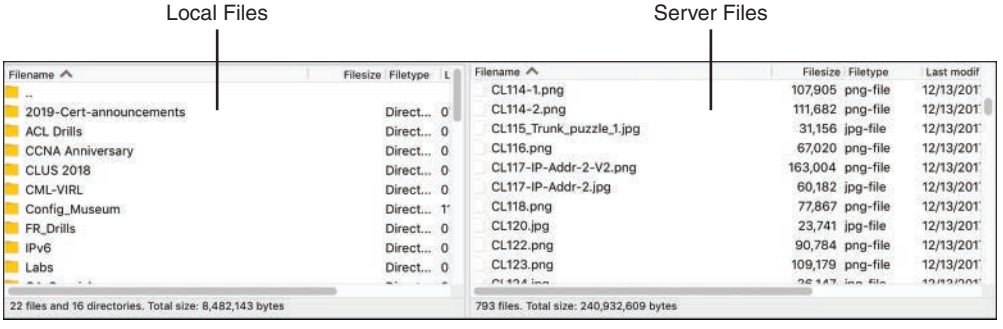


Figure 12-16 *FTP Client Example with Filezilla*

The client application in Figure 12-16 lists the client computer's local file system on the left and the FTP server's file system on the right. The user can click on the right to change directories, much like using any app that browses a file system, with FTP performing the commands behind the scenes. The user can also drag and drop files from the left to the right to put a file on the server, or vice versa to get a file from the server.

The FTP server can be a server application installed and managed by others, or you can install or enable an FTP server for your own use. For instance, a network engineer might install an FTP server application on her laptop for use in upgrading IOS files, while the IT staff may keep an FTP server available 24/7 for all employees of the company to use. A simple Internet search can show a variety of FTP server applications that run on the common desktop OSs. Additionally, both Windows 10 and macOS come with an FTP or FTPS (FTP Secure) server option built into the OS; all you have to do is enable it. (The Linux distributions all have FTP servers available via simple downloads.)

Once installed, the server can be configured with a variety of settings. For instance, the server needs to specify which users can access the server, so it can use the same login credentials allowed for the host where it resides or specify other credentials. It can specify the directories that each user can access, and whether the user has read-only or read-write access.

FTP Active and Passive Modes

FTP can operate in either active or passive mode. The choice of mode may impact whether the TCP client can or cannot connect to the server and perform normal functions. The user

at the FTP client can choose which mode to use, so this section works through the underlying details to explain why FTP passive mode may be the more likely option to work.

First, note that FTP uses two types of TCP connections:

Key Topic

- **Control Connection:** Used to exchange FTP commands
- **Data Connection:** Used for sending and receiving data, both for file transfers and for output to display to a user

Given the two roles, when a client connects to an FTP server, the client first creates the FTP control connection as shown in Figure 12-17. The server listens for new control connections on its well-known port 21; the client allocates any new dynamic port (49222 in this case) and creates a TCP connection to the server.

Key Topic

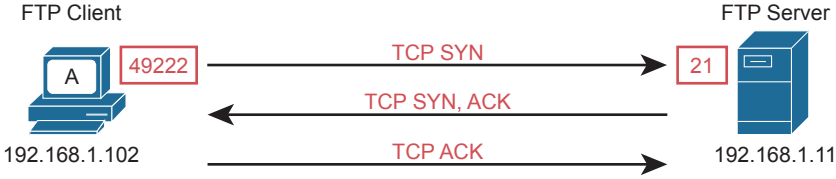


Figure 12-17 *FTP Client Creates an FTP Control Connection*

After creating the TCP connection, the user authenticates to the FTP server and takes some actions. Some of those actions require only the control connection, but eventually the user will take an action (like getting a file) that requires a data connection. When that happens, to create the FTP data connection, the client will either use active mode or passive mode, as shown in the next two examples.

Figure 12-18 shows an example of what happens in active mode. Following the steps in the figure:

1. The FTP client allocates a currently unused dynamic port and starts listening on that port.
2. The client identifies that port (and its IP address) to the FTP server by sending an FTP **PORT** command to the server.
3. The server, because it also operates in active mode, expects the **PORT** command; the server reacts and initiates the FTP data connection to the client's address (192.168.1.102) and port (49333).

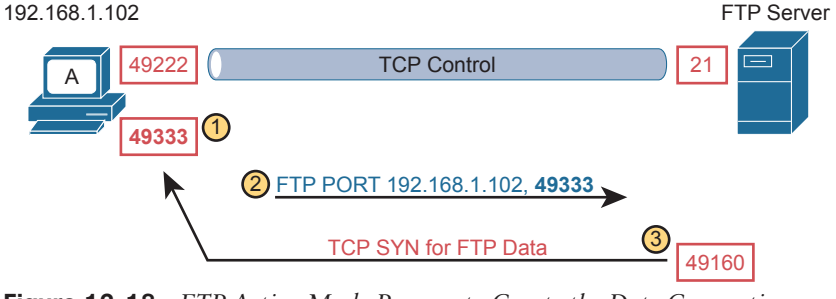


Figure 12-18 *FTP Active Mode Process to Create the Data Connection*

Active mode works well with both the FTP client and server sitting inside the same enterprise network. When within the same network, typically no NAT function and no firewall sits between the two. However, if the FTP client sits in an enterprise network, and the FTP server resides somewhere in the Internet, an active mode connection typically fails. Most firewalls do not allow Internet-based hosts to initiate TCP connections to hosts inside the enterprise without a specific firewall rule allowing connections to a known port, and in this case, the FTP client allocates any available port number. For instance, in Figure 12-18, the TCP connection (step 3) would be discarded by a firewall.

NOTE FTP reserves two well-known ports: port 21 for control connections and port 20 for data connections. However, due to changes to FTP over the years, FTP often uses other TCP ports for the TCP data connection, as seen in the examples in this chapter.

Passive mode helps solve the firewall restrictions by having the FTP client initiate the FTP data connection to the server. However, passive mode does not simply cause the FTP client to connect to a well-known port on the server; it requires more exchanges of port numbers to use between the server and client, as shown in Figure 12-19, with these steps:

1. The FTP client changes to use FTP passive mode, notifying the server using the FTP **PASV** command.
2. The server chooses a port to listen on for the upcoming new TCP connection, in this case TCP port 49444.
3. The FTP notifies the FTP client of its IP address and chosen port with the FTP **PORT** command.
4. The FTP client opens the TCP data connection to the IP address and port learned at the previous step.

Key Topic

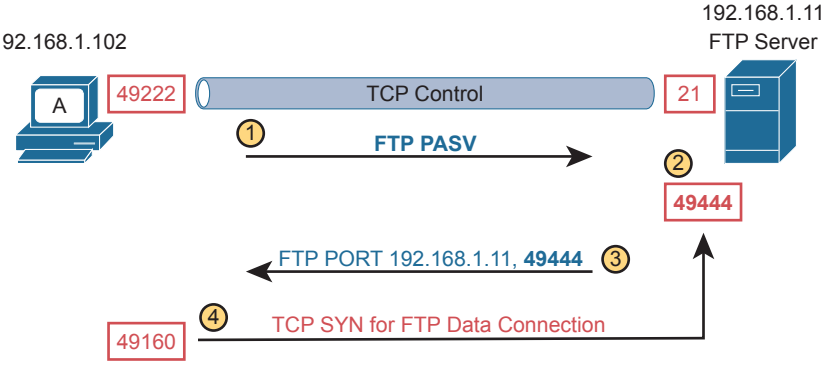


Figure 12-19 *FTP Passive Mode Process to Create the Data Connection*

FTP over TLS (FTP Secure)

FTP, defined in RFC 959 back in 1985, has some shortcomings with security. As originally defined, it does include the ability to use usernames and passwords for authentication and authorization; however, the username/password flows as clear text. Additionally, all data transfers flow as clear text.

Over the years, several RFCs defined security improvements for FTP. Those new features include using digital certificates for authentication as well as using Transport Layer Security (TLS) to encrypt all data (including usernames/passwords). Fast forward to today and many

of those features converge into what most FTP clients and servers support as FTP over TLS or as FTP Secure (FTPS).

With FTPS, the client and server still use FTP commands and still use both a control and data connection. However, FTPS encrypts both the control and data connections with TLS, including the exchange of the usernames and passwords. FTPS includes a few variations, including the FTPS explicit mode process shown in Figure 12-20:

1. The client creates the FTP control TCP connection to server well-known port 21.
2. The client initiates the use of TLS in the control connection with the FTP **AUTH** command.
3. When the user takes an action that requires an FTP data connection, the client creates an FTP data TCP connection to server well-known port 21.
4. The client initiates the use of TLS in the data connection with the FTP **AUTH** command.

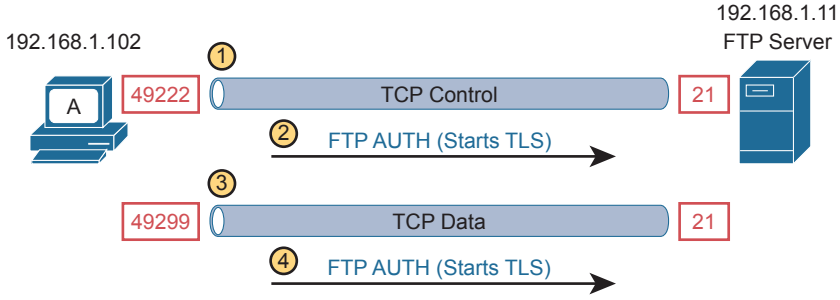


Figure 12-20 *FTPS Explicit Mode Control and Data Connection Establishment*

In contrast, the implicit mode process begins with a required TLS connection, with no need for an FTP AUTH command, using well-known ports 990 (for the control connection) and 989 (for the data connection).

NOTE SSH File Transfer Protocol (SFTP) is a different protocol than FTPS. SFTP uses SSH to encrypt file transfers over an SSH connection. However, the acronym SFTP does not refer to a secure version of FTP.

TFTP Protocol Basics

FTP has a role as a general file transfer tool for any user, with a good number of FTP client application options available. TFTP plays a much smaller role as a tool for the average user, but it does play a more useful role for IT support staff.

For the basics, Trivial File Transfer Protocol uses UDP well-known port 69. Because it uses UDP, TFTP adds a feature to check each file for transmission errors by using a checksum process on each file after the transfer completes.

Key Topic

The word *trivial* in the name refers to its relatively small number of features, meant to be an advantage by making the tool lightweight. For instance, it supports far fewer commands than FTP (fewer functions), meaning that the code requires less space to install, which can be useful for devices with limited memory. TFTP can Get and Put files, but it includes no commands to change directories, create/remove directories, or even to list files on the

server. TFTP does not support even simple clear-text authentication. In effect, if a TFTP server is running, it should accept requests from any TFTP client.

Ideally, TFTP has its best use as a temporary tool for quick file transfers in a controlled environment, particularly when the data itself does not have to be secure. For instance, imagine this scenario:

1. A network engineer keeps all router and switch IOS images in a folder.
2. The engineer enables a TFTP server on her laptop as needed; otherwise, the TFTP server remains disabled.
3. The engineer connects her laptop to a LAN and enables the TFTP server long enough to transfer IOS images into or out of a few devices.
4. If the engineer forgets to disable TFTP, the only risk is that someone may copy an IOS image—an image that is already available from Cisco.com to any customer.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 12-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 12-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review Command Tables		Book

Review All the Key Topics

Key Topic

Table 12-5 Key Topics for Chapter 12

Key Topic Element	Description	Page Number
List	Common characteristics of all FHRPs	260
Table 12-2	Comparisons of HSRP, VRRP, GLBP	260
Figure 12-5	HSRP concepts	261
Figure 12-6	HSRP failover results	262
Figure 12-9	The SNMP Get Request and Get Response message flow	265
Figure 12-10	SNMP notification with SNMP Trap messages	265
Figure 12-12	The use of SNMP RO and RW communities with SNMP Get and Set	267

Key Topic Element	Description	Page Number
List	SNMP security benefits	268
Figure 12-13	Process of upgrading IOS using TFTP	270
Example 12-2	Example of using TFTP to load new IOS	271
Example 12-5	Example of using FTP to load new IOS	274
List	FTP functions	276
List	FTP data and control connections	277
Figure 12-17	FTP Control connection establishment	277
Figure 12-19	FTP data connection establishment in passive mode	278
Paragraph	Description of limited functions of TFTP	279

Key Terms You Should Know

First Hop Redundancy Protocol (FHRP), Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), Gateway Load Balancing Protocol (GLBP), virtual IP address, virtual MAC address, HSRP active, HSRP standby, Simple Network Management Protocol (SNMP), SNMP community, read-only community, read-write community, notification community, SNMP Get, SNMP Set, SNMP Trap, SNMP Inform, Management Information Base (MIB), SNMPv2c, SNMPv3, Network Management System (NMS), SNMP manager, SNMP agent, IOS image, flash memory, IOS file system, code integrity, TFTP, FTP, FTP control connection, FTP data connection, FTP over TLS

Command References

Tables 12-6 and 12-7 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 12-6 Chapter 12 Configuration Command Reference

Command	Description
boot system flash [*flash-fs*:] [*filename*]	Global command that identifies the location of an IOS image in flash memory
boot system {**tftp** | **ftp**} *filename* [*ip-address*]	Global command that identifies an external server, protocol, and filename to use to load an IOS from an external server
ip ftp username *name*	Global command to define the username used when referencing the **ftp:** IOS file system but not supplying a username
ip ftp password *pass*	Global command to define the password used when referencing the **ftp:** IOS file system but not supplying a password

Table 12-7 Chapter 12 EXEC Command Reference

Command	Description
copy *from-location to-location*	Enable mode EXEC command that copies files from one file location to another. Locations include the startup-config and running-config files, files on TFTP and RPC servers, and flash memory.
show flash	Lists the names and size of the files in flash memory, and notes the amount of flash memory consumed and available.
dir *filesystem:* **dir** *filesystem:directory*	Lists the files in the referenced file system or file system directory.
verify /md5 *filesystem:name [MD5-bash]*	Performs an MD5 hash of the referenced file and displays the results. If listed, the command compares the MD5 hash in the command with the results of performing MD5 on the local file.

This page intentionally left blank

Part III Review

Keep track of your part review progress with the checklist shown in Table P3-1. Details on each task follow the table.

Table P3-1 Part III Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Review Videos		

Repeat All DIKTA Questions

For this task, use the PTP software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Use Per-Chapter Interactive Review Elements

Using the companion website, browse through the interactive review elements, such as memory tables and key term flashcards, to review the content from each chapter.

Labs

Depending on your chosen lab tool, here are some suggestions for what to do in the lab:

Pearson Network Simulator: If you use the full Pearson CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog Config Labs: The author's blog (https://blog.certskills.com) includes a series of configuration-focused labs that you can do on paper, each in 10–15 minutes. Review and perform the labs for this part of the book by using the menus to navigate to the per-chapter content and then finding all config labs related to that chapter. (You can see more detailed instructions at https://blog.certskills.com/config-labs.)

Other: If using other lab tools, here are a few suggestions: All the exam topics in Part III that include the word *configure* exist in Chapters 9 and 10, so focus on those chapters. Those chapters touch on CDP/LLDP, NTP, syslog, and NAT/PAT.

Watch Videos

Part III's Chapter 11 includes a mention of a video about QoS Classification and Marking. You can find a link to view that video in the section for videos in the companion website for this book.

Part IV turns the attention away from the concept-configure-verify approach needed for many of the topics seen earlier in this book and in *CCNA 200-301 Official Cert Guide, Volume 1*. Instead, this part collects topics that will be presented more from an architecture and design perspective. In fact, the CCNA 200-301 exam organizes six exam topics with this same approach, all listed under exam topic 1.2 "Describe characteristics of network topology architectures." The chapters in this part examine most of those topics.

First, Chapter 13 revisits LAN switching, which was covered to some depth in Volume 1. This chapter discusses campus LAN design concepts and terminology, like the *2 tier* and *3 tier* terms listed in the exam topics. This chapter also discusses how to supply power over that LAN infrastructure using Power over Ethernet (PoE), as well as the term *small office/home office* (SOHO).

CCNA 200-301 mentions WAN as an end to itself in one exam topic within the context of topology and architecture. Chapter 14 takes that thread and presents three major WAN architectures, going beyond the concepts you need to know to support the simple WAN cases used in the examples throughout both books so far. Those topics include MPLS VPN WANs, Ethernet WANs, and Internet VPNs.

Chapter 15 completes the architecture-focused chapters with a discussion of cloud architectures. This chapter begins by defining basic concepts and terms related to data centers and cloud and closes with design discussions that show packet flows in a public cloud environment.

Part IV

Network Architecture

CHAPTER 13

LAN Architecture

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.2 Describe characteristics of network topology architectures

1.2.a 2 tier

1.2.b 3 tier

1.2.e Small office/home office (SOHO)

1.3 Compare physical interface and cabling types

1.3.c Concepts of PoE

By now you have learned a lot about Ethernet and Ethernet switches. You have learned how individual links work, with cabling and duplex settings as well as framing. You know how addresses work and how switches forward frames based on those addresses. You have seen how switches deal with redundancy, using STP/RSTP and collecting links into EtherChannels. And here in Volume 2, you have learned about a variety of security features available for switches, including Dynamic ARP Inspection, DHCP Snooping, and ARP Inspection.

What the earlier discussions of individual features do not do to any great extent is discuss architecture and design. You now know how switches work, but why would you connect switches in one topology versus another? If you could connect switches in two different topologies, why would you prefer one over the other? This chapter examines a few such design questions, specifically the topic areas mentioned in the CCNA 200-301 exam topics. (Note that the CCNA 200-301 exam does not include a comprehensive look at LAN design issues, but one of the current CCNP Enterprise exams does.)

This chapter covers three specific topics that have design-related considerations. The first section looks at the topology of a wired Ethernet LAN and the design terms *two tier* and *three tier*, which describe how many switch layers exist between the endpoints and the devices that lead out of the campus to some other site. Following that, the second section examines small office/home office (SOHO) LANs and how they differ from enterprise LANs. The final section introduces the concepts behind Power over Ethernet (PoE), along with the reasons why LAN design activities need to consider PoE.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 13-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Analyzing Campus LAN Topologies	1–3
Small Office/Home Office	4
Power over Ethernet	5–6

1. In a two-tier campus LAN design, which of the following are typically true of the topology design? (Choose two answers.)
   a. The design uses a full mesh of links between access and distribution switches.
   b. The design uses a partial mesh of links between access and distribution switches.
   c. The design uses a partial mesh of links between the distribution and core switches.
   d. The end-user and server devices connect directly to access layer switches.
2. In a three-tier campus LAN design, which of the following are typically true of the topology design? (Choose two answers.)
   a. The design uses a partial mesh of links between access and distribution switches.
   b. The design uses a full mesh of links between access and distribution switches.
   c. The design uses a partial mesh of links between the distribution and core switches.
   d. The end-user and server devices connect directly to distribution layer switches.
3. Which one answer gives the strongest match between one part of a typical three-tier design with the idea behind the listed generic topology design term?
   a. The access layer looks like a partial mesh.
   b. The distribution layer looks like a full mesh.
   c. The distribution layer looks like a hybrid design.
   d. The access layer looks like a star design.
4. Which answers list criteria typical of a SOHO network? (Choose two answers.)
   a. The AP functions using standalone mode.
   b. The AP functions using a split-MAC architecture using a WLC.
   c. A single networking device implements the router, switch, AP, and firewall functions.
   d. A separate networking device implements each function (router, switch, AP, and firewall).
5. Which answer describes how a LAN switch dynamically chooses the initial power level to apply to a UTP cable with PoE?
   a. Autonegotiation
   b. CDP
   c. LLDP
   d. Preconfigured values

6. Which of the following refer to standards that deliver power over all four pairs in a UTP cable? (Choose two answers.)

   a. PoE
   b. UPoE
   c. PoE+
   d. UPoE+

Foundation Topics

Analyzing Campus LAN Topologies

The term *campus LAN* refers to the LAN created to support the devices in a building or in multiple buildings in somewhat close proximity to one another. For example, a company might lease office space in several buildings in the same office park. The network engineers can then build a campus LAN that includes switches in each building, plus Ethernet links between the switches in the buildings, to create a larger campus LAN.

When planning and designing a campus LAN, the engineers must consider the types of Ethernet available and the cabling lengths supported by each type. The engineers also need to choose the speeds required for each Ethernet segment. In addition, some thought needs to be given to the idea that some switches should be used to connect directly to end-user devices, whereas other switches might need to simply connect to a large number of these end-user switches. Finally, most projects require that the engineer consider the type of equipment that is already installed and whether an increase in speed on some segments is worth the cost of buying new equipment.

This first of three major sections of the chapter discusses the topology of a campus LAN design. Network designers do not just plug in devices to any port and connect switches to each other in an arbitrary way, like you might do with a few devices on the same table in a lab. Instead, there are known better ways to design the topology of a campus LAN, and this section introduces some of the key points and terms

Two-Tier Campus Design (Collapsed Core)

To sift through all the requirements for a campus LAN, and then have a reasonable conversation about it with peers, most Cisco-oriented LAN designs use some common terminology to refer to the design. For this book's purposes, you should be aware of some of the key campus LAN design terminology.

The Two-Tier Campus Design

Figure 13-1 shows a typical design of a large campus LAN, with the terminology included in the figure. This LAN has around 1000 PCs connected to switches that support around 25 ports each. Explanations of the terminology follow the figure.

Cisco uses three terms to describe the role of each switch in a campus design: *access*, *distribution*, and *core*. The roles differ based on whether the switch forwards traffic from user devices and the rest of the LAN (access), or whether the switch forwards traffic between other LAN switches (distribution and core).

Key Topic

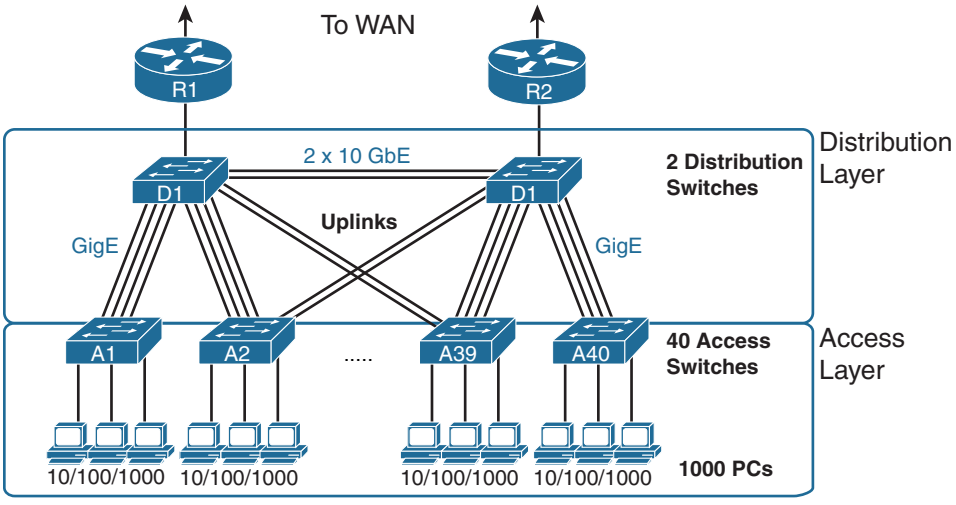


Figure 13-1 *Campus LAN with Design Terminology Listed*

Access switches connect directly to end users, providing user device access to the LAN. Access switches normally send traffic to and from the end-user devices to which they are connected and sit at the edge of the LAN.

Distribution switches provide a path through which the access switches can forward traffic to each other. By design, each of the access switches connects to at least one distribution switch, typically to two distribution switches for redundancy. The distribution switches provide the service of forwarding traffic to other parts of the LAN. Note that most designs use at least two uplinks to two different distribution switches (as shown in Figure 13-1) for redundancy.

The figure shows a two-tier design, with the tiers being the access tier (or layer) and the distribution tier (or layer). A two-tier design solves two major design needs:

- Provides a place to connect end-user devices (the access layer, with access switches)
- Connects the switches with a reasonable number of cables and switch ports by connecting all 40 access switches to two distribution switches

NOTE The terms two-tier and 2-tier are synonyms, as are the terms three-tier and 3-tier. Cisco happens to use the versions of these terms with numerals in the exam topics.

Topology Terminology Seen Within a Two-Tier Design

The networking world uses several common terms about LAN and WAN topology and design including these:

Star: A design in which one central device connects to several others, so that if you drew the links out in all directions, the design would look like a star with light shining in all directions.

Full mesh: For any set of network nodes, a design that connects a link between each pair of nodes.

Partial mesh: For any set of network nodes, a design that connects a link between some pairs of nodes, but not all. In other words, a mesh that is not a full mesh.

Hybrid: A design that combines topology design concepts into a larger (typically more complex) design.

Armed with those formal definitions, note that the two-tier design is indeed a hybrid design that uses both a star topology at the access layer and a partial mesh at the distribution layer. To see why, consider Figure 13-2. It redraws a typical access layer switch, but instead of putting the PCs all below the switch, it spreads them around the switch. Then on the right, a similar version of the same drawing shows why the term *star* might be used—the topology looks a little like a child's drawing of a star.

Key Topic

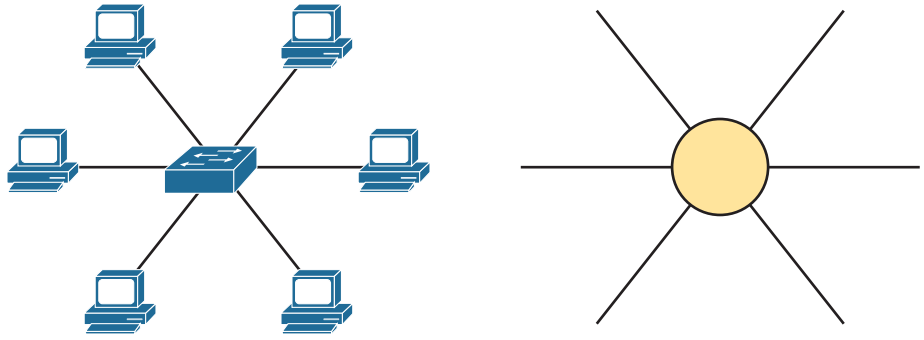


Figure 13-2 *The Star Topology Design Concept in Networking*

The distribution layer creates a partial mesh. If you view the access and distribution switches as nodes in a design, some nodes have a link between them, and some do not. Just refer to Figure 13-1 and note that, by design, none of the access layer switches connect to each other.

Finally, a design could use a full mesh. However, for a variety of reasons beyond the scope of the design discussion here, a campus design typically does not need to use the number of links and ports required by a full mesh design. However, just to make the point, first consider how many links and switch ports would be required for a single link between nodes in a full mesh, with six nodes, as shown in Figure 13-3.

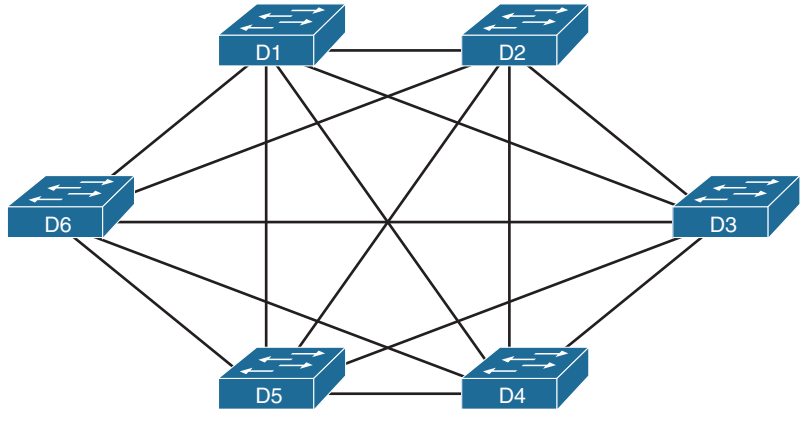


Figure 13-3 *Using a Full Mesh at the Distribution Layer, 6 Switches, 15 Links*

Answers to the "Do I Know This Already?" quiz:

1 B, D **2** A, C **3** D **4** A, C **5** A **6** B, D

Even with only six switches, a full mesh would consume 15 links (and 30 switch ports—two per link).

Now think about a full mesh at the distribution layer for a design like Figure 13-1, with 40 access switches and two distribution switches. Rather than drawing it and counting it, the number of links is calculated with this old math formula from high school: $N(N - 1) / 2$, or in this case, 42 * 41 / 2 = 861 links, and 1722 switch ports consumed among all switches.

For comparison's sake, the partial mesh design of Figure 13-1, with a pair of links from each access switch to each distribution switch, requires only 160 links and a total of 320 ports among all switches.

Three-Tier Campus Design (Core)

The two-tier design of Figure 13-1, with a partial mesh of links at the distribution layer, happens to be the most common campus LAN design. It also goes by two common names: a two-tier design (for obvious reasons) and a collapsed core (for less obvious reasons). The term *collapsed core* refers to the fact that the two-tier design does not have a third tier, the core tier. This next topic examines a three-tier design that does have a core, for perspective.

Imagine your campus has just two or three buildings. Each building has a two-tier design inside the building, with a pair of distribution switches in each building and access switches spread around the building as needed. How would you connect the LANs in each building? Well, with just a few buildings, it makes sense to simply cable the distribution switches together, as shown in Figure 13-4.

Key Topic

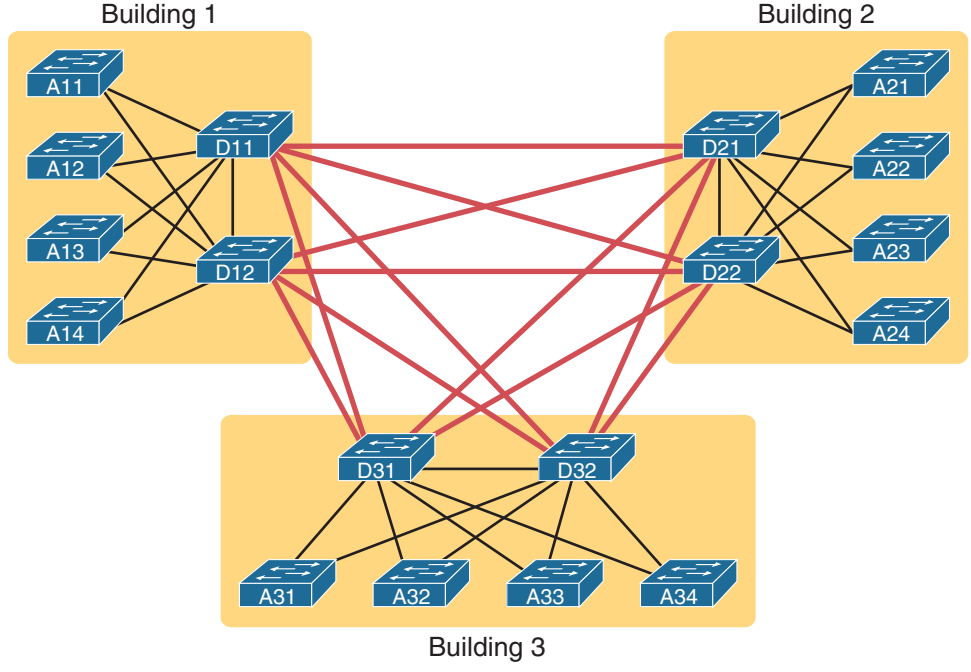


Figure 13-4 *Two-Tier Building Design, No Core, Three Buildings*

The design in Figure 13-4 works well, and many companies use this design. Sometimes the center of the network uses a full mesh, sometimes a partial mesh, depending on the availability of cables between the buildings.

However, a design with a third tier (a core tier) saves on switch ports and on cables in larger designs. And note that with the links between buildings, the cables run outside, are often more expensive to install, and are almost always fiber cabling with more expensive switch ports, so conserving the number of cables used between buildings can help reduce costs.

A three-tier core design, unsurprisingly at this point, adds a few more switches (core switches), which provide one function: to connect the distribution switches. Figure 13-5 shows the migration of the Figure 13-4 collapsed core (that is, a design without a core) to a three-tier core design.

Key Topic

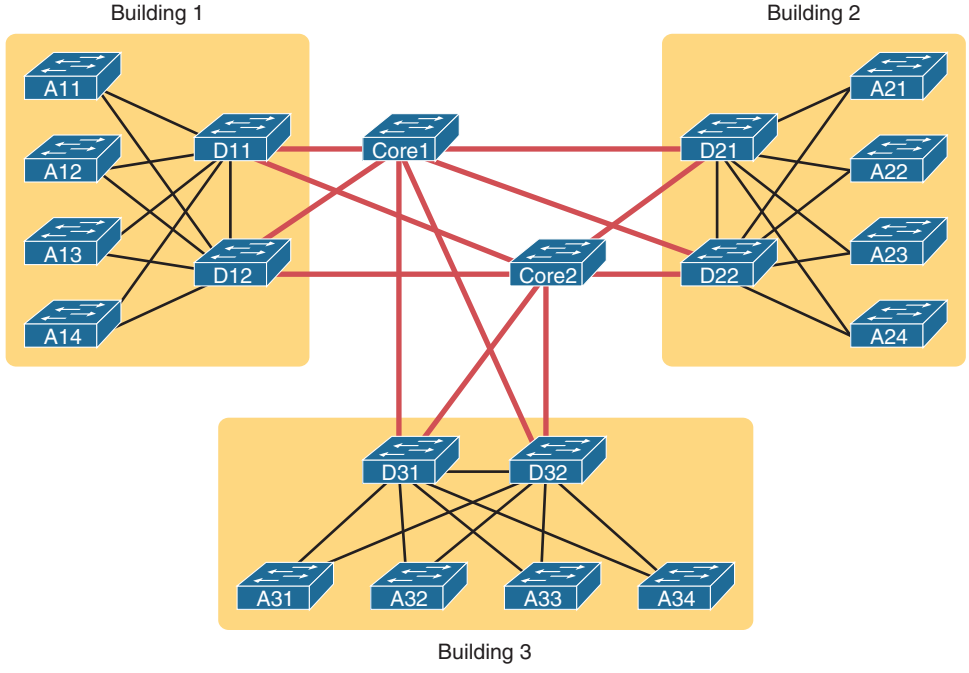


Figure 13-5 *Three-Tier Building Design (Core Design), Three Buildings*

NOTE The core switches sit in the middle of the figure. In the physical world, they often sit in the same room as one of the distribution switches, rather than in some purpose-built room in the middle of the office park. The figure focuses more on the topology rather than the physical location.

By using a core design, with a partial mesh of links in the core, you still provide connectivity to all parts of the LAN and to the routers that send packets over the WAN, just with fewer links between buildings.

The following list summarizes the terms that describe the roles of campus switches:

Key Topic

- **Access:** Provides a connection point (access) for end-user devices. Does not forward frames between two other access switches under normal circumstances.
- **Distribution:** Provides an aggregation point for access switches, providing connectivity to the rest of the devices in the LAN, forwarding frames between switches, but not connecting directly to end-user devices.
- **Core:** Aggregates distribution switches in very large campus LANs, providing very high forwarding rates for the larger volume of traffic due to the size of the network.

Topology Design Terminology

To close the discussion of Enterprise LAN topology, the next topic applies some of the generic topology terms to a typical two-tier design.

Consider Figure 13-6, which shows a few of the terms. First, on the left, drawings often show access switches with a series of cables, parallel to each other. However, the combinations of an access switch and its access links is often called a *star topology*. Why? Look at the redrawn access switch in the center of the figure, with the cables radiating out from the center. It does not look like a real star, but it looks a little like a child's drawing of a star, hence the term *star topology*.

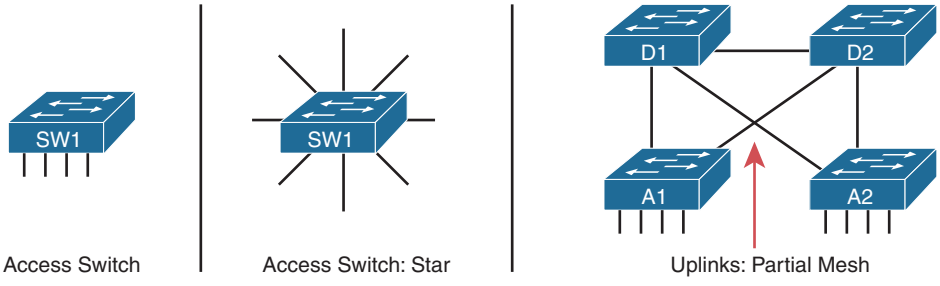


Figure 13-6 *LAN Design Terminology*

The right side of the figure repeats a typical two-tier design, focusing on the mesh of links between the access and distribution switches. Any group of nodes that connect with more links than a star topology is typically called a *mesh*. In this case, the mesh is a *partial mesh*, because not all nodes have a direct link between each other. A design that connects all nodes with a link would be a *full mesh*.

Real networks make use of these topology ideas, but often a network combines the ideas together. For instance, the right side of Figure 13-6 combines the star topology of the access layer with the partial mesh of the distribution layer. So you might hear these designs that combine concepts called a *hybrid design*.

Small Office/Home Office

Now that you know more about design choices and terms for an enterprise LAN, this next section examines one particular type of smaller LAN: the small office/home office (SOHO) LAN. SOHO refers to designs and implementations that have such a small volume of requirements—few switch ports, few APs, few routers and WAN links—that the design differs significantly. The term itself refers to the two most common cases: a user who works

from home or a small office with a small number of workers and devices. This next short topic points out a few of the highlights that make a SOHO network different from an enterprise network.

First, as a reminder, the IEEE defines both Ethernet LANs and wireless LANs (WLANs). In case it was not obvious yet, all Ethernet standards use cables—that is, Ethernet defines wired LANs. The IEEE 802.11 working group defines wireless LANs, also called Wi-Fi per a trademarked term from the Wi-Fi Alliance (www.wi-fi.org), a consortium that helps encourage wireless LAN development in the marketplace.

Most of you have used Wi-Fi, and may use it daily. Some of you may have set it up at home, with a basic setup as shown in Figure 13-7. In a home, you probably used a single consumer device called a *wireless router*. One side of the device connects to the Internet, while the other side connects to the devices in the home. In the home, the devices can connect either with Wi-Fi or with a wired Ethernet cable.

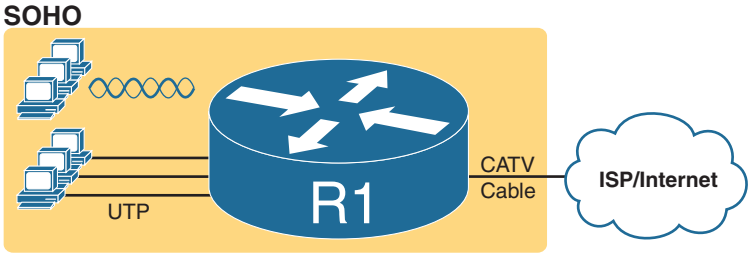


Figure 13-7 *A Typical Home Wired and Wireless LAN*

While the figure shows the hardware as a single router icon, internally, that one wireless router acts like separate devices you would find in an enterprise campus:

Key Topic

- An Ethernet switch, for the wired Ethernet connections
- A wireless access point (AP), to communicate with the wireless devices and forward the frames to/from the wired network
- A router, to route IP packets to/from the LAN and WAN (Internet) interfaces
- A firewall, which often defaults to allow only clients to connect to servers in the Internet, but not vice versa

Figure 13-8 repeats the previous figure, breaking out the internal components as if they were separate physical devices, just to make the point that a single consumer wireless router acts like several different devices.

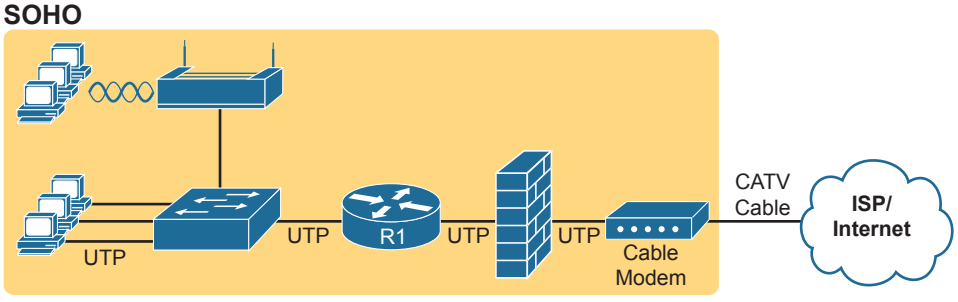


Figure 13-8 *A Representation of the Functions Inside a Consumer Wireless Routing Product*

In a SOHO wireless LAN, the wireless AP acts autonomously, rather than with a WLC, doing all the work required to create and control the WLAN. In other words, the autonomous AP communicates with the various wireless devices using 802.11 protocols and radio waves. It uses Ethernet protocols on the wired side. It converts between the differences in header formats between 802.11 and 802.3 frames before forwarding to/from 802.3 Ethernet and 802.11 wireless frames. But it does not encapsulate frames in CAPWAP, because the AP will not send the frames to a WLC.

For the Internet connection, the router (combo) device connects with any available Internet access technology, including cable Internet, DSL, 4G/5G wireless, or fiber Ethernet. Note that Chapter 14, "WAN Architecture," introduces those technologies.

Power over Ethernet (PoE)

Just walk around any building and you see electrical power outlets everywhere. When finishing the interior of a building, electricians run electrical cables and install electrical outlets to any and every location that might need power. They also run power cables so that devices such as light fixtures can be wired to power as well. And when network engineers thought about electrical power, they thought in terms of making sure the electricians had run enough power to the wiring closets and other locations to power the networking devices.

Power over Ethernet (PoE) changes that thinking so that the responsibility to provide electrical power to some devices can fall to the network engineering team. Some classes of device types have been built to be able to receive their power over the Ethernet cable, rather than using a separate power cord. To make that work, the LAN switch connected to the cable must supply that power over the cable. By using PoE, companies can gain several advantages, including reduced cost by requiring fewer cable runs and better power management capabilities as compared with using a traditional electrical power cable run and power outlet. This final section of the chapter examines PoE.

PoE Basics

The family of standards that supply power goes by the general name *Power over Ethernet* (PoE). With PoE, some device, typically a LAN switch, acts as the Power Sourcing Equipment (PSE)—that is, the device that supplies DC power over the Ethernet UTP cable (as shown in Figure 13-9). A device that has the capability to be powered over the Ethernet cable, rather than by some other power connector on the device, is called the Powered Device (PD).

Key Topic

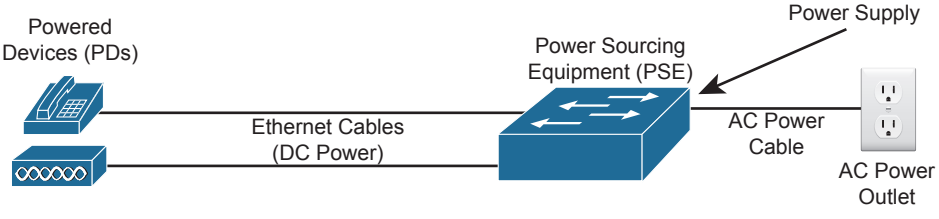


Figure 13-9 *Power over Ethernet Terminology*

PoE has a great advantage for devices installed to locations that often do not have a pre-installed power cable or power output. For instance, wireless design places APs in a wide range across the ceiling of a floor (or story) in a building. Also, IP video cameras might be placed in the ceiling corners inside or at various outside locations. Instead of running new power and new network cables to support each device, a single Ethernet cable run can supply power to the device while allowing normal Ethernet communications over the same cable and same wire pairs.

PoE also helps in some less obvious practical ways because it supplies DC power over the Ethernet cable, so the device does not need an AC/DC converter. For instance, devices like laptops and IP phones use a power cord that includes a power brick—an AC-to-DC converter—which converts the AC power from the power outlet to the DC power needed by the device. PoE supplies DC current over the Ethernet cable. So, for an IP Phone, for instance, no more power cable and no more power brick cluttering the desk or taking up a power outlet.

PoE Operation

PoE must have a means to avoid harming the devices on the end of the circuit. Every electrical device can be harmed by receiving too much current into the device, which is why electricians install circuit breakers and why we use surge protectors. Applying power over an Ethernet cable could have the same effect, harming the device on the other end, if the device does not support PoE. So PoE must (and does) have processes in place to determine if PoE is needed, and for how much power, before applying any potentially harmful power levels to the circuit.

PoE, standardized by the IEEE, extends the same IEEE autonegotiation mechanisms. In fact, the mechanisms need to work before the PD has booted, because the PD needs power before it can boot and initialize. By using these IEEE autonegotiation messages and watching for the return signal levels, PoE can determine whether the device on the end of the cable requires power (that is, it is a PD) and how much power to supply. This list details the major steps:

Step 1. Do not supply power on a PoE-capable port unless negotiation identifies that the device needs power.

Step 2. Use Ethernet autonegotiation techniques, sending low power signals and monitoring the return signal, to determine the PoE power class, which determines how much power to supply to the device.

Step 3. If the device is identified as a PD, supply the power per the power class, which allows the device to boot.

Step 4. Monitor for changes to the power class, both with autonegotiation and listening for CDP and LLDP messages from the PD.

Step 5. If a new power class is identified, adjust the power level per that class.

The negotiation processes result in the PDs signaling how many watts of power they would like to receive from the PSE. Depending on the specific PoE standard, the PSE will then supply the power, either over two pairs or four pairs, as noted in Table 13-2.

Table 13-2 Power over Ethernet Standards

Name	Standard	Watts at PSE	Powered Wire Pairs
Cisco Inline Power	Cisco	7	2
PoE	802.3af	15	2
PoE+	802.3at	30	2
UPoE	802.3bt	60	4
UpoE+	802.3bt	100	4

Cisco has been developing products to use some form of PoE since around 2000. Cisco has often developed prestandard power capabilities, like its original Cisco Inline Power (ILP) feature. Over time, the IEEE has produced standards similar to Cisco's power features, with Cisco supporting the standard version once completed. However, for the most part, the Cisco literature refers to the more common names in the first column of the table.

PoE and LAN Design

Most of the LAN switch features discussed in this book (and in *CCNA 200-301 Official Cert Guide, Volume 1*) exist as software features. Once you learn about a software feature, in some cases all you have to do is configure the feature and start using it. (In some cases, you might need to research and license the feature first.) Regardless, adding the feature takes little or no prior planning.

PoE does require some planning and engineering effort when designing a LAN, both when planning for the cable plant (both Ethernet and electrical), as well as when planning for new networking hardware. Planning with PoE in mind prepares the network to supply power to network devices, rather than reacting and missing opportunities to save money and time.

The following list includes some of the key points to consider when planning a LAN design that includes PoE:

- **Powered Devices:** Determine the types of devices and specific models, along with their power requirements.
- **Power Requirements:** Plan the numbers of different types of PDs to connect into each wiring closet to build a power budget. That power budget can then be processed to determine the amount of PoE power to make available through each switch.
- **Switch Ports:** Some switches support PoE standards on all ports, some on no ports, some on a subset of ports. Research the various switch models so that you purchase enough PoE-capable ports for the switches planned for each wiring closet.
- **Switch Power Supplies:** Without PoE, when purchasing a switch, you choose a power supply so that it delivers enough power to power the switch itself. With PoE, the switch acts as a distributor of electrical power, so the switch power supply must deliver many more watts than it needs to run the switch itself. You will need to create a power budget per switch, based on the number of connected PDs, and purchase power supplies to match those requirements.
- **PoE Standards versus Actual:** Consider the number of PoE switch ports needed, the standards they support, the standards supported by the PDs, and how much power they consume. For instance, a PD and a switch port may both support PoE+, which supports up to 30 watts supplied by the PSE. However, that powered device may need at most 9 watts to operate, so your power budget needs to reserve less power than the maximum for those devices.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 13-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 13-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, app
Review key terms		Book, app
Answer DIKTA questions		Book, PTP
Review memory tables		Book, app

Review All the Key Topics

Key Topic

Table 13-4 Key Topics for Chapter 13

Key Topic Element	Description	Page Number
Figure 13-1	Campus LAN design terms	291
Figure 13-2	Star topology	292
Figure 13-4	A two-tier (collapsed core) LAN topology	293
Figure 13-5	A three-tier (core) LAN topology	294
List	Definitions for LAN core, distribution, and access layers	295
List	Components in an integrated SOHO network device	296
Figure 13-9	PoE roles and terms	297
List	Typical steps to discover power requirements with PoE	298

Key Terms You Should Know

star topology, full mesh, partial mesh, collapsed core design, core design, access layer, distribution layer, core layer, SOHO, powered device (PD), power sourcing equipment (PSE), PoE, UPoE

This page intentionally left blank

CHAPTER 14

WAN Architecture

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.2 Describe the characteristics of network topology architecture

1.2.d WAN

5.0 Security Fundamentals

5.5 Describe remote access and site-to-site VPNs

The CCNA 200-301 exam topics include only brief mentions of WAN topics. Because of that sparse attention to WANs, the *CCNA 200-301 Official Cert Guide, Volume 1*, introduced just enough detail about two types of WAN links—point-to-point serial and point-to-point Ethernet WAN links—so that you could understand IP routing, which is a major focus in CCNA.

This chapter now turns our attention to WAN topics for a deeper look at three branches of WAN technology. As usual for this book's discussion of WAN services, the service is viewed mostly from the perspective of the enterprise, as the customer of some WAN service provider (SP). That means the discussion focuses on what the enterprise receives from the service, rather than how the service provider implements the service inside its network. (Note that Cisco's Service Provider certification track explores the details of how an SP implements its network.)

This chapter begins with a discussion of Metro Ethernet, a technology that defines how to use Ethernet links between a customer site and the SP. The second section then examines MPLS VPNs, even though MPLS VPNs came before Metro Ethernet historically. The chapter introduces Metro Ethernet first because the many similarities between using Ethernet in the LAN and using Ethernet in the WAN make this topic easier to learn.

The chapter closes with a third section about how to use the Internet as a private WAN service by using virtual private network (VPN) technology. The Internet does not inherently provide a private service in that any attacker who gets a copy of your packets as they pass through the Internet can read the contents. VPN servers secure the data sent over the Internet, effectively creating a private WAN service over the public Internet.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 14-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Metro Ethernet	1–3
Multiprotocol Label Switching (MPLS)	4–6
Internet VPNs	7

1. Which of the following topology terms most closely describe the topology created by a Metro Ethernet Tree (E-Tree) service? (Choose two answers.)
   a. Full mesh
   b. Partial mesh
   c. Hub and spoke
   d. Point-to-point
2. Which of the following is the most likely technology used for an access link to a Metro Ethernet service?
   a. 100BASE-LX10
   b. High-speed TDM (for example, T3, E3)
   c. MPLS
   d. 100BASE-T
3. An enterprise uses a Metro Ethernet WAN with an Ethernet LAN (E-LAN) service, with the company headquarters plus 10 remote sites connected to the service. The enterprise uses OSPF at all sites, with one router connected to the service from each site. Which of the following are true about the Layer 3 details most likely used with this service and design? (Choose two answers.)
   a. The WAN uses one IP subnet.
   b. The WAN uses 10 or more IP subnets.
   c. A remote site router would have one OSPF neighbor.
   d. A remote site router would have 10 OSPF neighbors.
4. An enterprise uses an MPLS Layer 3 VPN with the company headquarters connected plus 10 remote sites connected to the service. The enterprise uses OSPF at all sites, with one router connected to the service from each site. Which of the following are true about the Layer 3 details most likely used with this service and design? (Choose two answers.)
   a. The WAN uses one IP subnet.
   b. The WAN uses 10 or more IP subnets.
   c. A remote site router would have one OSPF neighbor.
   d. A remote site router would have 10 or more OSPF neighbors.

5. Which of the following answers is most accurate about access link options for an MPLS network?
   a. Uses only TDM (T1, T3, E1, E3, etc.)
   b. Uses only Ethernet
   c. Uses only DSL and cable
   d. Uses a wide variety of Layer 1 and Layer 2 networking technologies
6. An enterprise connects 20 sites into an MPLS VPN WAN. The enterprise uses OSPF for IPv4 routes at all sites. Consider the OSPF area design options and the PE-CE links. Which of the following answers is most accurate about OSPF areas and the PE-CE links?
   a. The PE-CE link may or may not be chosen to be in backbone area 0.
   b. The PE-CE link must not be in the backbone area 0.
   c. The PE-CE link must be in the backbone area 0.
   d. The PE-CE link will not be in any OSPF area.
7. A colleague mentions using a remote access VPN. Which of the following protocols or technologies would you expect your colleague to have used?
   a. TLS
   b. IPsec
   c. GRE
   d. FTPS

Foundation Topics

Metro Ethernet

Metro Ethernet (MetroE) includes a variety of WAN services with some common features. Each MetroE service uses Ethernet physical links to connect the customer's device to the service provider's device. Second, the service is a Layer 2 service in that the WAN provider forwards Ethernet frames from one customer device to another.

To begin the conversation with a basic view, Metro Ethernet acts much as if the WAN service were created by one Ethernet switch, as shown in Figure 14-1. The figure shows four sites in the same company, each with a router. Each router is connected to the WAN service with an Ethernet link of some kind; those Ethernet links typically use one of the fiber Ethernet standards due to the distances involved. From the customer's perspective (that is, from the perspective of the enterprise that is the customer of the WAN SP), the WAN service acts like a LAN switch in that it forwards Ethernet frames.

NOTE Throughout this chapter, the word *customer* refers to the customer of the service provider—that is, the enterprise that is purchasing the WAN service.

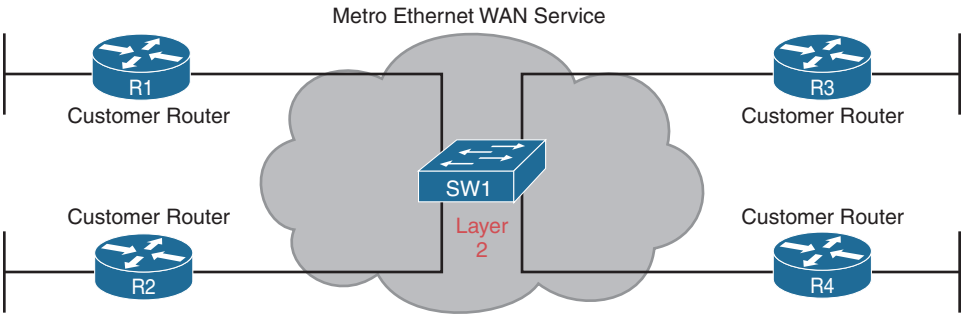


Figure 14-1 *Metro Ethernet Concept as a Large Ethernet Switch*

Although the main concept makes a Metro Ethernet service act like a big LAN switch, there are many options, and you should understand the basics of each. Additionally, many customers connect to a Metro Ethernet service with either routers or Layer 3 switches, which brings up some Layer 3 issues with IP addressing and routing protocols. This section closes with a discussion of the Layer 3 issues.

Metro Ethernet Physical Design and Topology

From an enterprise perspective, to use a Metro Ethernet service, each site needs to connect to the service with (at least) one Ethernet link. There is no need to connect each enterprise router to each other enterprise router directly with a physical link. For instance, in Figure 14-1 in the previous section, each of the four enterprise routers connects to the SP's MetroE service with one physical Ethernet link, rather than connecting directly to the other enterprise routers.

From the SP perspective, the SP needs to build a network to create the Metro Ethernet service. To keep costs lower the SP puts a device (typically an Ethernet switch) physically near to as many customer sites as possible, in an SP facility called a *point of presence* (PoP). Those SP switches need to be near enough to many customer locations so that some Ethernet standard supports the distance from the SP's PoP to each customer site. Figure 14-2 collects some of these terms and ideas together.

Key Topic

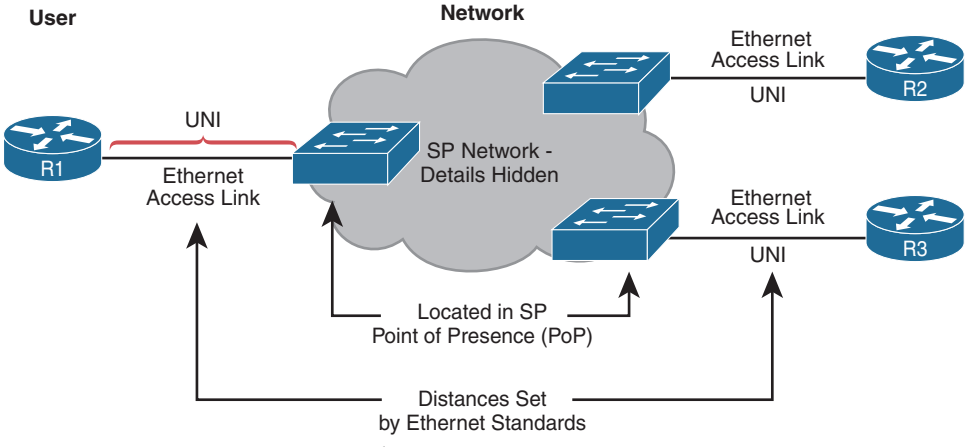


Figure 14-2 *Ethernet Access Links into a Metro Ethernet Service*

Working through the details in the figure, the physical link between the customer and the SP is called an *access link* or, when using Ethernet specifically, an *Ethernet access link*. Everything that happens on that link falls within the definition of the *user network interface*, or UNI. Breaking down the term UNI, the word *network* refers to the SP's network, while the SP's customer (the enterprise) is known as the *user* of the network.

Focusing on the center of Figure 14-2, the SP's network remains hidden to a great extent. The SP promises to deliver Ethernet frames across the WAN. To do that, the access links connect to an Ethernet switch. As you can imagine, the switch will look at the Ethernet header's MAC address fields and at 802.1Q trunking headers for VLAN tags, but the details inside the network remain hidden.

The UNI references a variety of standards, including the fact that any IEEE Ethernet standard can be used for the access link. Table 14-2 lists some of the standards you might expect to see used as Ethernet access links, given their support of longer distances than the standards that use UTP cabling.

Table 14-2 IEEE Ethernet Standards Useful for Metro Ethernet Access

Name	Speed	Distance
100BASE-LX10	100 Mbps	10 Km
1000BASE-LX	1 Gbps	5 Km
1000BASE-LX10	1 Gbps	10 Km
1000BASE-ZX	1 Gbps	100 Km
10GBASE-LR	10 Gbps	10 Km
10GBASE-ER	10 Gbps	40 Km

Ethernet WAN Services and Topologies

Beyond adding a physical Ethernet connection from each site into the SP's Metro Ethernet WAN service, the enterprise must choose between several possible variations of MetroE services. Those variations use different topologies that meet different customer needs.

MEF (www.mef.net) defines the standards for Metro Ethernet, including the specifications for different kinds of MetroE services. Table 14-3 lists three service types described in this chapter and their topologies. The next few pages after the table go into more depth about each.

Key Topic

Table 14-3 Three MEF Service Types and Their Topologies

MEF Service Name	MEF Short Name	Topology Terms	Description
Ethernet Line Service	E-Line	Point-to-point	Two customer premise equipment (CPE) devices can exchange Ethernet frames, similar in concept to a leased line.
Ethernet LAN Service	E-LAN	Full mesh	This service acts like a LAN, in that all devices can send frames to all other devices.
Ethernet Tree Service	E-Tree	Hub and spoke; partial mesh; point-to-multipoint	A central site can communicate to a defined set of remote sites, but the remote sites cannot communicate directly.

Answers to the "Do I Know This Already?" quiz:

1 B, C **2** A **3** A, D **4** B, C **5** D **6** A **7** A

NOTE You might see the term *Virtual Private Wire Service* (VPWS) used for what MEF defines as E-Line service, and *Virtual Private LAN Service* (VPLS) used for what MEF defines as E-LAN service. You might also see the term *Ethernet over MPLS* (EoMPLS). All these terms refer to cases in which the SP uses MPLS internally to create what the customer sees as an Ethernet WAN service.

Ethernet Line Service (Point-to-Point)

The Ethernet Line Service, or E-Line, is the simplest of the Metro Ethernet services. The customer connects two sites with access links. Then the MetroE service allows the two customer devices to send Ethernet frames to each other. Figure 14-3 shows an example, with routers as the CPE devices.

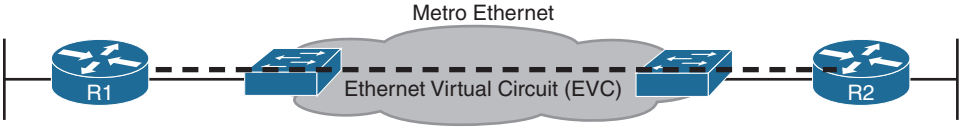


Figure 14-3 *Point-to-Point Topology in Metro Ethernet E-Line Service Between Routers*

As with all MetroE services, the promise made by the service is to deliver Ethernet frames across the service, as if the two customer routers had a rather long crossover cable connected between them. In fact, the E-Line service is the same Ethernet WAN service you have already seen in many examples throughout this book and *CCNA 200-301 Official Cert Guide, Volume 1*. For instance, in this case:

- The routers would use physical Ethernet interfaces.
- The routers would configure IP addresses in the same subnet as each other.
- Their routing protocols would become neighbors and exchange routes.

The MetroE specifications define the concept of an *Ethernet Virtual Connection*, or EVC, to define which user (customer) devices can communicate with which. By definition, an E-Line service (as shown in Figure 14-4) creates a point-to-point EVC, meaning that the service allows two endpoints to communicate.

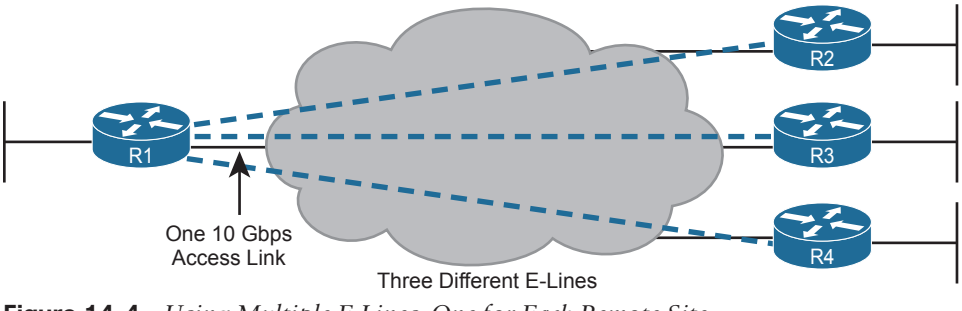


Figure 14-4 *Using Multiple E-Lines, One for Each Remote Site*

It may be that an enterprise wants to implement a network exactly as shown in Figure 14-3, with two sites and two routers, with MetroE WAN connectivity using an E-Line service. Other variations exist, even other variations using an E-Line.

For example, think of a common enterprise WAN topology with a central site and 100 remote sites. As shown so far, with an E-Line service, the central site router would need 100 physical Ethernet interfaces to connect to those 100 remote sites. That could be expensive. As an alternative, the enterprise could use the design partially shown in Figure 14-4 (just three remote sites shown). In this case:

- The central site router uses a single 10-Gbps access link.
- The central site connects to 100 E-Lines (only three shown).
- All the E-Lines send and receive frames over the same access link.

Note that this chapter does not get into the configuration details for WAN services. However, designs like Figure 14-4, with multiple E-Line services on a single access link, use 802.1Q trunking, with a different VLAN ID for each E-Line service. As a result, the router configuration can use a typical router configuration with trunking and subinterfaces.

Before moving on to the next MetroE service, note that the customer could use switches instead of routers to connect to the WAN. Historically, enterprise engineers place routers at the edge of a WAN, in part because that device connected to both the WAN and the LAN, and the LAN and WAN used different types of physical interfaces and different data-link protocols. As a result of how routing works, routers served as the perfect device to sit at the edge between LAN and WAN (called the WAN edge). With MetroE, the LAN and WAN are both Ethernet, so an Ethernet switch becomes an option.

Ethernet LAN Service (Full Mesh)

Imagine an enterprise needs to connect several sites to a WAN, and the goal is to allow every site to send frames directly to every other site. You could do that with E-Lines, but you would need possibly lots of E-Lines. For instance, to connect three sites with E-Lines so that each site could send frames directly to each other, you only need three E-Lines. But with four, five, and six sites, you would need 6, 10, and 15 E-Lines, respectively. Get up to 20 sites for which all could send frames directly to each other, and you would need 190 E-Lines. (The formula is $N(N - 1) / 2$.)

The people who created MetroE anticipated the need for designs that allow a full mesh—that is, for each pair of nodes in the service to send frames to each other directly. In fact, allowing all devices to send directly to every other device sounds a lot like an Ethernet LAN, so the MetroE service is called an *Ethernet LAN service*, or E-LAN.

One E-LAN service allows all devices connected to that service to send Ethernet frames directly to every other device, just as if the Ethernet WAN service were one big Ethernet switch. Figure 14-5 shows a representation of a single E-LAN EVC. In this case, the one EVC connects to four customer sites, creating one E-LAN. Routers R1, R2, R3, and R4 can all send frames directly to each other. They would also all be in the same Layer 3 subnet on the WAN.

An E-LAN service connects the sites in a full mesh. The term *full mesh* refers to a design that, for a set of devices, creates a direct communication path for each pair. In contrast, a partial mesh refers to a design in which only some of the pairs can communicate directly. The Ethernet Tree service (E-Tree), as discussed in the next topic, creates a partial mesh design.

Key Topic

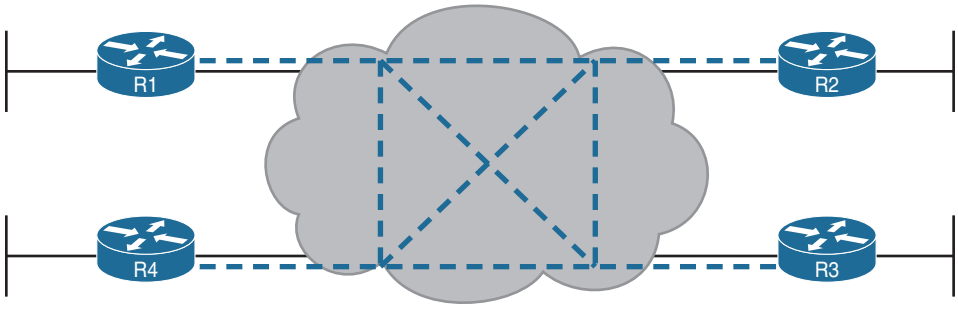


Figure 14-5 *MetroE Ethernet LAN Service—Any-to-Any Forwarding over the Service*

Ethernet Tree Service (Hub and Spoke)

The Ethernet Tree service (E-Tree) creates a WAN topology in which the central site device can send Ethernet frames directly to each remote (leaf) site, but the remote (leaf) sites can send only to the central site. Figure 14-6 shows the topology, again with a single EVC. In this case, router R1 is the root site, and can send to all three remote sites. Routers R2, R3, and R4 can send only to R1.

Key Topic

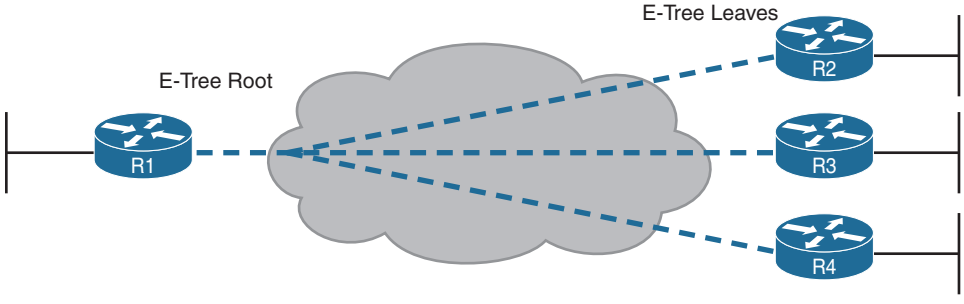


Figure 14-6 *E-Tree Service Creates a Hub-and-Spoke Topology*

With an E-Tree, the central site serves as the root of a tree and each remote site as one of the leaves. The topology goes by many names: partial mesh, hub and spoke, and point-to-multipoint. Regardless of the term you use, an E-Tree service creates a service that works well for designs with a central site plus many remote sites.

Layer 3 Design Using Metro Ethernet

Now that you know the basics of the E-Line (point-to-point), E-LAN (full mesh), and E-Tree (point-to-multipoint, hub and spoke) services, this next topic reviews some Layer 3 design details when using E-Line and E-Tree services. That is, if the enterprise uses routers or Layer 3 switches as its WAN edge devices, how should the engineer plan for IP addresses and subnets? What is the impact on routing protocols? This section answers those questions.

Note that this section uses routers as the enterprise's devices, but the concepts apply to Layer 3 switches as well.

Layer 3 Design with E-Line Service

Every E-Line uses a point-to-point topology. As a result, the two routers on the ends of an E-Line need to be in the same subnet. Similarly, when an enterprise uses multiple E-Lines, each should be in a different subnet. As an example, consider Figure 14-7, which shows two E-Lines, both of which connect to router R1 on the left.

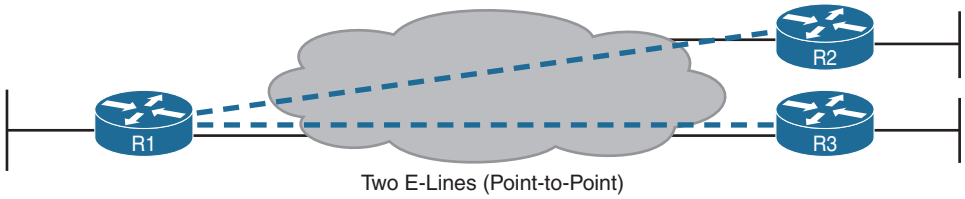
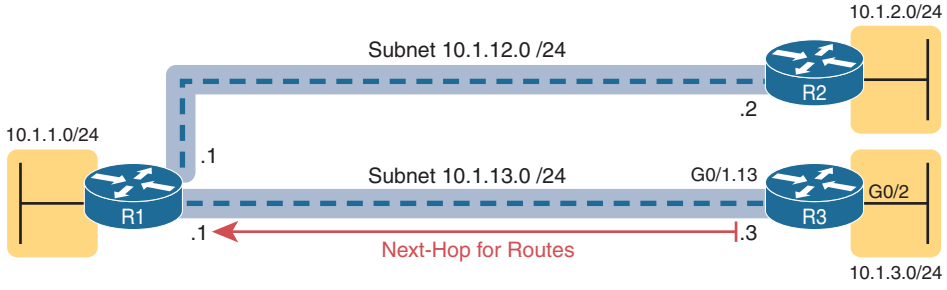


Figure 14-7 *Routing Protocol Neighbor Relationships over Metro Ethernet E-Line*

Focusing on the E-Lines and ignoring the access links for the most part, think of each E-Line as a subnet. Each router needs an IP address in each subnet, and the subnets need to be unique. All the addresses come from the enterprise's IP address space. Figure 14-8 shows an example of the addresses, subnets, and three OSPF-learned routes in the routing table of R3.



R3 Routing Table

Code	Subnet	Interface	Next-hop
O	10.1.1.0/24	G0/1.13	10.1.13.1
O	10.1.2.0/24	G0/1.13	10.1.13.1
O	10.1.12.0/24	G0/1.13	10.1.13.1
C	10.1.3.0/24	G0/2	N/A
C	10.1.13.0/24	G0/1.13	N/A

Figure 14-8 *Layer 3 Forwarding Between Remote Sites—Through Central Site*

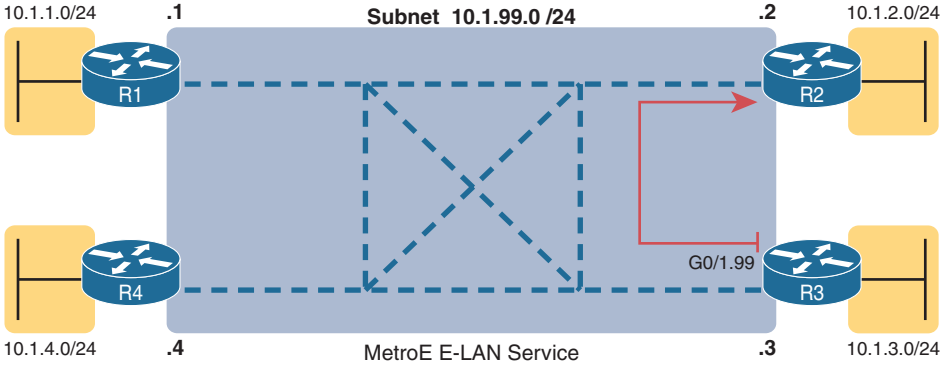
Examine the IP routing table in the lower right of the figure, first focusing on the route to subnet 10.1.1.0/24, which is the LAN subnet off router R1. R3's route points to a next-hop router IP address that is R1's IP address on the Ethernet WAN, specifically the address on the other side of the E-Line that connects R1 and R3. This route should not be a surprise: for R3 to send packets to a subnet connected to R1, R3 sends the packets to R1. Also, it happens to use a subinterface (G0/1.13), which means that the design is using 802.1Q trunking on the link.

Next, look at R3's route for subnet 10.1.2.0/24, which supports the fact that R3 cannot send packets directly to R2 with the current WAN design. R3 does not have an E-Line that allows R3 to send frames directly to R2. R3 will not become routing protocol neighbors with R2 either. So, R3 will learn its route for subnet 10.1.2.0/24 from R1, with R1's 10.1.13.1 address as the next-hop address. As a result, when forwarding packets, R3 will forward packets to R1, which will then forward them over the other E-Line to R2.

Layer 3 Design with E-LAN Service

If you connected four routers to one LAN switch, all in the same VLAN, what would you expect for the IP addresses on those routers? And if all four routers used the same routing protocol, which would become neighbors? Typically, with four routers connected to the same switch, on the same VLAN, using the same routing protocol, normally all four routers would have IP addresses in the same subnet, and all would become neighbors.

On an E-LAN service, the same IP addressing design is used, with the same kinds of routing protocol neighbor relationships. Figure 14-9 shows an example that includes subnets and addresses, plus one route as an example. Note that the four routers connected to the E-LAN service in the center all have addresses in subnet 10.1.99.0/24.



R3 Routing Table (OSPF Routes Only)

Subnet	Interface	Next-hop
10.1.2.0/24	G0/1.99	**10.1.99.2**
10.1.1.0/24	G0/1.99	10.1.99.1
10.1.4.0/24	G0/1.99	10.1.99.4

Figure 14-9 *Layer 3 Forwarding Between Sites with E-LAN Service*

Look at R3's routing table in the figure, the route from R3 to R2's LAN subnet (10.1.2.0/24). In this case, R3's next-hop address is the WAN address on R2 (10.1.99.2), and R3 will send packets (encapsulated in Ethernet frames) directly to R2. Note also that the other two routes in the routing table list the next-hop addresses of R1 (10.1.99.1) and R4 (10.1.99.4).

The details in this first section of the chapter should provide plenty of perspective on how enterprise routers use Ethernet WANs for connectivity. However, if you want a little more detail, the section titled "Ethernet Virtual Circuit Bandwidth Profiles" in Appendix D, "Topics from Previous Editions," discusses the logic behind how Ethernet WANs use physical links at one speed while supporting services that run at a variety of slower speeds.

Multiprotocol Label Switching (MPLS)

From your CCNA preparation so far, you already understand a lot about the Layer 3 routing, as represented by the packet flowing left to right in Figure 14-10. Each router makes a separate forwarding decision to forward the packet, as shown as steps 1, 2, and 3 in the figure. Each router makes a comparison between the packet's destination IP address and that router's IP routing table; the matching IP routing table entry tells the router where to forward the packet next. To learn those routes, the routers typically run some routing protocol.

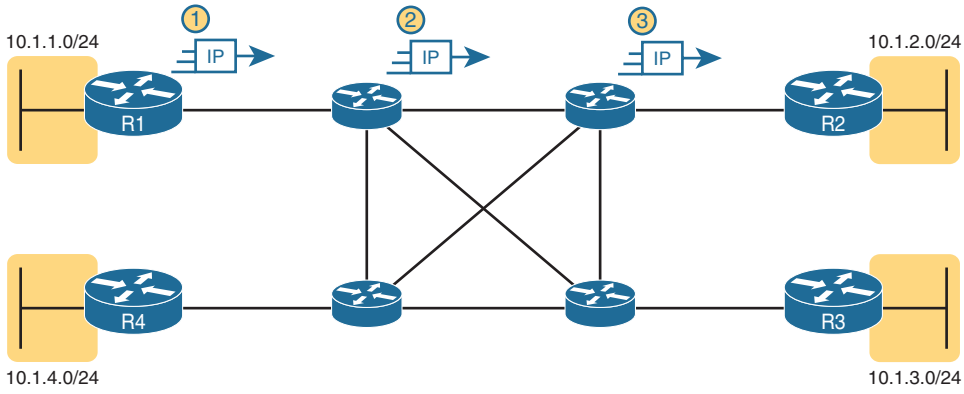


Figure 14-10 *Basic IP Routing of IP Packets*

MPLS creates a WAN service that routes IP packets between customer sites. The enterprise deploys routers and switches as usual. The SP then creates its own IP network, spanning a large geographic area. The customer can then connect to the MPLS network, with a link from each site, with the SP routing IP packets from one customer site to the other. For instance, in Figure 14-10, the middle four routers could represent the SP's MPLS network, with the numbered routers on the edges being routers owned by one company.

However, an SP cannot just build a large IP network and connect all its customers to that same IP network because of some issues that arise to support multiple customers at the same time. For instance, many customers will use the same private IP network (for instance, network 10.0.0.0), so the SP's IP network would learn large numbers of routes to overlapping subnets.

To overcome this and other issues, the SP builds its IP network to also use Multiprotocol Label Switching (MPLS), in particular MPLS VPNs. MPLS VPNs allow the SP to build one large MPLS network, which also creates a private IP-based WAN for each of its customers. With MPLS VPNs, the SP can separate the routes learned from one customer from the routes learned for the next customer; consequently, the SP can support each customer while preventing packets from leaking from one customer to the next.

To give you a little insight as to why MPLS is not just an IP network with routers, internally, the devices in an MPLS network use label switching—hence, the name MPLS. The routers on the edge of the MPLS network add and remove an MPLS header to packets as they enter and exit the MPLS network. The devices inside the MPLS network then use the label field inside that MPLS header when forwarding data across the MPLS network. The choices of the labels to use, along with other related logic, allow the MPLS VPN to create separate VPNs to keep different customers' traffic separate.

NOTE While MPLS VPNs provide a Layer 3 service to customers, MPLS itself is sometimes called a Layer 2.5 protocol because it adds the MPLS header between the data-link header (Layer 2) and the IP header (Layer 3).

As usual, the discussion of WAN services in this book ignores as much of the SP's network as possible. For instance, you do not need to know how MPLS labels work. However, because MPLS VPNs create a Layer 3 service, the customer must be more aware of what

the SP does than with other WAN servers, so you need to know a few facts about how an MPLS network approaches some Layer 3 functions. In particular, the SP's MPLS VPN network

Key Topic

- Will use a routing protocol to build routing protocol neighbor relationships with customer routers
- Will learn customer subnets/routes with those routing protocols
- Will advertise a customer's routes with a routing protocol so that all routers that customer connects to the MPLS VPN can learn all routes as advertised through the MPLS VPN network
- Will make decisions about MPLS VPN forwarding, including what MPLS labels to add and remove, based on the customer's IP address space and customer IP routes

As an aside, MPLS VPNs create a private network by keeping customer data separate, but not by encrypting the data. Some VPN services encrypt the data, expecting that attackers might be able to receive copies of the packets. With MPLS, even though the packets for two customers may pass through the same devices and links inside the MPLS network, MPLS logic can keep the packets separate for each customer.

This second of two major sections of the chapter works through the basics of MPLS, specifically MPLS VPNs. This section first looks at the design, topology, and terminology related to building the customer-facing parts of an MPLS network. It then looks at the impact and issues created by the fact that the MPLS network provides a Layer 3 service.

MPLS VPN Physical Design and Topology

MetroE provides a Layer 2 service by forwarding Layer 2 Ethernet frames. To do that, the SP often uses Ethernet switches at the edge of its network. Those switches are configured to do more than what you learn about Ethernet LAN switches for CCNA, but a LAN switch's most fundamental job is to forward an Ethernet frame, so it makes sense for MetroE to use an Ethernet switch at the edge of the SP's MetroE network.

MPLS provides a Layer 3 service in that it promises to forward Layer 3 packets (IPv4 and IPv6). To support that service, MPLS SPs typically use routers at the edge of the MPLS networks because routers provide the function of forwarding Layer 3 packets.

As usual, each WAN technology has its own set of terms and acronyms, so Figure 14-11 shows two important MPLS terms in context: customer edge (CE) and provider edge (PE). Because MPLS requires so much discussion about the devices on the edge of the customer and SP network, MPLS uses specific terms for each. The *customer edge* device is typically a router, and it sits at a customer site—that is, at a site in the company that is buying the MPLS service. The *provider edge* devices sit at the edge of the SP's network, on the other end of the access link.

Next, to appreciate what MPLS does, think back to how routers use their different kinds of physical interfaces and different kinds of data-link protocols. When routing a packet, routers discard an incoming data-link frame's data-link header and trailer and then build a new data-link header/trailer. That action allows the incoming packet to arrive inside a frame of one data-link protocol and leave out an interface with another data-link protocol.

Key Topic

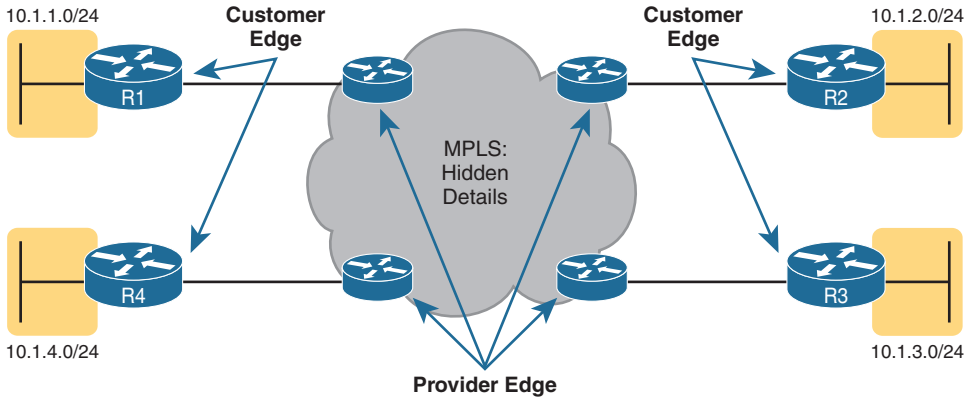


Figure 14-11 *MPLS Layer 3 Design, with PE and CE Routers*

With MPLS, the fact that the devices are routers, discarding and adding new data-link headers, means that MPLS networks support a variety of access links. The fact that MPLS acts as a Layer 3 service, discarding incoming data-link headers, means that any data-link protocol could in theory be used on MPLS access links. In reality, MPLS does support many types of access links, as shown in Figure 14-12.

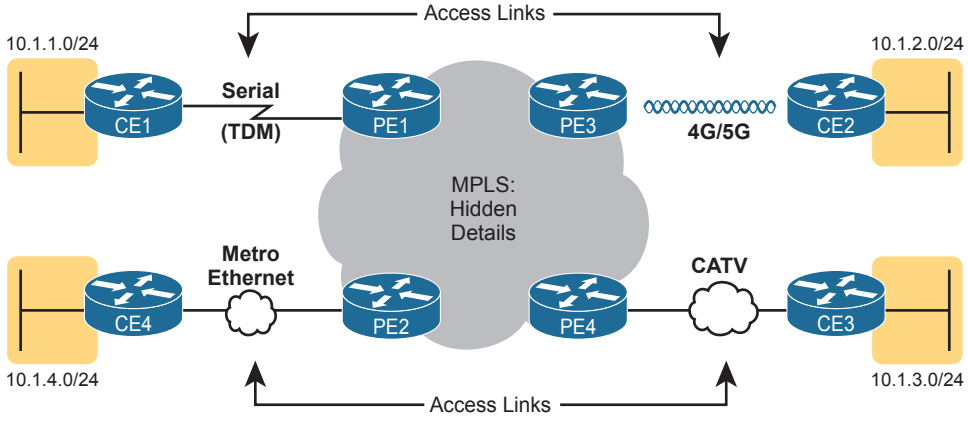


Figure 14-12 *Popular MPLS Access Link Technologies*

The variety of access links available for MPLS networks makes MPLS a great option for building large enterprise networks. For sites that are near MetroE services, especially for sites that need at least 10 Mbps of bandwidth, using MetroE as an access link makes great sense. Then, for sites that are more remote, the carrier may not offer MetroE services to that area, but many carriers can install a serial link to remote sites. Or, common Internet access technologies, like cable and wireless 4G/5G services, can also be used to access an MPLS network.

MPLS and Quality of Service

MPLS stands apart from other WAN services as the first WAN service for which the SP provided effective Quality of Service (QoS) features. You should be able to get a general idea of an MPLS QoS benefit with the following basic example.

IP networks can and often do forward voice traffic in IP packets, called Voice over IP (VoIP). If a WAN service does not provide QoS, that means that the WAN service does not treat one packet any differently than any other packet. With QoS, the SP's network can treat packets differently, giving some packets (like VoIP) better treatment. For a voice call to sound good, each voice packet must have low loss (that is, few packets are discarded); low one-way delay through the network; and low variation in delay (called jitter). Without QoS, a voice call over an IP network will not sound good.

With a QoS-capable WAN, the customer can mark VoIP packets so that the MPLS network can recognize VoIP packets and treat them better, resulting in better voice call quality. But to make it work correctly, the customer and MPLS provider need to cooperate.

For instance, for VoIP packets traveling left to right in Figure 14-13, router CE1 could be configured with QoS marking tools. Marking tools could recognize VoIP packets and place a specific value in the IP header of VoIP packets (a value called DSCP EF, per the figure). The MPLS WAN provider would then configure its QoS tools to react for packets that have that marking, typically sending that packet as soon as possible. The result: low delay, low jitter, low loss, and a better call quality.

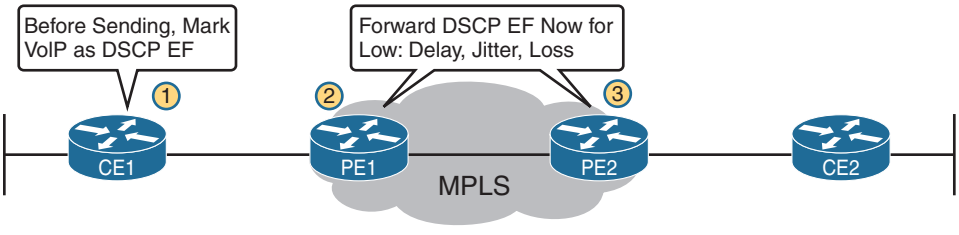


Figure 14-13 *MPLS VPN QoS Marking and Reaction in the MPLS WAN*

Summarizing the ideas so far, MPLS supports a variety of access links. An enterprise would select the type and speed of access link for each site based on the capacity (bandwidth) required for each site. Beyond that basic connectivity, the enterprise will want to work with the SP to define other features of the service. The customer and SP will need to work through the details of some Layer 3 design choices (as discussed in more depth in the next section). The customer will also likely want to ask for QoS services from the MPLS provider and define those details.

Layer 3 with MPLS VPN

Because MetroE provides a Layer 2 service, the SP has no need to understand anything about the customer's Layer 3 design. The SP knows nothing about the customer's IP addressing plan and has no need to participate with routing protocols.

MPLS VPNs take the complete opposite approach. As a Layer 3 service, MPLS must be aware of the customer IP addressing. The SP will even use routing protocols and advertise those customer routes across the WAN. This section takes a closer look at what that means.

First, keep the primary goals in mind. The customer pays good money for a WAN service to deliver data between sites, with certain levels of availability and quality (for instance, low delay, jitter, and loss for VoIP). But to support that base function of allowing packet delivery from each WAN site to the other, the CE routers need to exchange routes with the PE routers in the MPLS network. Additionally, all the CE routers need to learn routes from the other CE routers—a process that relies on the PE routers.

First, the CE routers and the PE router on the ends of the same access link need to exchange routes, as shown in Figure 14-14. The figure shows the CE-PE routing protocol neighbor relationships (as lines with circles on the ends). In this case, the customer chose to use OSPF. However, MPLS allows for many familiar routing protocols on the edge of the MPLS network: RIPv2, EIGRP, OSPF, and even eBGP.

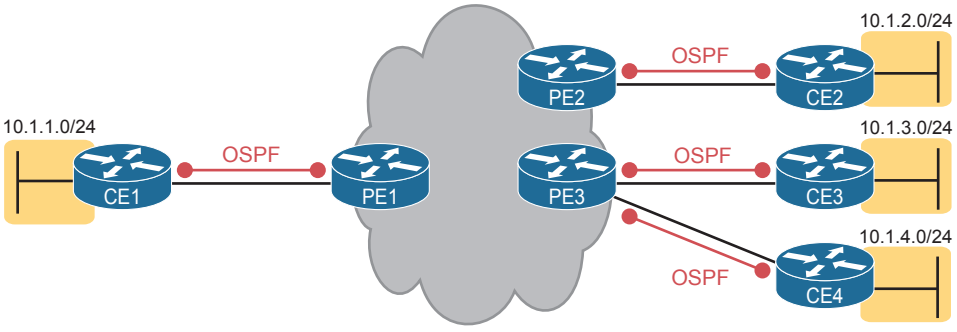


Figure 14-14 *Routing Protocol Neighbor Relationships with MPLS Customer Edge Routers*

Additionally, all the CE routers need to learn routes from the other CE routers. However, a CE router does not form routing protocol neighbor relationships directly with the other CE routers, as noted in Figure 14-14. Summarizing what does and does not happen:

Key Topic

- A CE router does become neighbors with the PE router on the other end of the access link.
- A CE router does not become neighbors with other CE routers.
- The MPLS network does advertise the customer's routes between the various PE routers so that the CE routers can learn all customer routes through their PE-CE routing protocol neighbor relationship.

To advertise the customer routes between the PE routers, the PE routers use another routing protocol along with a process called *route redistribution*. Route redistribution happens inside one router, taking routes from one routing protocol process and injecting them into another. MPLS does route redistribution in the PE routers between the routing protocol used by the customer and a variation of BGP called Multiprotocol BGP (MPBGP). (Redistribution is needed when the PE-CE routing protocol is not BGP.) Figure 14-15 shows the idea.

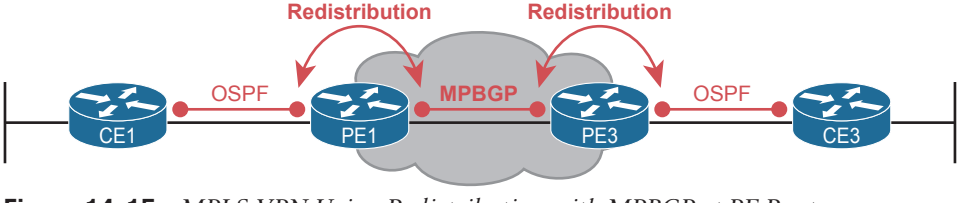


Figure 14-15 *MPLS VPN Using Redistribution with MPBGP at PE Router*

Just as a quick aside about MPBGP, MPLS VPNs use MPBGP (as opposed to other routing protocols) because MPBGP can advertise routes from multiple customers while keeping the

routes logically separated. For instance, continuing the example in Figure 14-15, router PE1 might sit in one PoP but connect to dozens of different customers. Likewise, router PE3 might connect to many of those same customers. MPBGP can advertise routes for all those customers and mark which routes are from which customers so that only the correct routes are advertised to each CE router for different customers.

At the end of the process, for all single enterprises, all the routers can learn routes to all the subnets reachable over the MPLS VPN WAN. WAN routes on the CE routers refer to the neighboring PE router as the *next-hop router*. Each CE router becomes a routing protocol neighbor with the SP's PE router on the other end of the access link. Plus, MPLS provides the flexibility to use whatever type of physical access link makes sense for the location at each site, while still connecting to the same MPLS network.

Internet VPNs

To build the Internet, Internet service providers (ISP) need links to other ISPs as well as links to the ISPs' customers. The Internet core connects ISPs to each other using a variety of high-speed technologies. Additionally, Internet access links connect an ISP to each customer, again with a wide variety of technologies. The combination of ISP networks and customer networks that connect to the ISPs together create the worldwide Internet.

For these customer access links, the technologies need to be inexpensive so that a typical consumer can afford to pay for the service. But businesses can use many of these same technologies to connect to the Internet. Some WAN technologies happen to work particularly well as Internet access technologies. For example, several use the same telephone line installed into most homes by the phone company so that the ISPs do not have to install additional cabling. Some use the TV cabling, whereas others use wireless.

While consumers typically connect to the Internet to reach destinations on the Internet, businesses can also use the Internet as a WAN service. First, the enterprise can connect each business site to the Internet. Then, using virtual private network (VPN) technology, the enterprise can create an Internet VPN. An Internet VPN can keep the enterprise's packet private through encryption and other means, even while sending the data over the Internet.

This final major section of the chapter discusses some of the basics of Internet access links. The section then details how an enterprise can communicate securely over the Internet, making the public Internet act like a private network, by creating an Internet VPN.

Internet Access

Private WAN technology may be used to access an ISP's network, including the Ethernet WAN and MPLS technologies discussed earlier in this chapter. Figure 14-16 shows a few of these, just as a visual reminder of these options.

In addition to the traditional services shown in the figure, enterprises can also use Internet access technologies more commonly used by consumers, including DSL, cable, 4G/5G, and fiber Ethernet. The chapter includes this information about Internet access technologies to provide useful background information before getting into Internet VPN topics.

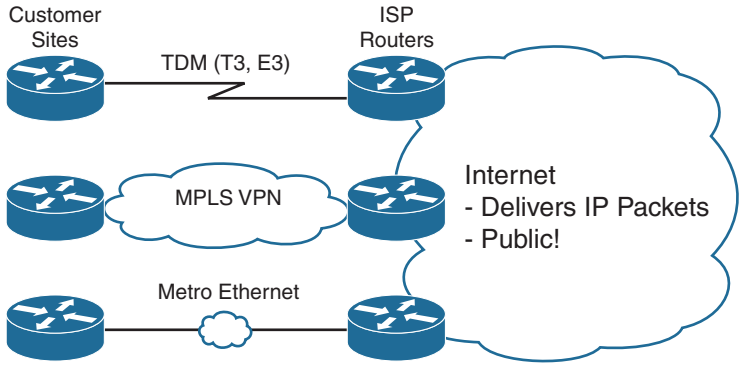


Figure 14-16 *Three Examples of Internet Access Links for Companies*

Digital Subscriber Line

In the consumer Internet access space, one big speed breakthrough happened with the introduction of the digital subscriber line (DSL). It represented a big technological breakthrough in terms of raw speed in comparison to some older technologies, such as analog modems. These faster speeds available through DSL also changed how people could use the Internet because many of today's common applications would be unusable with the earlier Internet access technologies (analog modems and Integrated Services Digital Network, or ISDN).

Telephone companies (telcos) greatly influenced the creation of DSL. As a technology, DSL gave telcos a way to offer much faster Internet access speeds. As a business opportunity, DSL gave telcos a way to offer a valuable high-speed Internet service to many of their existing telephone customers, over the same physical phone line already installed, which created a great way for telcos to make money.

Figure 14-17 shows some of the details of how DSL works on a home phone line. The phone can do what it has always done: plug into a phone jack and send analog signals. For the data, a DSL modem connects to a spare phone outlet. The DSL modem sends and receives the data, as digital signals, at higher frequencies, over the same local loop, even at the same time as a telephone call. (Note that the physical installation often uses frequency filters that are not shown in the figure or discussed here.)

Because DSL sends analog (voice) and digital (data) signals on the same line, the telco has to somehow split those signals on the telco side of the connection. To do so, the local loop must be connected to a *DSL access multiplexer* (DSLAM) located in the nearby telco central office (CO). The DSLAM splits out the digital data over to the router on the lower right in Figure 14-17, which completes the connection to the Internet. The DSLAM also splits out the analog voice signals over to the voice switch on the upper right.

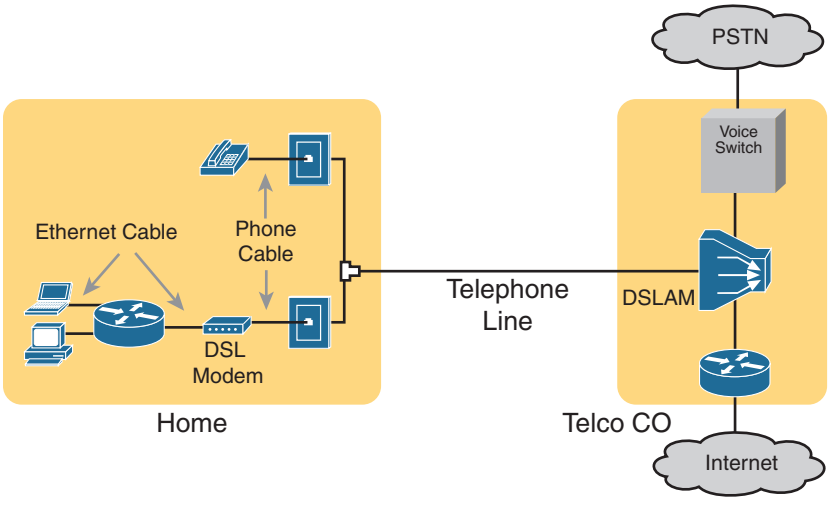


Figure 14-17 *Wiring and Devices for a Home DSL Link*

Cable Internet

DSL uses the local link (telephone line) from the local telco. Cable Internet instead uses the cabling from what has become the primary competitor to the telco in most markets: the cable company.

Cable Internet creates an Internet access service that, when viewed generally rather than specifically, has many similarities to DSL. Like DSL, cable Internet takes full advantage of existing cabling, using the existing cable TV (CATV) cable to send data. Like DSL, cable Internet uses asymmetric speeds, sending data faster downstream than upstream, which works well for most consumer locations. And cable Internet still allows the normal service on the cable (cable TV), at the same time as the Internet access service is working.

Cable Internet also uses the same general idea for in-home cabling as DSL, just using CATV cabling instead of telephone cabling. The left side of Figure 14-18 shows a TV connected to the CATV cabling, just as it would normally connect. At another cable outlet, a cable modem connects to the same cable. The Internet service flows over one frequency, like yet another TV channel, just reserved for Internet service.

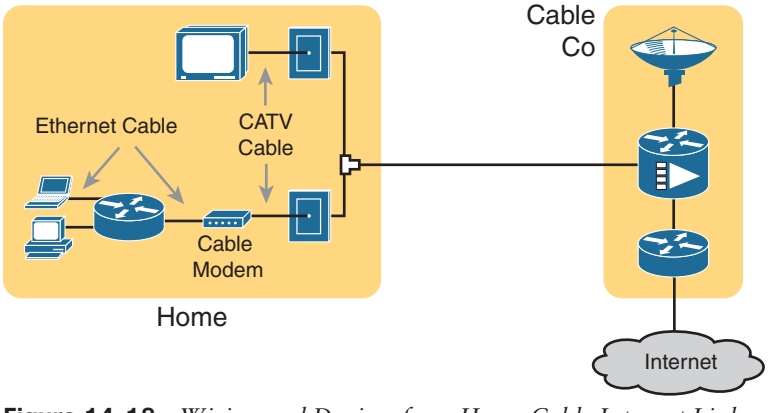


Figure 14-18 *Wiring and Devices for a Home Cable Internet Link*

Similar to DSL, on the CATV company side of the connection (on the right side of the figure), the CATV company must split out the data and video traffic. Data flows to the lower right, through a router, to the Internet. The video comes in from video dishes for distribution out to the TVs in people's homes.

Wireless WAN (3G, 4G, LTE, 5G)

Many of you reading this book have a mobile phone that has Internet access. That is, you can check your email, surf the Web, download apps, and watch videos. Many of us today rely on our mobile phones, and the Internet access built in to those phones, for most of our tweets and the like. This section touches on the big concepts behind the Internet access technology connecting those mobile phones.

Mobile phones use radio waves to communicate through a nearby mobile phone tower. The phone has a small radio antenna, and the provider has a much larger antenna sitting at the top of a tower somewhere within miles of you and your phone. Phones, tablet computers, laptops, and even routers (with the correct interface cards) can communicate through to the Internet using this technology, as represented in Figure 14-19.

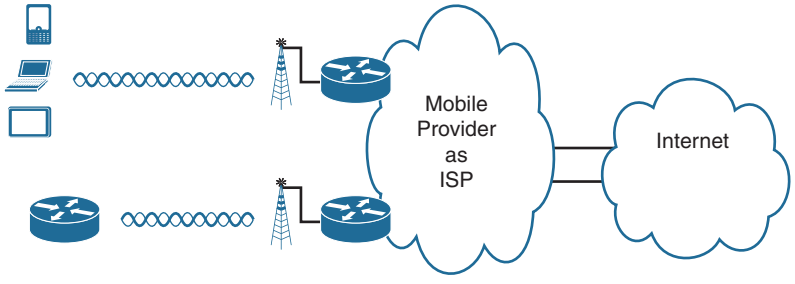


Figure 14-19 *Wireless Internet Access Using 3G/4G/5G Technology*

The mobile phone radio towers also have cabling and equipment, including routers. The mobile provider builds its own IP network, much like an ISP builds out an IP network. The customer IP packets pass through the IP router at the tower into the mobile provider's IP network and then out to the Internet.

The market for mobile phones and wireless Internet access for other devices is both large and competitive. As a result, the mobile providers spend a lot of money advertising their services, with lots of names for one service or the other. Frankly, it can be difficult to tell what all the marketing jargon means, but a few terms tend to be used throughout the industry:

Wireless Internet: This general term refers to Internet services from a mobile phone or from any device that uses the same technology.

3G/4G Wireless: Short for third generation and fourth generation, these terms refer to the major changes over time to the mobile phone companies' wireless networks.

LTE: Long-Term Evolution is a newer and faster technology considered to be part of fourth generation (4G) technology.

5G Wireless: This is the fifth major generation of wireless phone technology.

The takeaway from all this jargon is this: when you hear about wireless Internet services with a mobile phone tower in the picture—whether the device is a phone, tablet, or PC—it

is probably a 3G, 4G, or LTE wireless Internet connection, with newer services offering 5G capabilities by 2020 and beyond.

Enterprises can use this same wireless technology to connect to the Internet. For instance, a network engineer can install a 4G wireless card in a router. ISPs team with wireless operators to create contracts for wireless and Internet service.

Fiber (Ethernet) Internet Access

The consumer-focused Internet access technologies discussed in this section use a couple of different physical media. DSL uses the copper wiring installed between the telco CO and the home. Cable uses the copper CATV cabling installed from the cable company to the home. And, of course, wireless WAN technologies do not use cables for Internet access.

The cabling used by DSL and cable Internet uses copper wires, but, comparing different types of physical media, fiber-optic cabling generally supports faster speeds for longer distances. That is, just comparing physical layer technologies across the breadth of networking, fiber-optic cabling supports longer links, and those links often run at equivalent or faster speeds.

Some ISPs now offer Internet access that goes by the name *fiber Internet*, or simply *fiber*. To make that work, some local company that owns the rights to install cabling underground in a local area (often a telephone company) installs new fiber-optic cabling. Once the cable plant is in place (a process that often takes years as well as a large budget), the fiber ISP then connects customers to the Internet using the fiber-optic cabling. Often, the fiber uses Ethernet protocols over the fiber. The end result: high-speed Internet to the home, often using Ethernet technology.

Internet VPN Fundamentals

Private WANs have some wonderful security features. In particular, the customers who send data through the WAN have good reason to believe that no attackers saw the data in transit or even changed the data to cause some harm. The private WAN service provider promises to send one customer's data to other sites owned by that customer, but not to sites owned by other customers, and vice versa.

VPNs try to provide the same secure features as a private WAN while sending data over a network that is open to other parties (such as the Internet). Compared to a private WAN, the Internet does not provide for a secure environment that protects the privacy of an enterprise's data. Internet VPNs can provide important security features, such as the following:

- **Confidentiality (privacy):** Preventing anyone in the middle of the Internet (man in the middle) from being able to read the data
- **Authentication:** Verifying that the sender of the VPN packet is a legitimate device and not a device used by an attacker
- **Data integrity:** Verifying that the packet was not changed as the packet transited the Internet
- **Anti-replay:** Preventing a man in the middle from copying and later replaying the packets sent by a legitimate user, for the purpose of appearing to be a legitimate user

To accomplish these goals, two devices near the edge of the Internet create a VPN, sometimes called a *VPN tunnel*. These devices add headers to the original packet, with these

headers including fields that allow the VPN devices to make the traffic secure. The VPN devices also encrypt the original IP packet, meaning that the original packet's contents are undecipherable to anyone who happens to see a copy of the packet as it traverses the Internet.

Figure 14-20 shows the general idea of what typically occurs with a VPN tunnel. The figure shows a VPN created between a branch office router and a Cisco firewall. In this case, the VPN is called a *site-to-site VPN* because it connects two sites of a company.

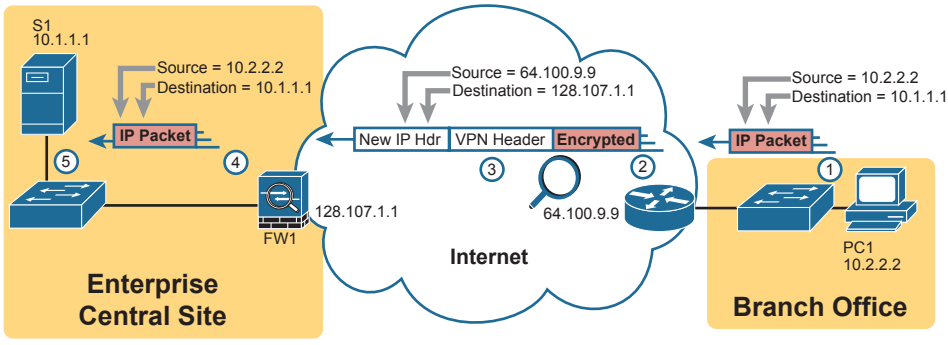


Figure 14-20 *VPN Tunnel Concepts for a Site-to-Site Intranet VPN*

The figure shows the following steps, which explain the overall flow:

1. Host PC1 (10.2.2.2) on the right sends a packet to the web server (10.1.1.1), just as it would without a VPN.
2. The router encrypts the packet, adds some VPN headers, adds another IP header (with public IP addresses), and forwards the packet.
3. An attacker in the Internet copies the packet (called a man-in-the-middle attack). However, the attacker cannot change the packet without being noticed and cannot read the contents of the original packet.
4. Firewall FW1 receives the packet, confirms the authenticity of the sender, confirms that the packet has not been changed, and then decrypts the original packet.
5. Server S1 receives the unencrypted packet.

The benefits of using an Internet-based VPN as shown in Figure 14-20 are many. The cost of a high-speed Internet access connection as discussed in the last few pages is usually much less than that of many private WAN options. The Internet is seemingly everywhere, making this kind of solution available worldwide. And by using VPN technology and protocols, the communications are secure.

NOTE The term *tunnel* refers to any protocol's packet that is sent by encapsulating the packet inside another packet. The term *VPN tunnel* may or may not imply that the tunnel also uses encryption.

Site-to-Site VPNs with IPsec

A site-to-site VPN provides VPN services for the devices at two sites with a single VPN tunnel. For instance, if each site has dozens of devices that need to communicate between

sites, the various devices do not have to act to create the VPN. Instead, the network engineers configure devices such as routers and firewalls (as shown in Figure 14-20) to create one VPN tunnel. The tunnel endpoints create the tunnel and leave it up and operating all the time, so that when any device at either site decides to send data, the VPN is available. All the devices at each site can communicate using the VPN, receiving all the benefits of the VPN, without requiring each device to create a VPN for themselves.

IPsec defines one popular set of rules for creating secure VPNs. IPsec is an architecture or framework for security services for IP networks. The name itself is not an acronym, but rather a name derived from the title of the RFC that defines it (RFC 4301, "Security Architecture for the Internet Protocol"), more generally called IP Security, or IPsec.

IPsec defines how two devices, both of which connect to the Internet, can achieve the main goals of a VPN as listed at the beginning of this section: confidentiality, authentication, data integrity, and anti-replay. IPsec does not define just one way to implement a VPN, instead allowing several different protocol options for each VPN feature. One of IPsec's strengths is that its role as an architecture allows it to be added to and changed over time as improvements to individual security functions are made.

The idea of IPsec encryption might sound intimidating, but if you ignore the math—and thankfully, you can—IPsec encryption is not too difficult to understand. IPsec encryption uses a pair of encryption algorithms, which are essentially math formulas, to meet a couple of requirements. First, the two math formulas are a matched set:

- One to hide (encrypt) the data
- Another to re-create (decrypt) the original data based on the encrypted data

Besides those somewhat obvious functions, the two math formulas were chosen so that if an attacker intercepted the encrypted text but did not have the secret password (called an *encryption key*), decrypting that one packet would be difficult. In addition, the formulas are also chosen so that if an attacker did happen to decrypt one packet, that information would not give the attacker any advantages in decrypting the other packets.

The process for encrypting data for an IPsec VPN works generally as shown in Figure 14-21. Note that the *encryption key* is also known as the *session key*, *shared key*, or *shared session key*.

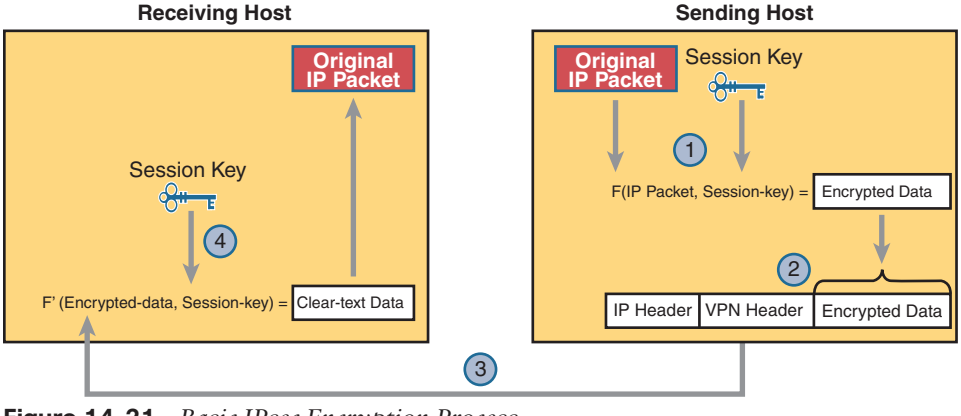


Figure 14-21 *Basic IPsec Encryption Process*

The four steps highlighted in the figure are as follows:

1. The sending VPN device (like the remote office router in Figure 14-21) feeds the original packet and the session key into the encryption formula, calculating the encrypted data.
2. The sending device encapsulates the encrypted data into a packet, which includes the new IP header and VPN header.
3. The sending device sends this new packet to the destination VPN device (FW1 back in Figure 14-21).
4. The receiving VPN device runs the corresponding decryption formula, using the encrypted data and session key—the same key value as was used on the sending VPN device—to decrypt the data.

While Figure 14-21 shows the basic encryption process, Figure 14-22 shows a broader view of IPsec VPNs in an enterprise. First, devices use some related VPN technology like Generic Routing Encapsulation (GRE) to create the concept of a tunnel (a virtual link between the routers), with three such tunnels shown in the figure. Without IPsec, each GRE tunnel could be used to forward unencrypted traffic over the Internet. IPsec adds the security features to the data that flows over the tunnel. (Note that the figure shows IPsec and GRE, but IPsec teams with other VPN technologies as well.)

Key Topic

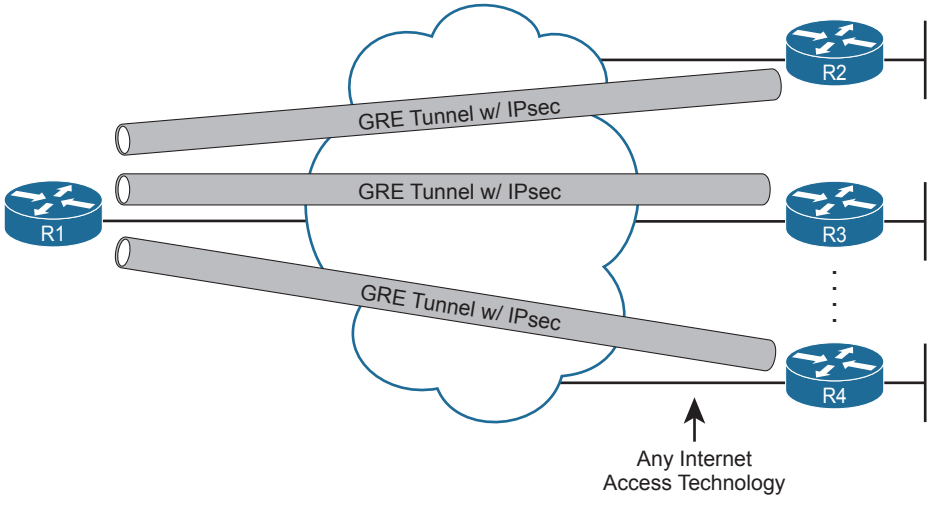


Figure 14-22 *Site-to-Site VPN Tunnels with GRE and IPsec*

Remote Access VPNs with TLS

A site-to-site VPN exists to support multiple devices at each site and is typically created by devices supported by the IT staff. In contrast, individual devices can dynamically initiate their own VPN connections in cases where a permanent site-to-site VPN does not exist. For instance, a user can walk into a coffee shop and connect to the free Wi-Fi, but that coffee shop does not have a site-to-site VPN to the user's enterprise network. Instead, the user's device creates a secure remote access VPN connection back to the enterprise network before sending any data to hosts in the enterprise.

While IPsec and GRE (or other) tunnels work well for site-to-site VPNs, remote access VPNs often use the Transport Layer Security (TLS) protocol to create a secure VPN session.

TLS has many uses today, but most commonly, TLS provides the security features of HTTP Secure (HTTPS). Today's web browsers support HTTPS (with TLS) as a way to dynamically create a secure connection from the web browser to a web server, supporting safe online access to financial transactions. To do so, the browser creates a TCP connection to server well-known port 443 (default) and then initializes a TLS session. TLS encrypts data sent between the browser and the server and authenticating the user. Then, the HTTP messages flow over the TLS VPN connection.

NOTE In years past, Secure Sockets Layer (SSL) played the same role as TLS. SSL has been deprecated (see RFC 7568) and has been replaced by TLS.

The built-in TLS functions of a web browser create one secure web browsing session, but each session secures only the data sent in that session. This same TLS technology can be used to create a client VPN that secures all packets from the device to a site by using a *Cisco VPN client*. The Cisco AnyConnect Secure Mobility Client (or AnyConnect Client for short) is software that sits on a user's PC and uses TLS to create one end of a VPN remote-access tunnel. As a result, all the packets sent to the other end of the tunnel are encrypted, not just those sent over a single HTTP connection in a web browser.

Figure 14-23 compares the option to create a VPN remote access VPN session from a computer to a site versus for a single HTTPS session. The figure shows a VPN tunnel for PC using the AnyConnect Client to create a client VPN. The AnyConnect Client creates a TLS tunnel to the firewall that has been installed to expect VPN clients to connect to it. The tunnel encrypts all traffic so that PC A can use any application available at the enterprise network on the right.

Key Topic

Cisco AnyConnect Secure Mobility Client

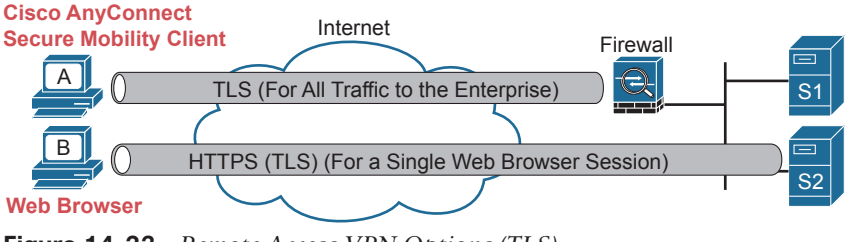


Figure 14-23 *Remote Access VPN Options (TLS)*

Note that while the figure shows a firewall used at the main enterprise site, many types of devices can be used on the server side of a TLS connection as well.

The bottom of Figure 14-23 shows a client VPN that supports a web application for a single web browser tab. The experience is much like when you connect to any other secure website today: the session uses TLS, so all traffic sent to and from that web browser tab is encrypted with TLS. Note that PC B does not use the AnyConnect Client; the user simply opens a web browser to browse to server S2.

VPN Comparisons

The CCNA 200-301 exam topics mention the terms *site-to-site VPN* and *remote access VPN*. To close the section, Table 14-4 lists several key comparison points between the two technologies for easier review and comparison.

Key Topic

Table 14-4 Comparisons of Site-to-Site and Remote Access VPNs

	Remote Access	Site-to-Site
Typical security protocol	TLS	IPsec
Devices supported by one VPN (one or many)	One	Many
Typical use: on-demand or permanent	On-demand	Permanent

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 14-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 14-5 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used:
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website

Review All the Key Topics

Key Topic

Table 14-6 Key Topics for Chapter 14

Key Topic Element	Description	Page Number
Figure 14-2	Metro Ethernet terminology in context	305
Table 14-3	MetroE service types per MEF	306
Figure 14-5	MetroE Ethernet LAN (E-LAN) service concept	309
Figure 14-6	MetroE Ethernet Tree (E-Tree) service concept	309
List	Ideas about customer Layer 3 addressing and what an MPLS VPN provider needs to know	313
Figure 14-11	MPLS terminology in context	314
List	Ideas about routing protocol neighbor relationships with MPLS VPN	316
Figure 14-22	Concepts of site-to-site VPNs with IPsec and GRE	324
Figure 14-23	Concepts of remote access VPNS with TLS	325
Table 14-4	Comparisons between site-to-site and remote access VPNs	326

Key Terms You Should Know

point-to-point, hub and spoke, partial mesh, full mesh, Ethernet WAN, Metro Ethernet, service provider (SP), point of presence (PoP), access link, E-Line, E-LAN, E-Tree, Multiprotocol Label Switching (MPLS), MPLS VPN, customer edge (CE), provider edge (PE), Multiprotocol BGP (MPBGP), IPsec, shared key, TLS, remote access VPN, site-to-site VPN, Cisco AnyConnect Secure Mobility Client

CHAPTER 15

Cloud Architecture

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.g Servers

1.2 Describe the characteristics of network topology architectures

1.2.f On-premises and cloud

1.12 Explain virtualization fundamentals (virtual machines)

Cloud computing is an approach to offering IT services to customers. However, cloud computing is not a product, or a set of products, a protocol, or any single thing. So, while there are accepted descriptions and definitions of cloud computing today, it takes a broad knowledge of IT beyond networking to know whether a particular IT service is or is not worthy of being called a cloud computing service.

Cloud computing, or cloud, is an approach as to how to offer services to customers. For an IT service to be considered to be cloud computing, it should have these characteristics: It can be requested on-demand; it can dynamically scale (that is, it is elastic); it uses a pool of resources; it has a variety of network access options; and it can be measured and billed back to the user based on the amount used. Cloud computing relies on data centers that can be automated. For instance, to service requests, a cloud computing system will create virtual server instances—virtual machines (VMs)—and configure the settings on each VM to provide the requested service.

This chapter gives you a general idea of the cloud services and network architecture. To do that, this chapter begins with a discussion of server virtualization basics. The next section then discusses the big ideas in cloud computing, with the final section discussing the impact of public clouds on packet flows in enterprise networks.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 15-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Server Virtualization	1, 2
Cloud Computing Concepts	3, 4
WAN Traffic Paths to Reach Cloud Services	5, 6

1. Three virtual machines run on one physical server. Which of the following server resources are commonly virtualized so each VM can use the required amount of that resource? (Choose three answers.)
   a. NIC
   b. RAM
   c. Power
   d. Hypervisor
   e. CPU
2. Eight virtual machines run on one physical server; the server has two physical Ethernet NICs. Which answer describes a method that allows all eight VMs to communicate?
   a. The VMs must share two IP addresses and coordinate to avoid using duplicate TCP or UDP ports.
   b. The hypervisor acts as an IP router using the NICs as routed IP interfaces.
   c. Each VM uses a virtual NIC that is mapped to a physical NIC.
   d. Each VM uses a virtual NIC that logically connects to a virtual switch.
3. Which of the following cloud services is most likely to be used for software development?
   a. IaaS
   b. PaaS
   c. SaaS
   d. SLBaaS
4. Which of the following cloud services is most likely to be purchased and then used to later install your own software applications?
   a. IaaS
   b. PaaS
   c. SaaS
   d. SLBaaS

5. An enterprise plans to start using a public cloud service and is considering different WAN options. The answers list four options under consideration. Which one option has the most issues if the company chooses one cloud provider but then later wants to change to use a different cloud provider instead?
   a. Using private WAN connections directly to the cloud provider
   b. Using an Internet connection without VPN
   c. Using an intercloud exchange
   d. Using an Internet connection with VPN
6. An enterprise plans to start using a public cloud service and is considering different WAN options. The answers list four options under consideration. Which options provide good security by keeping the data private while also providing good QoS services? (Choose two answers.)
   a. Using private WAN connections directly to the cloud provider
   b. Using an Internet connection without VPN
   c. Using an intercloud exchange
   d. Using an Internet connection with VPN

Foundation Topics

Server Virtualization

When you think of a server, what comes to mind? Is it a desktop computer with a fast CPU? A desktop computer with lots of RAM? Is it hardware that would not sit upright on the floor but could be easily bolted into a rack in a data center? When you think of a server, do you not even think of hardware, but of the server operating system (OS), running somewhere as a virtual machine (VM)?

All those answers are accurate from one perspective or another, but in most every other discussion within the scope of the CCNA certification, we ignore those details. From the perspective of most CCNA discussions, a server is a place to run applications, with users connecting to those applications over the network. The book then represents the server with an icon that looks like a desktop computer (that is the standard Cisco icon for a server). This next topic breaks down some different perspectives on what it means to be a server and prepares us to then discuss cloud computing.

Cisco Server Hardware

Think about the form factor of servers for a moment—that is, the shape and size of the physical server. If you were to build a server of your own, what would it look like? How big, how wide, how tall, and so on? Even if you have never seen a device characterized as a server, consider these key facts:

No KVM: For most servers, there is no permanent user who sits near the server; all the users and administrators connect to the server over the network. As a result, there is no need for a permanent keyboard, video display, or mouse (collectively referred to as KVM).

Racks of servers in a data center: In the early years of servers, a server was any computer with relatively fast CPU, large amounts of RAM, and so on. Today, companies put many servers into one room—a data center—and one goal is to not waste space. So, making servers with a form factor that fits in a standard rack makes for more efficient use of the available space—especially when you do not expect people to be sitting in front of each server.

As an example, Figure 15-1 shows a photo of server hardware from Cisco. While you might think of Cisco as a networking company, around 2010, Cisco expanded its product line into the server market, with the Cisco Unified Computing System (UCS) product line. The photo shows a product from the UCS B-Series (Blade series) that uses a rack-mountable chassis, with slots for server blades. The product shown in the figure can be mounted in a rack—note the holes on the sides—with eight server blades (four on each side) mounted horizontally. It also has four power supplies at the bottom of the chassis.

Figure 15-1 *Cisco UCS Servers: B-Series (Blade)*

No matter the form factor, server hardware today supplies some capacity of CPU chips, RAM, storage, and network interface cards (NIC). But you also have to think differently about the OS that runs on the server because of a tool called *server virtualization*.

Server Virtualization Basics

Think of a server—the hardware—as one computer. It can be one of the blades in Figure 15-1, a powerful computer you can buy at the local computer store…whatever. Traditionally, when you think of one server, that one server runs one OS. Inside, the hardware includes a CPU, some RAM, some kind of permanent storage (like disk drives), and one or more NICs. And that one OS can use all the hardware inside the server and then run one or more applications. Figure 15-2 shows those main ideas.

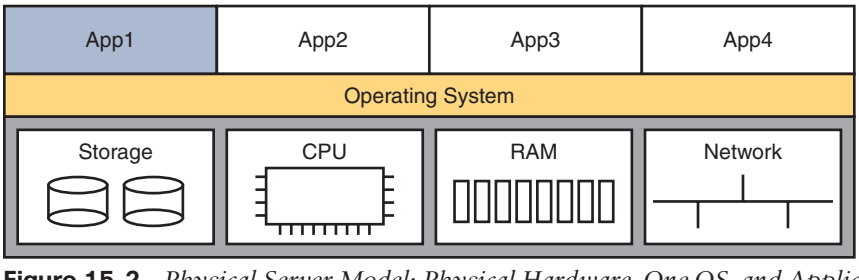


Figure 15-2 *Physical Server Model: Physical Hardware, One OS, and Applications*

With the physical server model shown in Figure 15-2, each physical server runs one OS, and that OS uses all the hardware in that one server. That was true of servers in the days before server virtualization.

Today, most companies instead create a virtualized data center. That means the company purchases server hardware, installs it in racks, and then treats all the CPU, RAM, and so on as capacity in the data center. Then, each OS instance is decoupled from the hardware and is therefore virtual (in contrast to physical). Each piece of hardware that we would formerly have thought of as a server runs multiple instances of an OS at the same time, with each virtual OS instance called a *virtual machine*, or VM.

A single physical host (server) often has more processing power than you need for one OS. Thinking about processors for a moment, modern server CPUs have multiple cores (processors) in a single CPU chip. Each core may also be able to run multiple threads with a feature called *multithreading*. So, when you read about a particular Intel processor with 8 cores and multithreading (typically two threads per core), that one CPU chip can execute 16 different programs concurrently. The hypervisor (introduced shortly) can then treat each available thread as a virtual CPU (vCPU) and give each VM a number of vCPUs, with 16 available in this example.

A VM—that is, an OS instance that is decoupled from the server hardware—still must execute on hardware. Each VM has configuration as to the minimum number of vCPUs it needs, minimum RAM, and so on. The virtualization system then starts each VM on some physical server so that enough physical server hardware capacity exists to support all the VMs running on that host. So, at any one point in time, each VM is running on a physical server, using a subset of the CPU, RAM, storage, and NICs on that server. Figure 15-3 shows a graphic of that concept, with four separate VMs running on one physical server.

Key Topic

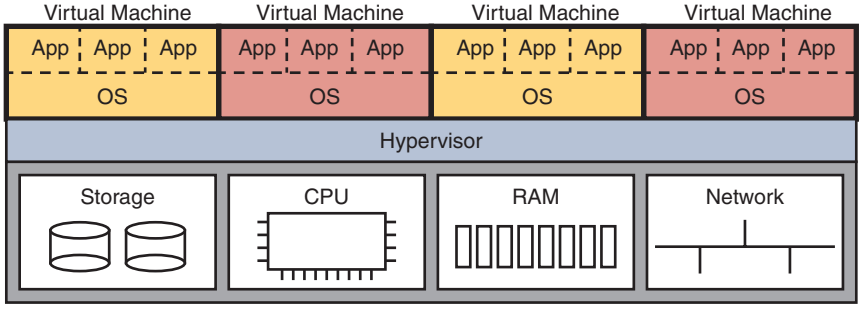


Figure 15-3 *Four VMs Running on One Host; Hypervisor Manages the Hardware*

To make server virtualization work, each physical server (called a *host* in the server virtualization world) uses a *hypervisor*. The hypervisor manages and allocates the host hardware (CPU, RAM, etc.) to each VM based on the settings for the VM. Each VM runs as if it is running on a self-contained physical server, with a specific number of virtual CPUs and NICs and a set amount of RAM and storage. For instance, if one VM happens to be configured to use four CPUs, with 8 GB of RAM, the hypervisor allocates the specific parts of the CPU and RAM that the VM actually uses.

Answers to the "Do I Know This Already?" quiz:

1 A, B, E **2** D **3** B **4** A **5** A **6** A, C

To connect the marketplace to the big ideas discussed thus far, the following list includes a few of the vendors and product family names associated with virtualized data centers:

- VMware vCenter
- Microsoft HyperV
- Citrix XenServer
- Red Hat KVM

Beyond the hypervisor, companies like those in the list (and others) sell complete virtualization systems. These systems allow virtualization engineers to dynamically create VMs, start them, move them (manually and automatically) to different servers, and stop them. For instance, when hardware maintenance needs to be performed, the virtualization engineer can move the VMs to another host (often while running) so that the maintenance can be done.

Networking with Virtual Switches on a Virtualized Host

Server virtualization tools provide a wide variety of options for how to connect VMs to networks. This book does not attempt to discuss them all, but it can help to get some of the basics down before thinking more about cloud computing.

First, what does a physical server include for networking functions? Typically it has one or more NICs, maybe as slow as 1 Gbps, often 10 Gbps today, and maybe as fast as 40 Gbps.

Next, think about the VMs. Normally, an OS has one NIC, maybe more. To make the OS work as normal, each VM has (at least) one NIC, but for a VM, it is a virtual NIC. (For instance, in VMware's virtualization systems, the VM's virtual NIC goes by the name vNIC.)

Finally, the server must combine the ideas of the physical NICs with the vNICs used by the VMs into some kind of a network. Most often, each server uses some kind of an internal Ethernet switch concept, often called (you guessed it) a virtual switch, or vSwitch. Figure 15-4 shows an example, with four VMs, each with one vNIC. The physical server has two physical NICs. The vNICs and physical NICs connect internally to a virtual switch.

Key Topic

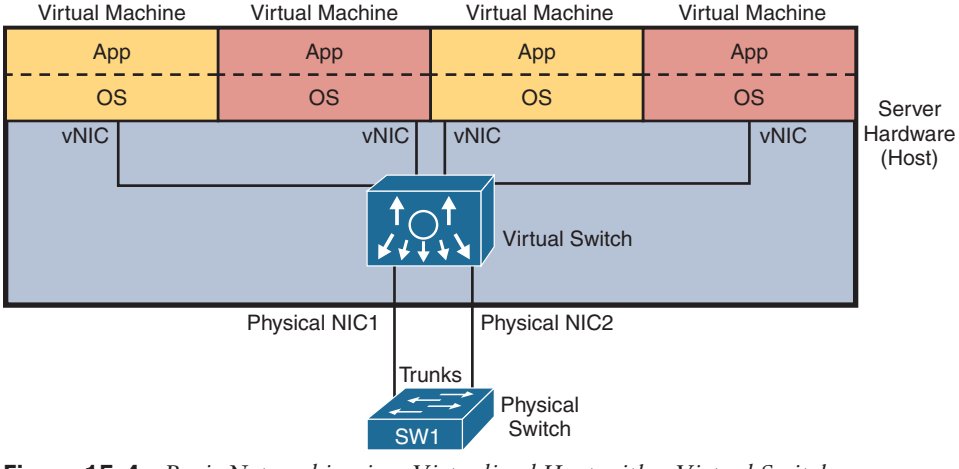


Figure 15-4 *Basic Networking in a Virtualized Host with a Virtual Switch*

Interestingly, the vSwitch can be supplied by the hypervisor vendor or by Cisco. For instance, Cisco offers the Nexus 1000VE virtual switch (which replaces the older and popular Nexus 1000V virtual switch). The Nexus 1000VE runs the NX-OS operating system found in some of the Cisco Nexus data center switch product line. Additionally, Cisco offers the Cisco ACI Virtual Edge, another virtual switch, this one following Cisco ACI networking as detailed in Chapter 16, "Introduction to Controller-Based Networking."

The vSwitch shown in Figure 15-4 uses the same networking features you now know from your CCNA studies; in fact, one big motivation to use a vSwitch from Cisco is to use the same networking features, with the same configuration, as in the rest of the network. In particular:

- **Ports connected to VMs:** The vSwitch can configure a port so that the VM will be in its own VLAN, or share the same VLAN with other VMs, or even use VLAN trunking to the VM itself.
- **Ports connected to physical NICs:** The vSwitch uses the physical NICs in the server hardware so that the switch is adjacent to the external physical LAN switch. The vSwitch can (and likely does) use VLAN trunking.
- **Automated configuration:** The configuration can be easily done from within the same virtualization software that controls the VMs. That programmability allows the virtualization software to move VMs between hosts (servers) and reprogram the vSwitches so that the VM has the same networking capabilities no matter where the VM is running.

The Physical Data Center Network

To pull these ideas together, next consider what happens with the physical network in a virtualized data center. Each host—that is, the physical host—needs a physical connection to the network. Looking again at Figure 15-4, that host, with two physical NICs, needs to connect those two physical NICs to a LAN switch in the data center.

Figure 15-5 shows the traditional cabling for a data center LAN. Each taller rectangle represents one rack inside a data center, with the tiny squares representing NIC ports, and the lines representing cables.

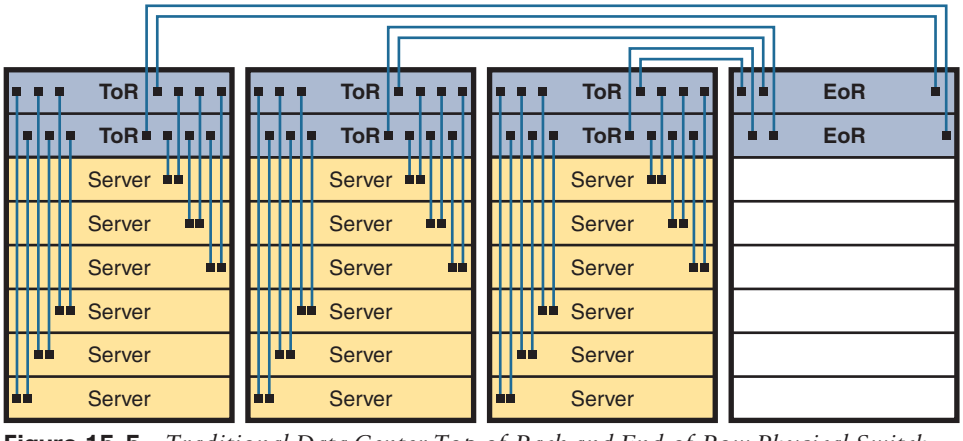


Figure 15-5 *Traditional Data Center Top-of-Rack and End-of-Row Physical Switch Topology*

Often, each host is cabled to two different switches in the top of the rack—called Top of Rack (ToR) switches—to provide redundant paths into the LAN. Each ToR switch acts as an access layer switch from a design perspective. Each ToR switch is then cabled to an End of Row (EoR) switch, which acts as a distribution switch and also connects to the rest of the network.

The design in Figure 15-5 uses a traditional data center cabling plan. Some data center technologies call for different topologies, in particular, Cisco Application Centric Infrastructure (ACI). ACI places the server and switch hardware into racks, but cables the switches with a different topology—a topology required for proper operation of the ACI fabric. Chapter 16 introduces ACI concepts.

Workflow with a Virtualized Data Center

So far, the first part of this chapter has described background information important to the upcoming discussions of cloud computing. Server virtualization has been a great improvement to the operations of many data centers, but virtualization alone does not create a cloud computing environment. Continuing the discussion of these fundamental technologies before discussing cloud computing, consider this example of a workflow through a virtualized (not cloud-based) data center.

Some of the IT staff, call them server or virtualization engineers or administrators, order and install new hosts (servers). They gather requirements, plan for the required capacity, shop for hardware, order it, and install the hardware. They play the role of long-time server administrators and engineers, but now they work with the virtualization tools as well.

For the virtualization parts of the effort, the virtualization engineers also install and customize the virtualization tools. Beyond the hypervisor on each host, many other useful tools help manage and control a virtualized data center. For instance, one tool might give the engineers a view of the data center as a whole, with all VMs running there, with the idea that one data center is just a lot of capacity to run VMs. Over time, the server/virtualization engineers add new physical servers to the data center and configure the virtualization systems to make use of the new physical servers and make sure it all works.

So far in this scenario, the work has been in preparation for providing services to some internal customer—a development team member, the operations staff, and so on. Now, a customer is requesting a "server." In truth, the customer wants a VM (or many), with certain requirements: a specific number of vCPUs, a specific amount of RAM, and so on. The customer makes a request to the virtualization/server engineer to set up the VMs, as shown in Figure 15-6.

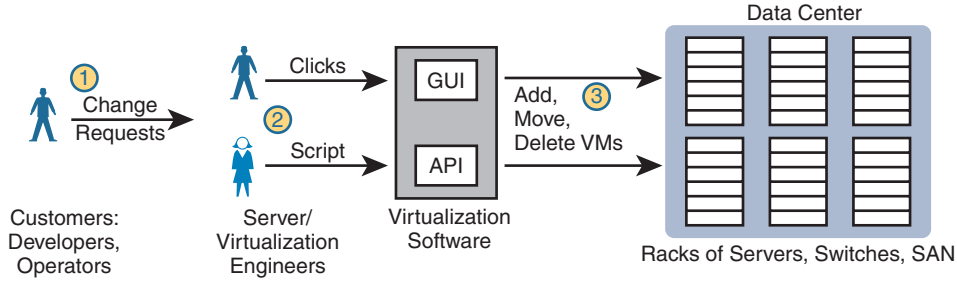


Figure 15-6 *Traditional Workflow: Customer (Human) Asks Virtualization (Human) for Service*

The figure emphasizes what happens after the customer makes a request, which flows something like this:

Step 1. The customer of the IT group, such as a developer or a member of the operations staff, wants some service, like a set of new VMs.

Step 2. The virtualization/server engineer reacts to the request from the customer. The server/virtualization engineer clicks away at the user interface, or if the number of VMs is large, she often runs a program called a script to more efficiently create the VMs.

Step 3. Regardless of whether the virtualization engineer clicked or used scripts, the virtualization software could then create a number of new VMs and start those on some hosts inside the data center.

The process shown in Figure 15-6 works great. However, that approach to providing services breaks some of the basic criteria of a cloud service. For instance, cloud computing requires self-service. For the workflow to be considered to be a cloud service, the process at step 2 should not require a human to service that request, but instead the request should be filled automatically. Want some new VMs in a cloud world? Click a user interface to ask for some new VMs, go get a cup of coffee, and your VMs will be set up and started, to your specification, in minutes.

Summarizing some of the key points about a virtualized data center made so far, which enable cloud computing:

- The OS is decoupled from the hardware on which it runs, so that the OS, as a VM, can run on any server in a data center that has enough resources to run the VM.
- The virtualization software can automatically start and move the VM between servers in the data center.
- Data center networking includes virtual switches and virtual NICs within each host (server).
- Data center networking can be programmed by the virtualization software, allowing new VMs to be configured, started, moved as needed, and stopped, with the networking details configured automatically.

Cloud Computing Services

Cloud computing is an approach to offering IT services. Cloud computing makes use of products such as the virtualization products but also uses products built specifically to enable cloud features. However, cloud computing is not just a set of products to be implemented; instead, it is a way of offering IT services. So, understanding what cloud computing is—and is not—takes a little work; this next topic introduces the basics.

From the just-completed discussions about virtualization, you already know one characteristic of a cloud service: it must allow self-service provisioning by the consumer of the service. That is, the consumer or customer of the service must be able to request the service and receive that service without the delay of waiting for a human to have time to work on it, consider the request, do the work, and so on.

To get a broader sense of what it means for a service to be a cloud service, examine this list of five criteria for a cloud computing service. The list is derived from the definition of

cloud computing as put forth by the US National Institute of Standards and Technology (NIST):

Key Topic

On-demand self-service: The IT consumer chooses when to start and stop using the service, without any direct interaction with the provider of the service.

Broad network access: The service must be available from many types of devices and over many types of networks (including the Internet).

Resource pooling: The provider creates a pool of resources (rather than dedicating specific servers for use only by certain consumers) and dynamically allocates resources from that pool for each new request from a consumer.

Rapid elasticity: To the consumer, the resource pool appears to be unlimited (that is, it expands quickly, so it is called *elastic*), and the requests for new service are filled quickly.

Measured service: The provider can measure the usage and report that usage to the consumer, both for transparency and for billing.

Keep this list of five criteria in mind while you work through the rest of the chapter. Later parts of the chapter will refer back to the list.

To further develop this definition, the next few pages look at two branches of the cloud universe—private cloud and public cloud—also with the goal of further explaining some of the points from the NIST definition.

Private Cloud (On-Premise)

Look back to the workflow example in Figure 15-6 with a virtualized data center. Now think about the five NIST criteria for cloud computing. If you break down the list versus the example around Figure 15-6, it seems like the workflow may meet at least some of these five NIST cloud criteria, and it does. In particular, as described so far in this chapter, a virtualized data center pools resources so they can be dynamically allocated. You could argue that a virtualized data center is elastic, in that the resource pool expands. However, the process may not be rapid because the workflow requires human checks, balances, and time before provisioning new services.

Private cloud creates a service, inside a company, to internal customers, that meets the five criteria from the NIST list. To create a private cloud, an enterprise often expands its IT tools (like virtualization tools), changes internal workflow processes, adds additional tools, and so on.

NOTE The world of cloud computing has long used the terms *private cloud* and *public cloud*. In more recent years, you may also find references that instead use a different pair of terms for the same ideas, with *on-premise* meaning *private cloud*, and *cloud* meaning *public cloud*. Note that the one CCNA 200-301 exam topic that mentions cloud happens to use the newer pair of terms.

As some examples, consider what happens when an application developer at a company needs VMs to use when developing an application. With private cloud, the developer can request those VMs and those VMs automatically start and are available within minutes, with most of the time lag being the time to boot the VMs. If the developer wants many more VMs, he can assume that the private cloud will have enough capacity, and new requests are

still serviced rapidly. And all parties should know that the IT group can measure the usage of the services for internal billing.

Focus on the self-service aspect of cloud for a moment. To make that happen, many cloud computing services use a *cloud services catalog*. That catalog exists for the user as a web application that lists anything that can be requested via the company's cloud infrastructure. Before using a private cloud, developers and operators who needed new services (like new VMs) sent a change request asking the virtualization team to add VMs (see Figure 15-6). With private cloud, the (internal) consumers of IT services—developers, operators, and the like—can click to choose from the cloud services catalog. And if the request is for a new set of VMs, the VMs appear and are ready for use in minutes, without human interaction for that step, as seen at step 2 of Figure 15-7.

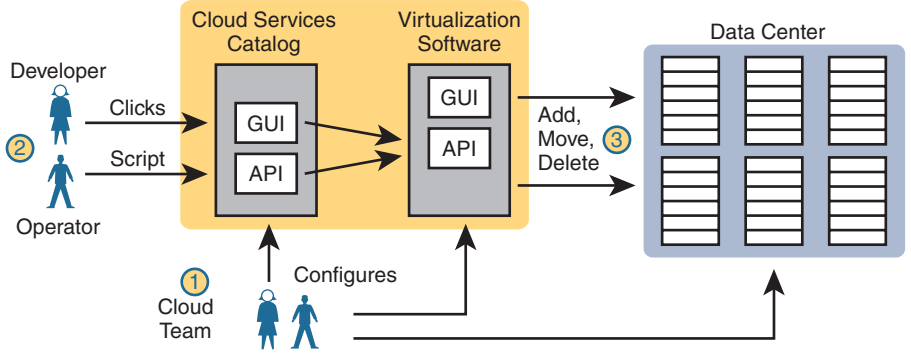


Figure 15-7 *Basic Private Cloud Workflow to Create One VM*

To make this process work, the cloud team has to add some tools and processes to its virtualized data center. For instance, it installs software to create the cloud services catalog, both with a user interface and with code that interfaces to the APIs of the virtualization systems. That services catalog software can react to consumer requests, using APIs into the virtualization software, to add, move, and create VMs, for instance. Also, the cloud team—composed of server, virtualization, network, and storage engineers—focuses on building the resource pool, testing and adding new services to the catalog, handling exceptions, and watching the reports (per the measured service requirement) to know when to add capacity to keep the resource pool ready to handle all requests.

Notably, with the cloud model, the cloud team no longer spends time handling individual requests for adding 10 VMs here, 50 there, with change requests from different groups.

Summarizing, with private cloud, you change your methods and tools to offer some of the same services. Private cloud is "private" in that one company owns the tools that create the cloud and employs the people who use the services. Even inside one company, using a cloud computing approach can improve the operational speed of deploying IT services.

Public Cloud

With a private cloud, the cloud provider and the cloud consumer are part of the same company. With public cloud, the reverse is true: a public cloud provider offers services, selling those services to consumers in other companies. In fact, if you think of Internet service providers and WAN service providers selling Internet and WAN services to many enterprises, the same general idea works here with public cloud providers selling their services to many enterprises.

The workflow in public cloud happens somewhat like private cloud when you start from the point of a consumer asking for some service (like a new VM). As shown on the right of Figure 15-8, at step 1, the consumer asks for the new service from the service catalog web page. At step 2, the virtualization tools react to the request to create the service. Once started, the services are available, but running in a data center that resides somewhere else in the world, and certainly not at the enterprise's data center (step 3).

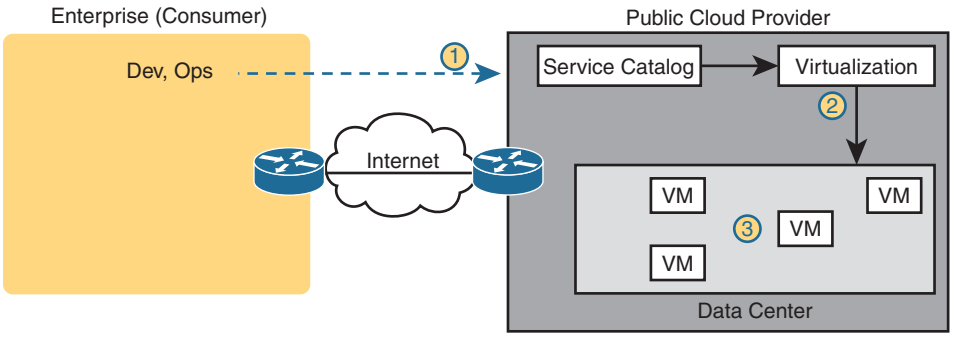


Figure 15-8 *Public Cloud Provider in the Internet*

Of course, the consumer is in a different network than the cloud provider with cloud computing, which brings up the issue of how to connect to a cloud provider. Cloud providers support multiple network options. They each connect to the Internet so that apps and users inside the consumer's network can communicate with the apps that the consumer runs in the cloud provider's network. However, one of the five NIST criteria for cloud computing is broad network access, so cloud providers offer different networking options as well, including virtual private network (VPN) and private wide-area network (WAN) connections between consumers and the cloud.

Cloud and the "As a Service" Model

So what do you get with cloud computing? So far, this chapter has just shown a VM as a service. With cloud computing, there are a variety of services, and three stand out as the most common seen in the market today.

First, a quick word about some upcoming terminology. The cloud computing world works on a services model. Instead of buying (consuming) hardware, buying or licensing software, installing it yourself, and so on, the consumer receives some service from the provider. But that idea, receiving a service, is more abstract than the idea of buying a server and installing a particular software package. So with cloud computing, instead of keeping the discussion so generic, the industry uses a variety of terms that end in "as a Service." And each "-aaS" term has a different meaning.

This next topic explains those three most common cloud services: Infrastructure as a Service, Software as a Service, and Platform as a Service.

Infrastructure as a Service

Infrastructure as a Service (IaaS) may be the easiest of the cloud computing services to understand for most people. For perspective, think about any time you have shopped for a computer. You thought about the OS to run (the latest Microsoft OS, or Linux, or macOS if shopping for a Mac). You compared prices based on the CPU and its speed, how much RAM the computer had, the size of the disk drive, and so on.

IaaS offers a similar idea, but the consumer receives the use of a VM. You specify the amount of hardware performance/capacity to allocate to the VM (number of virtual CPUs, amount of RAM, and so on), as shown in Figure 15-9. You can even pick an OS to use. Once selected, the cloud provider starts the VM, which boots the chosen OS.

NOTE In the virtualization and cloud world, starting a VM is often called *spinning up a VM* or *instantiating a VM*.

Key Topic

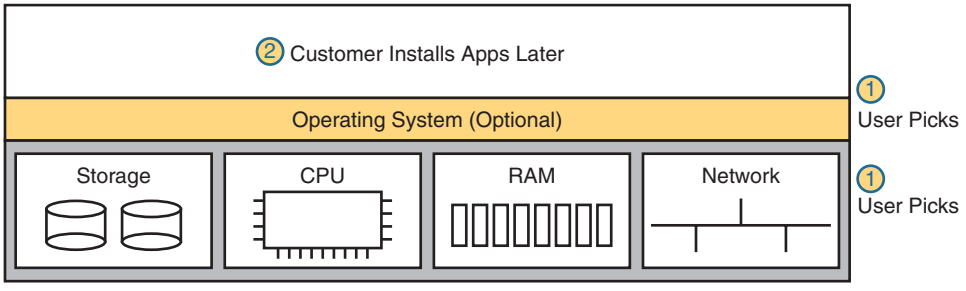


Figure 15-9 *IaaS Concept*

The provider also gives the consumer details about the VM so the consumer can connect to the OS's user interface, install more software, and customize settings. For example, imagine that the consumer wants to run a particular application on the server. If that customer wanted to use Microsoft Exchange as an email server, she would then need to connect to the VM and install Exchange.

Figure 15-10 shows a web page from Amazon Web Services (AWS), a public cloud provider, from which you could create a VM as part of its IaaS service. The screenshot shows that the user selected a small VM called "micro." If you look closely at the text, you may be able to read the heading and numbers to see that this particular VM has one vCPU and 1 GB of RAM.

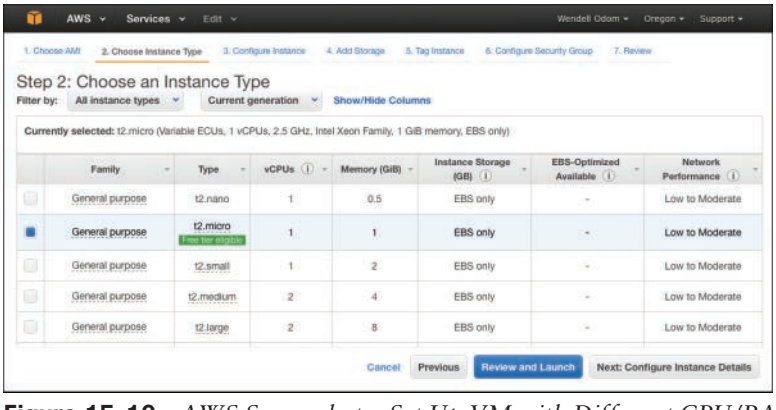


Figure 15-10 *AWS Screenshot—Set Up VM with Different CPU/RAM/OS*

Software as a Service

With Software as a Service (SaaS), the consumer receives a service with working software. The cloud provider may use VMs, possibly many VMs, to create the service, but those are hidden from the consumer. The cloud provider licenses, installs, and supports whatever software is required. The cloud provider then monitors performance of the application. However, the consumer chooses to use the application, signs up for the service, and starts using the application—no further installation work required. Figure 15-11 shows these main concepts.

Key Topic

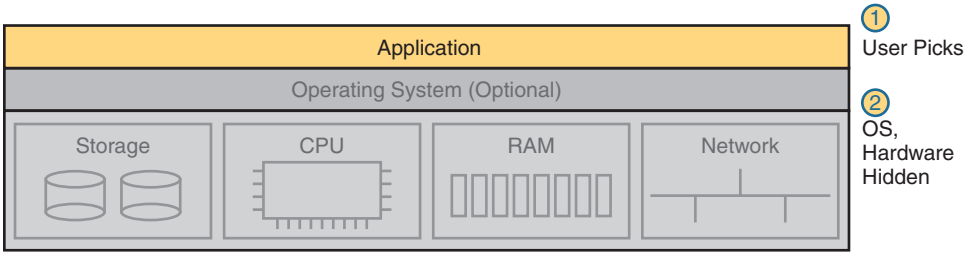


Figure 15-11 *SaaS Concept*

Many of you have probably used or at least heard of many public SaaS offerings. File storage services like Apple iCloud, Google Drive, Dropbox, and Box are all SaaS offerings. Most online email offerings can be considered SaaS services today. As another example, Microsoft offers its Exchange email server software as a service, so you can have private email services but offered as a service, along with all the other features included with Exchange—without having to license, install, and maintain the Exchange software on some VMs.

(Development) Platform as a Service

Platform as a Service (PaaS) is a development platform, prebuilt as a service. A PaaS service is like IaaS in some ways. Both supply the consumer with one or more VMs, with a configurable amount of CPU, RAM, and other resources.

The key difference between PaaS and IaaS is that PaaS includes many more software tools beyond the basic OS. Those tools are useful to a software developer during the software development process. Once the development process is complete, and the application has been rolled out in production, those tools are not needed on the servers running the application. So the development tools are particular to the work done when developing.

A PaaS offering includes a set of development tools, and each PaaS offering has a different combination of tools. PaaS VMs often include an integrated development environment (IDE), which is a set of related tools that enables the developer to write and test code easily. PaaS VMs include continuous integration tools that allow the developer to update code and have that code automatically tested and integrated into a larger software project. Examples include Google's App Engine PaaS offering (https://cloud.google.com/appengine), the Eclipse integrated development environment (see www.eclipse.org), and the Jenkins continuous integration and automation tool (see https://jenkins.io).

The primary reasons to choose one PaaS service over another, or to choose a PaaS solution instead of IaaS, is the mix of development tools. If you do not have experience as a developer, it can be difficult to tell whether one PaaS service might be better. You can still make

some choices about sizing the PaaS VMs, similar to IaaS tools when setting up some PaaS services, as shown in Figure 15-12, but the developer tools included are the key to a PaaS service.

Key Topic

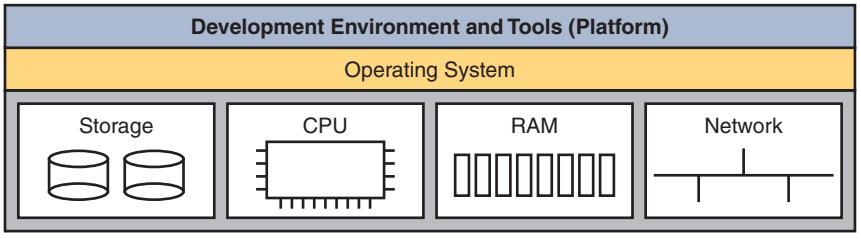


Figure 15-12 *PaaS Concept*

WAN Traffic Paths to Reach Cloud Services

This final major section of the chapter focuses on WAN options for public cloud, and the pros and cons of each. This section ignores private cloud for the most part, because using a private cloud—which is internal to an enterprise—has much less of an impact on an enterprise WAN compared to public cloud. With public cloud, the cloud services exist on the other side of some WAN connection as compared to the consumer of the services, so network engineers must think about how to best build a WAN when using public cloud services.

Enterprise WAN Connections to Public Cloud

Using the Internet to communicate between the enterprise and a public cloud provider is easy and convenient. However, it also has some negatives. This first section describes the basics and points out the issues, which then leads to some of the reasons why using other WAN connections may be preferred.

Accessing Public Cloud Services Using the Internet

Imagine an enterprise that operates its network without cloud. All the applications it uses to run its business run on servers in a data center inside the enterprise. The OS instances where those applications run can be hosted directly on physical servers or on VMs in a virtualized data center, but all the servers exist somewhere inside the enterprise.

Now imagine that the IT staff starts moving some of those applications out to a public cloud service. How do the users of the application (inside the enterprise) get to the user interface of the application (which runs at the public cloud provider's data center)? The Internet, of course. Both the enterprise and the cloud provider connect to the Internet, so using the Internet is the easy and convenient choice.

Now consider a common workflow to move an internal application to now run on the public cloud, for the purpose of making a couple of important points. First, Figure 15-13 shows the example. The cloud provider's services catalog can be reached by enterprise personnel, over the Internet, as shown at step 1. After choosing the desired services—for instance, some VMs for an IaaS service—the cloud provider (step 2) instantiates the VMs. Then, not shown as a step in the figure, the VMs are customized to now run the app that was formerly running inside the enterprise's data center.

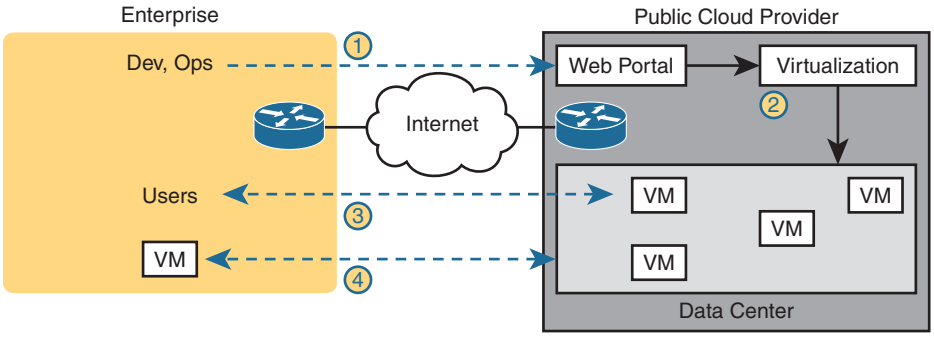


Figure 15-13 *Accessing a Public Cloud Service Using the Internet*

At this point, the new app is running in the cloud, and those services will require network bandwidth. In particular, step 3 shows users communicating with the applications, just as would happen with any other application. Additionally, most apps send much more data than just the data between the application and the end user. For instance, you might move an app to the public cloud, but you might keep authentication services on an internal server because those are used by a large number of applications—some internal and some hosted in the public cloud. So at step 4, any application communication between VMs hosted in the cloud to/from VMs hosted inside the enterprise also needs to take place.

Pros and Cons with Connecting to Public Cloud with Internet

Using the Internet to connect from the enterprise to the public cloud has several advantages. The most obvious advantage is that all companies and cloud providers already have Internet connections, so getting started using public cloud services is easy. Using the Internet works particularly well with SaaS services and a distributed workforce. For instance, maybe your sales division uses a SaaS customer contact app. Often, salespeople do not sit inside the enterprise network most of the work day. They likely connect to the Internet and use a VPN to connect to the enterprise. For apps hosted on the public cloud, with this user base, it makes perfect sense to use the Internet.

While that was just one example, the following list summarizes some good reasons to use the Internet as the WAN connection to a public cloud service:

Agility: An enterprise can get started using public cloud without having to wait to order a private WAN connection to the cloud provider because cloud providers support Internet connectivity.

Migration: An enterprise can switch its workload from one cloud provider to another more easily because cloud providers all connect to the Internet.

Distributed users: The enterprise's users are distributed and connect to the Internet with their devices (as in the sales SaaS app example).

Using the Internet as the WAN connectivity to a public cloud is both a blessing and a curse in some ways. Using the Internet can help you get started with public cloud and to get working quickly, but it also means that you do not have to do any planning before deploying a public cloud service. With a little planning, a network engineer can see some of the negatives of using the Internet—the same negatives when using the Internet for other

purposes—which then might make you want to use alternative WAN connections. Those negatives for using the Internet for public cloud access are

Key Topic

Security: The Internet is less secure than private WAN connections in that a "man in the middle" can attempt to read the contents of data that passes to/from the public cloud.

Capacity: Moving an internal application to the public cloud increases network traffic, so the question of whether the enterprise's Internet links can handle the additional load needs to be considered.

Quality of Service (QoS): The Internet does not provide QoS, whereas private WANs can. Using the Internet may result in a worse user experience than desired because of higher delay (latency), jitter, and packet loss.

No WAN SLA: ISPs typically will not provide a service-level agreement (SLA) for WAN performance and availability to all destinations of a network. WAN service providers are much more likely to offer performance and availability SLAs.

This list of concerns does not mean that an enterprise cannot use the Internet to access its public cloud services. It does mean that it should consider the pros and cons of each WAN option.

Private WAN and Internet VPN Access to Public Cloud

The NIST definition for cloud computing lists broad network access as one of the five main criteria. In the case of public cloud, that often means supporting a variety of WAN connections, including the most common enterprise WAN technologies. Basically, an enterprise can connect to a public cloud provider with WAN technologies discussed in this book. For the sake of discussion, Figure 15-14 breaks it down into two broad categories.

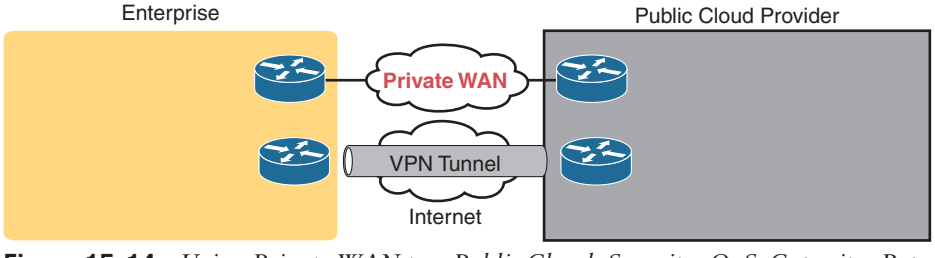


Figure 15-14 *Using Private WAN to a Public Cloud: Security, QoS, Capacity, Reporting*

To create a VPN tunnel between the enterprise and the cloud provider, you can use the same VPN features discussed earlier in Chapter 14, "WAN Architecture." The cloud provider can offer a VPN service—that is, the cloud side of the VPN tunnel is implemented by the cloud provider—and the enterprise configures the matching VPN service on one of its own routers. Or the enterprise can use its own router inside the cloud provider's network—a virtual router, running as a VM—and configure VPN services on that router. In fact, Cisco makes the *Cloud Services Router* (CSR) to do exactly that: to be a router, but a router that runs as a VM in a cloud service, controlled by the cloud consumer, to do various functions that routers do, including terminating VPNs. (Also, by running a virtual router as a VM and managing the configuration internally, the enterprise might save some of the cost of using a similar service offered by the cloud provider.)

To make a private Multiprotocol Label Switching (MPLS) VPN or Ethernet WAN connection, the enterprise needs to work with the cloud provider and the WAN provider. Because cloud providers connect to many customers with private WAN connections, they often have published set instructions to follow. In the most basic form, with MPLS, the enterprise and the cloud provider connect to the same MPLS provider, with the MPLS provider connecting the enterprise and cloud sites. The same basic process happens with Ethernet WAN services, with one or more Ethernet Virtual Connections (EVCs) created between the public WAN and the enterprise.

NOTE Often, the server/virtualization engineers will dictate whether the WAN connection needs to support Layer 2 or Layer 3 connectivity, depending on other factors.

Private WAN connections also require some physical planning. Each of the larger public cloud providers has a number of large data centers spread around the planet and with prebuilt connection points into the major WAN services to aid the creation of private WAN connections to customers. An enterprise might then look at the cloud provider's documentation and work with that provider to choose the best place to install the private WAN connection. (Those larger public cloud companies include Amazon Web Services, Google Compute Cloud, Microsoft Azure, and Rackspace, if you would like to look at their websites for information about their locations.)

Pros and Cons of Connecting to Cloud with Private WANs

Private WANs overcome some of the issues of using the Internet without VPN, so working through those issues, consider some of the different WAN options.

First, considering the issue of security, all the private options, including adding a VPN to the existing Internet connection, improve security significantly. An Internet VPN would encrypt the data to keep it private. Private WAN connections with MPLS and Ethernet have traditionally been considered secure without encryption, but companies are sometimes encrypting data sent over private WAN connections as well to make the network more secure.

Regarding QoS, using an Internet VPN solution still fails to provide QoS because the Internet does not provide QoS. WAN services like MPLS VPN and Ethernet WANs can. As discussed in Chapter 11, "Quality of Service (QoS)," WAN providers will look at the QoS markings for frames/packets sent by the customer and apply QoS tools to the traffic as it passes through the service provider's network.

Finally, as for the capacity issue, the concern of planning network capacity exists no matter what type of WAN is used. Any plan to migrate an app away from an internal data center to instead be hosted as a public cloud provider requires extra thought and planning.

Several negatives exist for using a private WAN, as you might expect. Installing the new private WAN connections takes time, delaying when a company gets started in cloud computing. Private WANs typically cost more than using the Internet. If using a WAN connection to one cloud provider (instead of using the Internet), then migrating to a new cloud provider can require another round of private WAN installation, again delaying work projects. Using the Internet (with or without VPN) would make that migration much easier, but as shown in the next section, a strong compromise solution exists as well.

Intercloud Exchanges

Public cloud computing also introduces a whole new level of competition because a cloud consumer can move his workload from one cloud provider to another. Moving the workload takes some effort, for a variety of reasons beyond the scope of this book. (Suffice it to say that most cloud providers differ in the detail of how they implement services.) But enterprises can and do migrate their workload from one cloud provider to another, choosing a new company for a variety of reasons, including looking for a less expensive cloud provider.

Now focus on the networking connections again. The main negative with using a private WAN for the cloud is that it adds another barrier to migrating to a new public cloud provider. One solution adds easier migration to the use of a private WAN through a cloud service called an intercloud exchange (or simply an intercloud).

Generically, the term *intercloud exchange* has come to be known as a company that creates a private network as a service. First, an intercloud exchange connects to multiple cloud providers on one side. On the other side, the intercloud connects to cloud consumers. Figure 15-15 shows the idea.

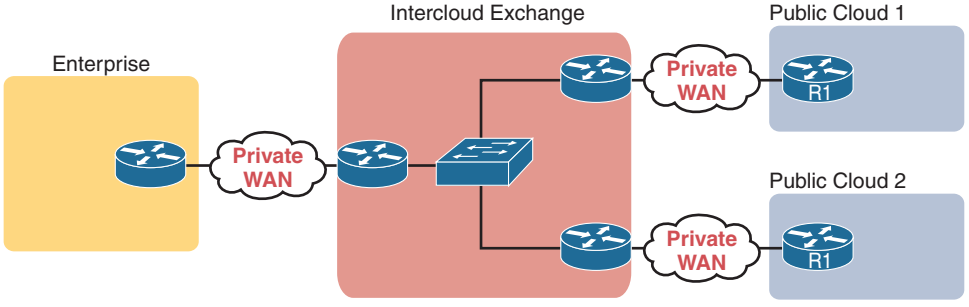


Figure 15-15 *Permanent Private WAN Connection to an Intercloud Exchange*

Once connected, the cloud consumer can be configured to communicate with one public cloud provider today, to specific cloud provider sites. Later, if the consumer wants to migrate to use another cloud provider, the consumer keeps the same private WAN links to the intercloud exchange and asks the provider to reconfigure to set up new private WAN connections to the new cloud provider.

As for pros and cons, with an intercloud exchange, you get the same benefits as when connecting with a private WAN connection to a public cloud, but with the additional pro of easier migration to a new cloud provider. The main con is that using an intercloud exchange introduces another company into the mix.

Summarizing the Pros and Cons of Public Cloud WAN Options

Table 15-2 summarizes some of these key pros and cons for the public WAN options for cloud computing, for study and reference.

Key Topic

Table 15-2 Comparison of Public Cloud WAN Options

	Internet	Internet VPN	MPLS VPN	Ethernet WAN	Intercloud Exchange
Makes data private	No	Yes	Yes	Yes	Yes
Supports QoS	No	No	Yes	Yes	Yes
Requires capacity planning	Yes	Yes	Yes	Yes	Yes
Eases migration to a new provider	Yes	Yes	No	No	Yes
Speeds initial installation	Yes	Yes	No	No	No

A Scenario: Branch Offices and the Public Cloud

So far in this major section about WAN design with public cloud, the enterprise has been shown as one entity, but most enterprise WANs have many sites. Those distributed enterprise sites impact some parts of WAN design for public cloud. The next discussion of WAN design issues with public cloud works through a scenario that shows an enterprise with a typical central site and branch office.

The example used in this section is a common one: the movement away from internal email servers, supported directly by the IT staff, to email delivered as a SaaS offering. Focus on the impact of the enterprise's remote sites like branch offices.

Migrating Traffic Flows When Migrating to Email SaaS

First, think of the traffic flow inside an enterprise before SaaS, when the company buys servers, licenses email server software, installs the hardware and software in an internal data center, and so on. The company may have hundreds or thousands of remote sites, like the branch office shown in Figure 15-16. To check email, an employee at the branch office sends packets back and forth with the email server at the central site, as shown.

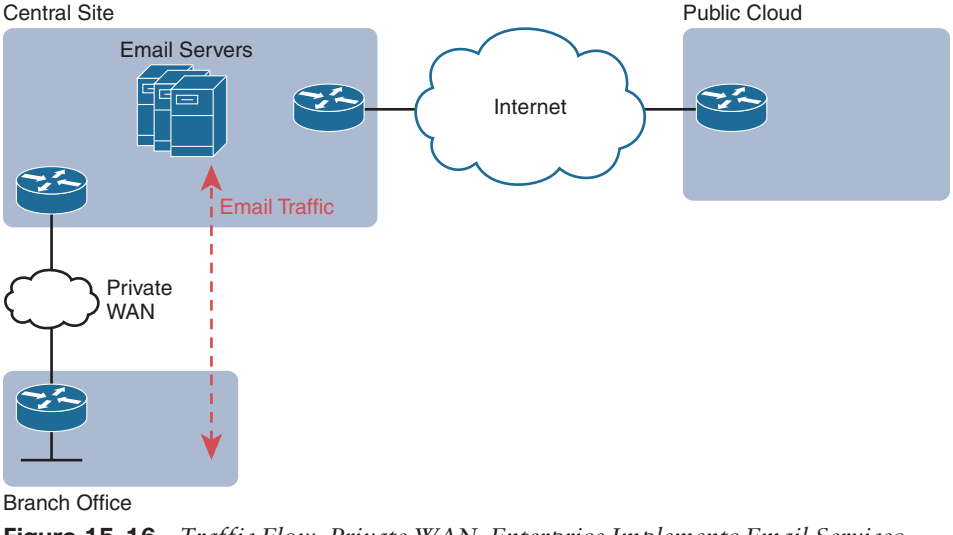


Figure 15-16 *Traffic Flow: Private WAN, Enterprise Implements Email Services*

The company then looks at the many different costs for email in this old model versus the new SaaS model. For instance, Microsoft Exchange is a very popular software package to build those enterprise email servers. Microsoft, a major player in the public cloud space with its Microsoft Azure service, offers Exchange as a SaaS service. (During the writing of this book, this particular service could be found as part of Office 365 or as "Exchange Online.") So the enterprise considers the options and chooses to migrate an email SaaS offering.

Once migrated, the email servers run in the cloud, but as a SaaS service. The enterprise IT staff, who are the customers of the SaaS service, do not have to manage the servers. Just to circle back to some big ideas, with a SaaS service, the consumer does not worry about installing VMs, sizing them, installing Exchange or some other email server software, and so on. The consumer receives email service in this case. The company does have to do some migration work to move existing email, contacts, and so on, but once completed, all users now communicate with email servers that run in the cloud as a SaaS service.

Now think about that enterprise branch office user, and the traffic flows shown in Figure 15-17, when a branch user sends or receives an email. For instance, think of an email with a large attachment, just to make the impact more dramatic. If the enterprise design connects branches to the central sites only, this is the net effect on WAN traffic:

- No reduction in private WAN traffic at all occurs because all the branch office email traffic flows to/from the central site.
- One hundred percent of the email traffic (even internal emails) that flows to/from branches now also flows over the Internet connection, consuming the bandwidth of the enterprise's Internet links.

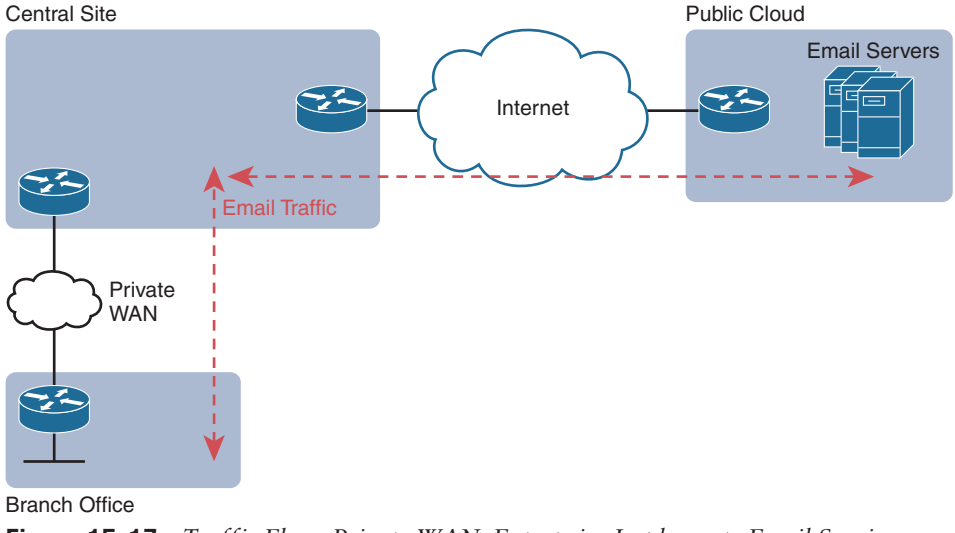


Figure 15-17 *Traffic Flow: Private WAN, Enterprise Implements Email Services*

Just to make the point, imagine two users at the same branch office. They can see each other across the room. One wants to share a file with the other, but the most convenient way they know to share a file is to email the file as an attachment. So one of them sends an email to the other, attaching the 20-MB file to the email. Before using SaaS, with an email

server at the central site, that email and file would flow over the private WAN, to the email server, and then back to the second user's email client. With this new design, that email with the 20-MB attachment would flow over the private WAN, then over the Internet to the email server, and then back again over the Internet and over the private WAN when the second user downloads her email.

Branch Offices with Internet and Private WAN

For enterprises that place their Internet connections primarily at the central sites, this public cloud model can cause problems like the one just described. One way to deal with this particular challenge is to plan the right capacity for the Internet links; another is to plan capacity for some private WAN connections to the public cloud. Another option exists as well: redesign the enterprise WAN to a small degree, and consider placing direct Internet connections at the branch offices. Then all Internet traffic, including the email traffic to the new SaaS service, could be sent directly, and not consume the private WAN bandwidth or the central site Internet link bandwidth, as shown in Figure 15-18.

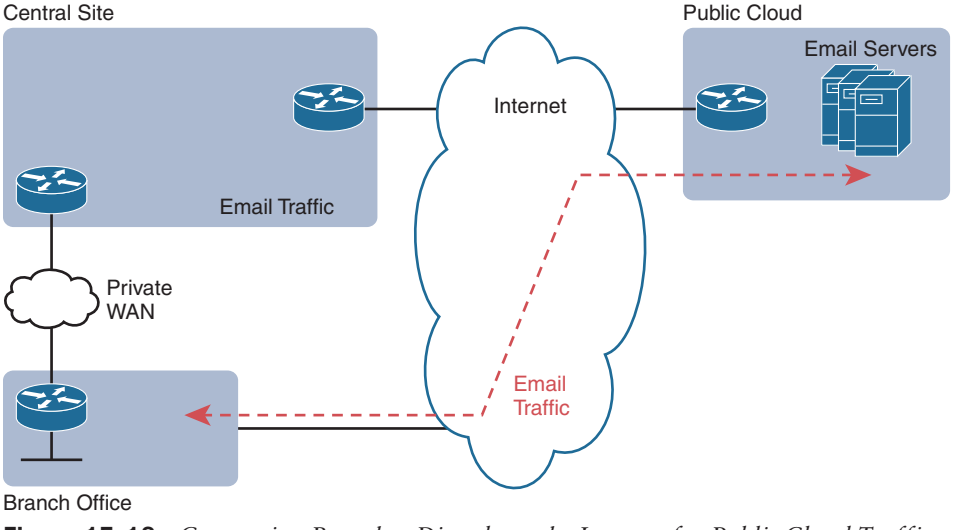


Figure 15-18 *Connecting Branches Directly to the Internet for Public Cloud Traffic*

The design in Figure 15-18 has several advantages. The traffic flows much more directly. It does not waste the WAN bandwidth for the central site. And broadband Internet connections are relatively inexpensive today compared to private WAN connections.

However, when the per-branch Internet connections are added for the first time, the new Internet links create security concerns. One of the reasons an enterprise might use only a few Internet links, located at a central site, is to focus the security efforts at those links. Using an Internet connection at each branch changes that approach. But many enterprises not only use the Internet at each site but also rely on it as their only WAN connection, as shown with Internet VPNs back in Chapter 14.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 15-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 15-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website

Review All the Key Topics

Key Topic

Table 15-4 Key Topics for Chapter 15

Key Topic Element	Description	Page Number
Figure 15-3	Organization of applications, on a VM, on an OS, with a hypervisor allocating and managing the host hardware	332
Figure 15-4	Virtual switch concept	333
List	Definition of cloud computing (paraphrased) based on the NIST standard	337
Figure 15-9	Organization and concepts for an IaaS service	340
Figure 15-11	Organization and concepts for a SaaS service	341
Figure 15-12	Organization and concepts for a PaaS service	342
List	Cons for using the Internet to access public WAN services	344
Table 15-2	Summary of pros and cons with different public cloud WAN access options	347

Key Terms You Should Know

Unified Computing System (UCS), virtual machine, virtual CPU (vCPU), hypervisor, Host (context: DC), virtual NIC (vNIC), virtual switch (vSwitch), on-demand self-service, resource pooling, rapid elasticity, cloud services catalog, public cloud, private cloud, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS)

This page intentionally left blank

Part IV Review

Keep track of your part review progress with the checklist shown in Table P4-1. Details on each task follow the table.

Table P4-1 Part IV Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		

Repeat All DIKTA Questions

For this task, use the PTP software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

This page intentionally left blank

Part V of this book includes most of the network automation topics from the CCNA blueprint; however, the part includes as much discussion of how Cisco and others have changed the way networks work to enable better automation as it discusses tools and processes to automate networks.

Chapters 16 and 17 examine a wide range of products and architectures that also enable better operations and automation. Chapter 16 discusses how controllers can separate out part of the work formerly done by networking devices. The chapter shows the advantages of these new controller-based models and details a few examples. Chapter 17 then goes on to give more detail about Cisco Software-Defined Access (SDA), a controller-based networking approach to building enterprise campus networks.

Chapters 18 and 19 discuss a few more specific details about network automation. Controllers typically include REST APIs and often return data to automation programs in the form of formatted data like JSON. Chapter 18 introduces these concepts. Chapter 19 then moves on to discuss IT automation tools, specifically Ansible, Puppet, and Chef, with focus on how to use these tools for network automation.

Part V

Network Automation

Chapter 16: Introduction to Controller-Based Networking

Chapter 17: Cisco Software-Defined Access (SDA)

Chapter 18: Understanding REST and JSON

Chapter 19: Understanding Ansible, Puppet, and Chef

Part V Review

CHAPTER 16

Introduction to Controller-Based Networking

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.f Endpoints

1.1.g Servers

1.2 Describe characteristics of network topology architectures

1.2.c Spine-leaf

6.0 Automation and Programmability

6.1 Explain how automation impacts network management

6.2 Compare traditional networks with controller-based networking

6.3 Describe controller-based and software defined architectures (overlay, underlay, and fabric)

6.3.a Separation of control plane and data plane

6.3.b Northbound and southbound APIs

The CCNA certification focuses on the traditional model for operating and controlling networks, a model that has existed for decades. You understand protocols that the devices use, along with the commands that can customize how those protocols operate. Then you plan and implement distributed configuration to the devices, device by device, to implement the network.

The 2010s have seen the introduction of a new network operational model: Software Defined Networking (SDN). SDN makes use of a controller that centralizes some network functions. The controller also creates many new capabilities to operate networks differently; in particular, controllers enable programs to automatically configure and operate networks through power application programming interfaces (APIs).

With traditional networking, the network engineer configured the various devices and changes requiring a long timeframe to plan and implement changes. With controller-based networking and SDN, network engineers and operators can implement changes more quickly, with better consistency, and often with better operational practices.

This chapter introduces the concepts of network programmability and SDN. Note that the topic area is large, with this chapter providing enough detail for you to understand the basics and to be ready for the other three chapters in this part.

The first major section of this chapter introduces the basic concepts of data and control planes, along with controllers and the related architecture. The second section then shows separate product examples of network programmability using controllers, all of which use different methods to implement networking features. The last section takes a little more exam-specific approach to these topics, comparing the benefits of traditional networking with the benefits of controller-based networking.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 16-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
SDN and Controller-Based Networks	1–3
Examples of Network Programmability and SDN	4–5
Comparing Traditional and Controller-Based Networks	6

1. A Layer 2 switch examines a frame's destination MAC address and chooses to forward that frame out port G0/1 only. That action occurs as part of which plane of the switch?
   - **a.** Data plane
   - **b.** Management plane
   - **c.** Control plane
   - **d.** Table plane
2. A router uses OSPF to learn routes and adds those to the IPv4 routing table. That action occurs as part of which plane of the switch?
   - **a.** Data plane
   - **b.** Management plane
   - **c.** Control plane
   - **d.** Table plane
3. A network uses an SDN architecture with switches and a centralized controller. Which of the following terms describes a function or functions expected to be found on the switches but not on the controller?
   - **a.** A northbound interface
   - **b.** A southbound interface
   - **c.** Data plane functions
   - **d.** Control plane functions

4. Which of the following controllers (if any) uses a mostly centralized control plane model?
   a. OpenDaylight Controller
   b. Cisco Application Policy Infrastructure Controller (APIC)
   c. Cisco APIC Enterprise Module (APIC-EM)
   d. None of these controllers uses a mostly centralized control plane.
5. To which types of nodes should an ACI leaf switch connect in a typical single-site design? (Choose two answers.)
   a. All of the other leaf switches
   b. A subset of the spine switches
   c. All of the spine switches
   d. Some of the endpoints
   e. None of the endpoints
6. Which answers list an advantage of controller-based networks versus traditional networks? (Choose two answers.)
   a. The ability to configure the features for the network rather than per device
   b. The ability to have forwarding tables at each device
   c. Programmatic APIs available per device
   d. More consistent device configuration

Foundation Topics

SDN and Controller-Based Networks

Networking devices forward data in the form of messages, typically data-link frames like Ethernet frames. You have learned about how switches and routers do that forwarding for the entire length of preparing for the CCNA exam.

Network programmability and Software Defined Networking (SDN) take those ideas, analyze the pieces, find ways to improve them for today's needs, and reassemble those ideas into a new way of making networks work. At the end of that rearrangement, the devices in the network still forward messages, but the how and why have changed.

This first major section explains the most central concepts of SDN and network programmability. It starts by breaking down some of the components of what exists in traditional networking devices. Then this section explains how some centralized controller software, called a controller, creates an architecture for easier programmatic control of a network.

The Data, Control, and Management Planes

Stop and think about what networking devices do. What does a router do? What does a switch do?

Many ideas should come to mind. For instance, routers and switches physically connect to each other with cables, and with wireless, to create networks. They forward messages: switches forward Ethernet frames, and routers forward packets. They use many different protocols to learn useful information such as routing protocols for learning network layer routes.

Everything that networking devices do can be categorized as being in a particular plane. This section takes those familiar facts about how networking devices work and describes the three planes most often used to describe how network programmability works: the data plane, the control plane, and the management plane.

The Data Plane

The term *data plane* refers to the tasks that a networking device does to forward a message. In other words, anything to do with receiving data, processing it, and forwarding that same data—whether you call the data a frame, a packet, or, more generically, a message—is part of the data plane.

For example, think about how routers forward IP packets, as shown in Figure 16-1. If you focus on the Layer 3 logic for a moment, the host sends the packet (step 1) to its default router, R1. R1 does some processing on the received packet, makes a forwarding (routing) decision, and forwards the packet (step 2). Routers R3 and R4 also receive, process, and forward the packet (steps 3 and 4).

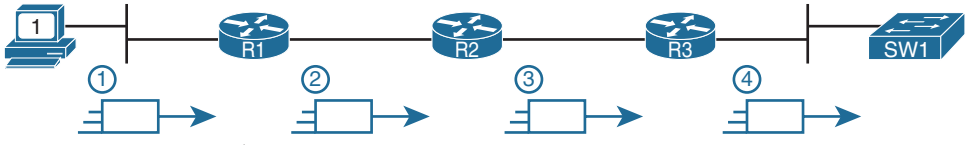


Figure 16-1 *Data Plane Processing on Routers: Basics*

Now broaden your thinking for a moment and try to think of everything a router or switch might do when receiving, processing, and forwarding a message. Of course, the forwarding decision is part of the logic; in fact, the data plane is often called the *forwarding plane*. But think beyond matching the destination address to a table. For perspective, the following list details some of the more common actions that a networking device does that fit into the data plane:

Key Topic

- De-encapsulating and re-encapsulating a packet in a data-link frame (routers, Layer 3 switches)
- Adding or removing an 802.1Q trunking header (routers and switches)
- Matching an Ethernet frame's destination Media Access Control (MAC) address to the MAC address table (Layer 2 switches)
- Matching an IP packet's destination IP address to the IP routing table (routers, Layer 3 switches)
- Encrypting the data and adding a new IP header (for virtual private network [VPN] processing)
- Changing the source or destination IP address (for Network Address Translation [NAT] processing)
- Discarding a message due to a filter (access control lists [ACLs], port security)

All the items in the list make up the data plane, because the data plane includes all actions done per message.

The Control Plane

Next, take a moment to ponder the kinds of information that the data plane needs to know beforehand so that it can work properly. For instance, routers need IP routes in a routing table before the data plane can forward packets. Layer 2 switches need entries in a MAC address table before they can forward Ethernet frames out the one best port to reach the destination. Switches must use Spanning Tree Protocol (STP) to limit which interfaces can be used for forwarding so that the data plane works well and does not loop frames forever.

From one perspective, the information supplied to the data plane controls what the data plane does. For instance, a router needs a route that matches a packet's destination address for the router to know how to route (forward) the packet. When a router's data plane tries to match the routing table and finds no matching route, the router discards the packet. And what controls the contents of the routing table? Various control plane processes.

The term *control plane* refers to any action that controls the data plane. Most of these actions have to do with creating the tables used by the data plane, tables like the IP routing table, an IP Address Resolution Protocol (ARP) table, a switch MAC address table, and so on. By adding to, removing, and changing entries to the tables used by the data plane, the control plane processes control what the data plane does. You already know about many control plane protocols—for instance, all the IP routing protocols.

Traditional networks use both a distributed data plane and a distributed control plane. In other words, each device has a data plane and a control plane, and the network distributes those functions into each individual device, as shown in the example in Figure 16-2.

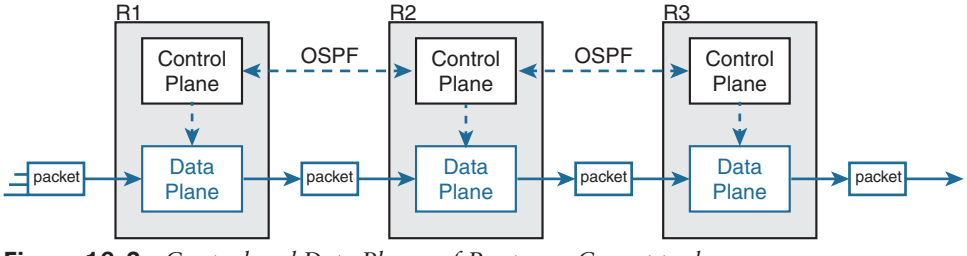


Figure 16-2 *Control and Data Planes of Routers—Conceptual*

In the figure, Open Shortest Path First (OSPF), the control plane protocol, runs on each router (that is, it is distributed among all the routers). OSPF on each router then adds to, removes from, and changes the IP routing table on each router. Once populated with useful routes, the data plane's IP routing table on each router can forward incoming packets, as shown from left to right across the bottom of the figure. The following list includes many of the more common control plane protocols:

Key Topic

- Routing protocols (OSPF, Enhanced Interior Gateway Routing Protocol [EIGRP], Routing Information Protocol [RIP], Border Gateway Protocol [BGP])
- IPv4 ARP
- IPv6 Neighbor Discovery Protocol (NDP)
- Switch MAC learning
- STP

Answers to the "Do I Know This Already?" quiz:

1 A **2** C **3** C **4** A **5** C, D **6** A, D

Without the protocols and activities of the control plane, the data plane of traditional networking devices would not function well. Routers would be mostly useless without routes learned by a routing protocol. Without learning MAC table entries, a switch could still forward unicasts by flooding them, but doing that for all frames would create much more load on the local-area network (LAN) compared to normal switch operations. So the data plane must rely on the control plane to provide useful information.

The Management Plane

The control plane performs overhead tasks that directly impact the behavior of the data plane. The *management plane* performs overhead work as well, but that work does not directly impact the data plane. Instead, the management plane includes protocols that allow network engineers to manage the devices.

Telnet and Secure Shell (SSH) are two of the most obvious management plane protocols. To emphasize the difference with control plane protocols, think about two routers: one configured to allow Telnet and SSH into the router and one that does not. Both could still be running a routing protocol and routing packets, whether or not they support Telnet and SSH.

Figure 16-3 extends the example shown in Figure 16-2 by now showing the management plane, with several management plane protocols.

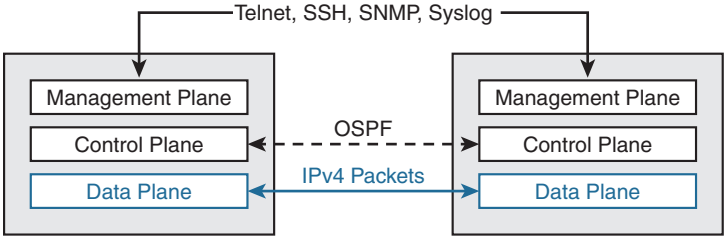


Figure 16-3 *Management Plane for Configuration of Control and Data Plane*

Cisco Switch Data Plane Internals

To better understand SDN and network programmability, it helps to think about the internals of switches. This next topic does just that.

From the very first days of devices called LAN switches, switches had to use specialized hardware to forward frames, because of the large number of frames per second (fps) required. To get a sense for the volume of frames a switch must be able to forward, consider the minimum frame size of an Ethernet frame, the number of ports on a switch, and the speeds of the ports; even low-end switches need to be able to forward millions of frames per second. For example, if a switch manufacturer wanted to figure out how fast its data plane needed to be in a new access layer switch with 24 ports, it might work through this bit of math:

- The switch has 24 ports.
- Each port runs at 1 Gbps.
- For this analysis, assume frames 125 bytes in length (to make the math easier, because each frame is 1000 bits long).
- With a 1000-bit-long frame and a speed of 1,000,000,000 bits/second, a port can send 1,000,000 frames per second (fps).

- Use full duplex on all ports, so the switch can expect to receive on all 24 ports at the same time.
- Result: Each port would be receiving 1,000,000 fps, for 24 million fps total, so the switch data plane would need to be ready to process 24 million fps.

Although 24 million fps may seem like a lot, the goal here is not to put an absolute number on how fast the data plane of a switch needs to be for any given era of switching technology. Instead, from their first introduction into the marketplace in the mid-1990s, LAN switches needed a faster data plane than a generalized CPU could process in software. As a result, hardware switches have always had specialized hardware to perform data plane processing.

First, the switching logic occurs not in the CPU with software, but in an *application-specific integrated circuit* (ASIC). An ASIC is a chip built for specific purposes, such as for message processing in a networking device.

Second, the ASIC needs to perform table lookup in the MAC address table, so for fast table lookup, the switch uses a specialized type of memory to store the equivalent of the MAC address table: *ternary content-addressable memory* (TCAM). TCAM memory does not require the ASIC to execute loops through an algorithm to search the table. Instead, the ASIC can feed the fields to be matched, like a MAC address value, into the TCAM, and the TCAM returns the matching table entry, without a need to run a search algorithm.

Note that a switch still has a general-purpose CPU and RAM as well, as shown in Figure 16-4. IOS runs in the CPU and uses RAM. Most of the control and management plane functions run in IOS. The data plane function (and the control plane function of MAC learning) happens in the ASIC.

Key Topic

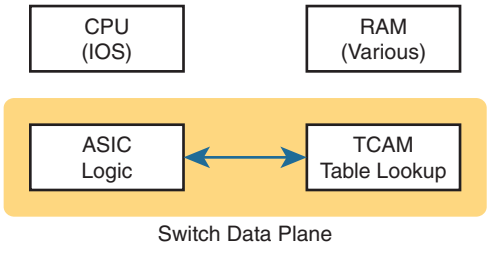


Figure 16-4 *Key Internal Processing Points in a Typical Switch*

Note that some routers also use hardware for data plane functions, for the same kinds of reasons that switches use hardware. (For instance, check out the Cisco Quantum Flow Processor for interesting reading about hardware data plane forwarding in Cisco routers.) The ideas of a hardware data plane in routers are similar to those in switches: use a purpose-built ASIC for the forwarding logic, and TCAM to store the required tables for fast table lookup.

Controllers and Software-Defined Architecture

New approaches to networking emerged in the 2010s, approaches that change where some of the control plane functions occur. Many of those approaches move parts of the control plane work into software that runs as a centralized application called a *controller*. This next topic looks at controller concepts, and the interfaces to the devices that sit below the controller and to any programs that use the controller.

NOTE The term *Software Defined Networking* (SDN) became common in the 2010s to refer to the types of controller-based networks described in the next few pages. More often today you might see terms like *software-defined architecture* or *controller-based networking*.

Controllers and Centralized Control

Most traditional control plane processes use a distributed architecture. For example, each router runs its own OSPF routing protocol process. To do their work, those distributed control plane processes use messages to communicate with each other, like OSPF protocol messages between routers. As a result, traditional networks are said to use a *distributed control plane*.

The people who created today's control plane concepts, like STP, OSPF, EIGRP, and so on, could have chosen to use a centralized control plane. That is, they could have put the logic in one place, running on one device, or on a server. Then the centralized software could have used protocol messages to learn information from the devices, but with all the processing of the information at a centralized location. But they instead chose a distributed architecture.

There are pros and cons to using distributed and centralized architectures to do any function in a network. Many control plane functions have a long history of working well with a distributed architecture. However, a centralized application can be easier to write than a distributed application, because the centralized application has all the data gathered into one place. And this emerging world of software-defined architectures often uses a centralized architecture, with a centralized control plane, with its foundations in a service called a controller.

A *controller*, or *SDN controller*, centralizes the control of the networking devices. The degree of control, and the type of control, varies widely. For instance, the controller can perform all control plane functions, replacing the devices' distributed control plane. Alternately, the controller can simply be aware of the ongoing work of the distributed data, control, and management planes on the devices, without changing how those operate. And the list goes on, with many variations.

To better understand the idea of a controller, consider one specific case as shown in Figure 16-5, in which one SDN controller centralizes all important control plane functions. First, the controller sits anywhere in the network that has IP reachability to the devices in the network. Each of the network devices still has a data plane; however, note that none of the devices has a control plane. In the variation of SDN as shown in Figure 16-5, the controller directly programs the data plane entries into each device's tables. The networking devices do not populate their forwarding tables with traditional distributed control plane processes.

NOTE Figure 16-5 shows the model used by one of the original SDN implementations that uses an industry standard called OpenFlow.

Figure 16-5 shows one model for network programmability and SDN, but not all. The figure does give us a great backdrop to discuss a few more important basic concepts; in particular, the idea of a southbound interface (SBI) and northbound interface (NBI).

Key Topic

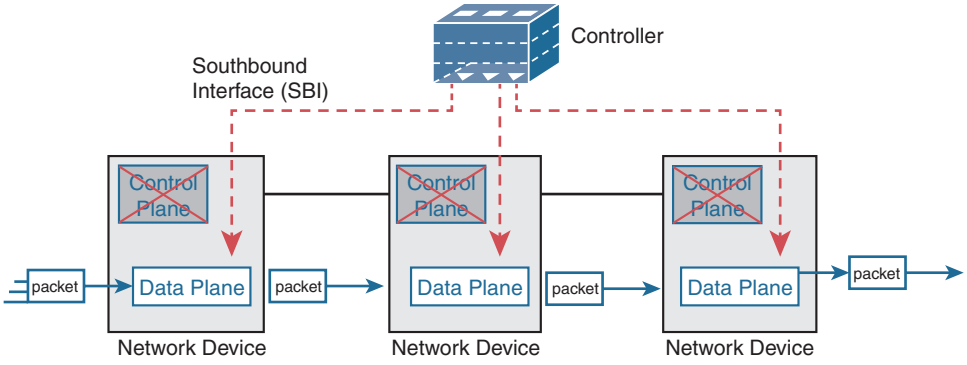


Figure 16-5 *Centralized Control Plane and a Distributed Data Plane*

The Southbound Interface

In a controller-based network architecture, the controller needs to communicate to the networking devices. In most network drawings and architecture drawings, those network devices typically sit below the controller, as shown in Figure 16-5. There is an interface between the controller and those devices, and given its location at the bottom part of drawings, the interface came to be known as the *southbound interface*, or SBI, as labeled in Figure 16-5.

NOTE In the context of this chapter's discussion of SDN, the word *interface* (including in the names of SBI, NBI, and API) refers to software interfaces unless otherwise noted.

Several different options exist for the SBI. The overall goal is network programmability, so the interface moves away from being only a protocol. An SBI often includes a protocol, so that the controller and devices can communicate, but it often includes an *application programming interface* (API). An API is a method for one application (program) to exchange data with another application. Rearranging the words to describe the idea, an API is an interface to an application program. Programs process data, so an API lets two programs exchange data. While a protocol exists as a document, often from a standards body, an API often exists as usable code—functions, variables, and data structures—that can be used by one program to communicate and copy structured data between the programs across a network.

So, back to the term *SBI*: it is an interface between a program (the controller) and a program (on the networking device) that lets the two programs communicate, with one goal being to allow the controller to program the data plane forwarding tables of the networking device.

Unsurprisingly, in a network architecture meant to enable network programmability, the capabilities of the SBIs and their APIs tell us a lot about what that particular architecture can and cannot do. For instance, some controllers might support one or a few SBIs, for a specific purpose, while others might support many more SBIs, allowing a choice of SBIs to use. The comparisons of SBIs go far beyond this chapter, but it does help to think about a few; the second major section gives three sample architectures that happen to show three separate SBIs, specifically:

- OpenFlow (from the ONF; www.opennetworking.org)
- OpFlex (from Cisco; used with ACI)

- CLI (Telnet/SSH) and SNMP (used with Cisco APIC-EM)
- CLI (Telnet/SSH) and SNMP, and NETCONF (used with Cisco Software-Defined Access)

The Northbound Interface

Think about the programming required at the controller related to the example in Figure 16-5. The figure focuses on the fact that the controller can add entries to the networking device's forwarding tables; however, how does the controller know what to add? How does it choose? What kind of information would your program need to gather before it could attempt to add something like MAC table entries or IP routes to a network? You might think of these:

- A list of all the devices in the network
- The capabilities of each devices
- The interfaces/ports on each device
- The current state of each port
- The topology—which devices connect to which, over which interfaces
- Device configuration—IP addresses, VLANs, and so on as configured on the devices

Key Topic

A controller does much of the work needed for the control plane in a centralized control model. It gathers all sorts of useful information about the network, like the items in the previous list. The controller itself can create a centralized repository of all this useful information about the network.

A controller's northbound interface (NBI) opens the controller so its data and functions can be used by other programs, enabling network programmability, with much quicker development. Programs can pull information from the controller, using the controller's APIs. The NBIs also enable programs to use the controller's capabilities to program flows into the devices using the controller's SBIs.

To see where the NBI resides, first think about the controller itself. The controller is software, running on some server, which can be a VM or a physical server. An application can run on the same server as the controller and use an NBI, which is an API, so that two programs can communicate.

Figure 16-6 shows just such an example. The big box in the figure represents the system where the controller software resides. This particular controller happens to be written in Java and has a Java-based native API. Anyone—the same vendor as the controller vendor, another company, or even you—can write an app that runs on this same operating system that uses the controller's Java API. By using that API to exchange data with the controller, the application can learn information about the network. The application can also program flows in the network—that is, ask the controller to add the specific match/action logic (flows) into the forwarding tables of the networking devices.

NOTE The northbound interface (NBI) gets its name from its normal location as shown above the controller—that is, in what would be north on a map.

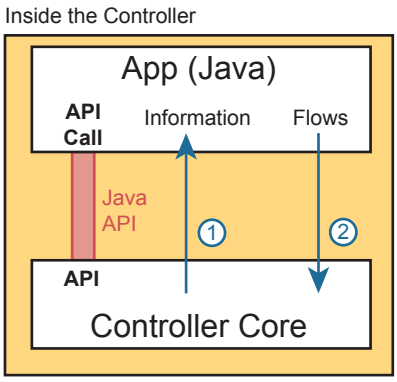


Figure 16-6 *Java API: Java Applications Communicates with Controller*

Before leaving the topic of NBIs, let me close with a brief explanation of a REST API as used for a controller. REST (*Representational State Transfer*) describes a type of API that allows applications to sit on different hosts, using HTTP messages to transfer data over the API. When you see SDN figures like Figure 16-6, with the application running on the same system as the controller, the API does not need to send messages over a network because both programs run on the same system. But when the application runs on a different system somewhere else in the network other than running on the controller, the API needs a way to send the data back and forth over an IP network, and RESTful APIs meet that need.

Figure 16-7 shows the big ideas with a REST API. The application runs on a host at the top of the figure. In this case, at step 1, it sends an HTTP GET request to a particular URI. The HTTP GET is like any other HTTP GET, even like those used to retrieve web pages. However, the URI is not for a web page, but rather identifies an object on the controller, typically a data structure that the application needs to learn and then process. For example, the URI might identify an object that is the list of physical interfaces on a specific device along with the status of each.

Key Topic

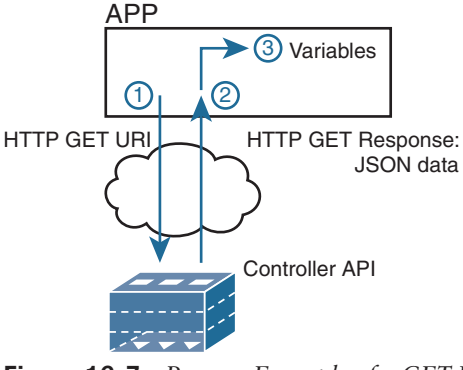


Figure 16-7 *Process Example of a GET Using a REST API*

At step 2, the controller sends back an HTTP GET response message with the object. Most REST APIs will ask for and receive structured data. That is, instead of receiving data that is a web page, like a web browser would receive, the response holds variable names and their values, in a format that can be easily used by a program. The common formats for data used

for network programmability are JavaScript Object Notation (JSON) and eXtensible Markup Language (XML), shown as step 3.

Software Defined Architecture Summary

SDN and network programmability introduce a new way to build networks. The networking devices still exist and still forward data, but the control plane functions and locations can change dramatically. The centralized controller acts as the focal point, so that at least some of the control plane functions move from a distributed model to a centralized model.

However, the world of network programmability and SDN includes a wide array of options and solutions. Some options pull most control plane functions into the controller, whereas others pull only some of those functions into the controller. The next section takes a look at three different options, each of which takes a different approach to network programmability and the degree of centralized control.

Examples of Network Programmability and SDN

This second of three major sections of the chapter introduces three different SDN and network programmability solutions available from Cisco. Others exist as well. These three were chosen because they give a wide range of comparison points:

- OpenDaylight Controller
- Cisco Application Centric Infrastructure (ACI)
- Cisco APIC Enterprise Module (APIC-EM)

OpenDaylight and OpenFlow

One common form of SDN comes from the Open Networking Foundation (ONF) and is billed as Open SDN. The ONF (www.opennetworking.org) acts as a consortium of users (operators) and vendors to help establish SDN in the marketplace. Part of that work defines protocols, SBIs, NBIs, and anything that helps people implement their vision of SDN.

The ONF model of SDN features OpenFlow. OpenFlow defines the concept of a controller along with an IP-based SBI between the controller and the network devices. Just as important, OpenFlow defines a standard idea of what a switch's capabilities are, based on the ASICs and TCAMs commonly used in switches today. (That standardized idea of what a switch does is called a *switch abstraction*.) An OpenFlow switch can act as a Layer 2 switch, a Layer 3 switch, or in different ways and with great flexibility beyond the traditional model of a Layer 2/3 switch.

The Open SDN model centralizes most control plane functions, with control of the network done by the controller plus any applications that use the controller's NBIs. In fact, earlier Figure 16-5, which showed the network devices without a control plane, represents this mostly centralized OpenFlow model of SDN.

In the OpenFlow model, applications may use any APIs (NBIs) supported on the controller platform to dictate what kinds of forwarding table entries are placed into the devices; however, it calls for OpenFlow as the SBI protocol. Additionally, the networking devices need to be switches that support OpenFlow.

Because the ONF's Open SDN model has this common thread of a controller with an OpenFlow SBI, the controller plays a big role in the network. The next few pages provide a brief background about two such controllers.

The OpenDaylight Controller

First, if you were to look back at the history of OpenFlow, you could find information on dozens of different SDN controllers that support the OpenFlow SDN model. Some were more research oriented, during the years in which SDN was being developed and was more of an experimental idea. As time passed, more and more vendors began building their own controllers. And those controllers often had many similar features, because they were trying to accomplish many of the same goals. As you might expect, some consolidation eventually needed to happen.

The OpenDaylight open-source SDN controller is one of the more successful SDN controller platforms to emerge from the consolidation process over the 2010s. OpenDaylight took many of the same open-source principles used with Linux, with the idea that if enough vendors worked together on a common open-source controller, then all would benefit. All those vendors could then use the open-source controller as the basis for their own products, with each vendor focusing on the product differentiation part of the effort, rather than the fundamental features. The result was that back in the mid-2010s, the *OpenDaylight SDN controller* (www.opendaylight.org) was born. OpenDaylight (ODL) began as a separate project but now exists as a project managed by the Linux Foundation.

Figure 16-8 shows a generalized version of the ODL architecture. In particular, note the variety of SBIs listed in the lower part of the controller box: OpenFlow, NetConf, PCEP, BGP-LS, and OVSDB; many more exist. The ODL project has enough participants so that it includes a large variety of options, including multiple SBIs, not just OpenFlow.

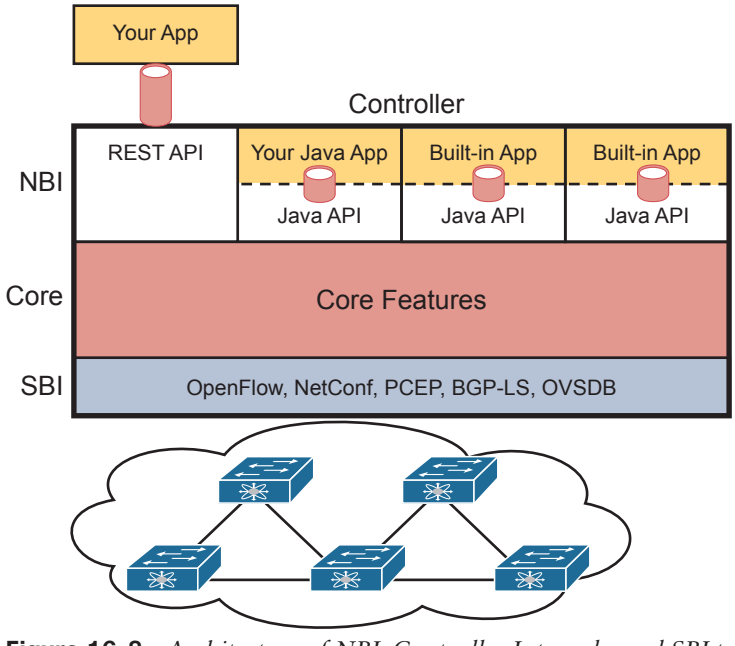


Figure 16-8 *Architecture of NBI, Controller Internals, and SBI to Network Devices*

ODL has many features, with many SBIs, and many core features. A vendor can then take ODL, use the parts that make sense for that vendor, add to it, and create a commercial ODL controller.

The Cisco Open SDN Controller (OSC)

At one point back in the 2010s, Cisco offered a commercial version of the OpenDaylight controller called the Cisco *Open SDN Controller* (OSC). That controller followed the intended model for the ODL project: Cisco and others contributed labor and money to the ODL open-source project; once a new release was completed, Cisco took that release and built new versions of their product.

Cisco no longer produces and sells the Cisco OSC, but I decided to keep a short section about OSC here in this chapter for a couple of reasons. First, if you do any of your own research, you will find mention of Cisco OSC; however, well before this chapter was written in 2019, Cisco had made a strong strategic move toward different approaches to SDN using intent-based networking (IBN). That move took Cisco away from OpenFlow-based SDN. But because you might see references to Cisco OSC online, or in the previous edition of this book, I wanted to point out this transition in Cisco's direction.

This book describes two Cisco offerings that use an IBN approach to SDN. The next topic in this chapter examines one of those: Application Centric Infrastructure (ACI), Cisco's data center SDN product. Chapter 17, "Cisco Software-Defined Access," discusses yet another Cisco SDN option that uses intent-based networking: Software-Defined Access (SDA).

Cisco Application Centric Infrastructure (ACI)

Interestingly, many SDN offerings began with research that discarded many of the old networking paradigms in an attempt to create something new and better. For instance, OpenFlow came to be from the Stanford University Clean Slate research project that had researchers reimagining (among other things) device architectures. Cisco took a similar research path, but Cisco's work happened to arise from different groups, each focused on different parts of the network: data center, campus, and WAN. That research resulted in Cisco's current SDN offerings of ACI in the data center, Software-Defined Access (SDA) in the enterprise campus, and Software-Defined WAN (SD-WAN) in the enterprise WAN.

When reimagining networking for the data center, the designers of SCI focused on the applications that run in a data center and what they need. As a result, they built networking concepts around application architectures. Cisco made the network infrastructure become application centric, hence the name of the Cisco data center SDN solution: *Application Centric Infrastructure*, or ACI.

For example, Cisco looked at the data center world beyond networking and saw lots of automation and control. As discussed in Chapter 15, "Cloud Architecture," virtualization software routinely starts, moves, and stops VMs. Additionally, cloud software enables self-service for customers so they can enable and disable highly elastic services as implemented with VMs and containers in a data center. From a networking perspective, some of those VMs need to communicate, but some do not. And those VMs can move based on the needs of the virtualization and cloud systems.

ACI set about to create data center networking with the flexibility and automation built into the operational model. Old data center networking models with a lot of per-physical-interface configuration on switches and routers were just poor models for the rapid pace of change and automated nature of modern data centers. This section looks at some of the detail of ACI to give you a sense of how ACI creates a powerful and flexible network to

support a modern data center in which VMs and containers are created, run, move, and are stopped dynamically as a matter of routine.

ACI Physical Design: Spine and Leaf

The Cisco ACI uses a specific physical switch topology called spine and leaf. While the other parts of a network might need to allow for many different physical topologies, the data center could be made standard and consistent. But what particular standard and consistent topology? Cisco decided on the spine and leaf design, also called a Clos network after one of its creators.

With ACI, the physical network has a number of spine switches and a number of leaf switches, as shown in Figure 16-9. The figure shows the links between switches, which can be single links or multiple parallel links. Of note in this design (assuming a single-site design):

Key Topic

- Each leaf switch must connect to every spine switch.
- Each spine switch must connect to every leaf switch.
- Leaf switches cannot connect to each other.
- Spine switches cannot connect to each other.
- Endpoints connect only to the leaf switches.

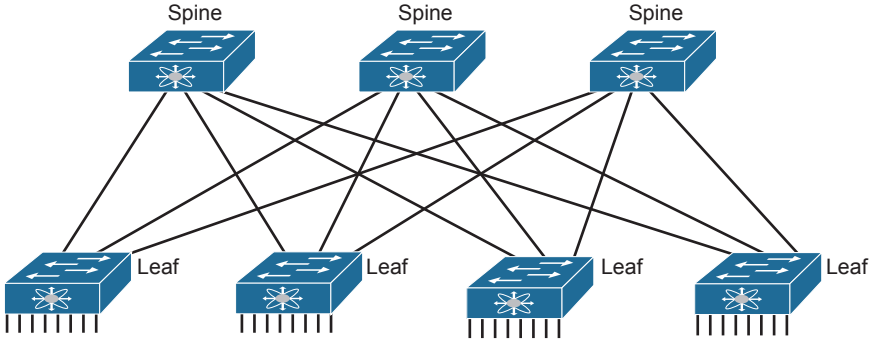


Figure 16-9 *Spine-Leaf Network Design*

Endpoints connect only to leaf switches and never to spine switches. To emphasize the point, Figure 16-10 shows a more detailed version of Figure 16-9, this time with endpoints connected to the leaf switches. None of the endpoints connect to the spine switches; they connect only to the leaf switches. The endpoints can be connections to devices outside the data center, like the router on the left. By volume, most of the endpoints will be either physical servers running a native OS or servers running virtualization software with numbers of VMs and containers as shown in the center of the figure.

Also, note that the figure shows a typical design with multiple leaf switches connecting to a single hardware endpoint like a Cisco Unified Computing System (UCS) server. Depending on the design requirements, each UCS might connect to at least two leaf switches, both for redundancy and for greater capacity to support the VMs and containers running on the UCS hardware. (In fact, in a small design with UCS or similar server hardware, every UCS might connect to every leaf switch.)

Key Topic

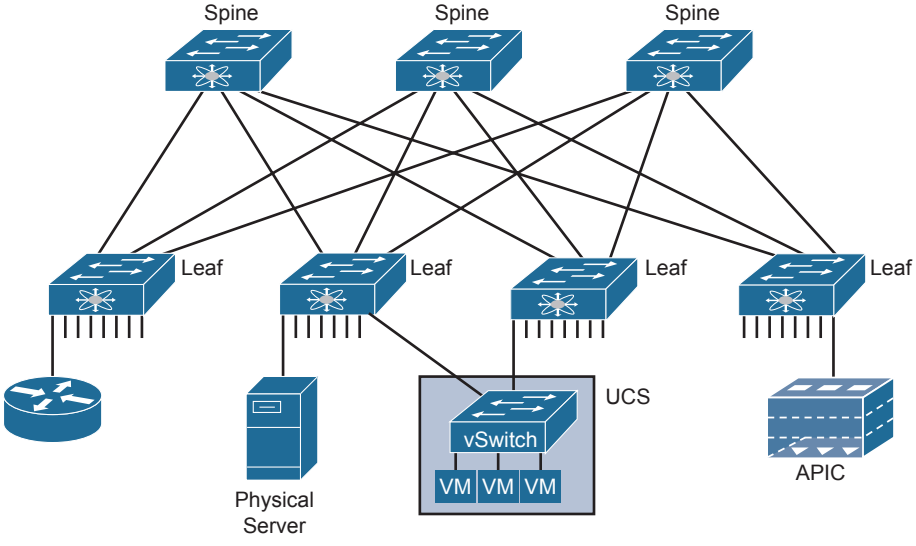


Figure 16-10 *Endpoints Found on the Leaf Switches Only*

ACI Operating Model with Intent-Based Networking

The model that Cisco defines for ACI uses a concept of endpoints and policies. The *endpoints* are the VMs, containers, or even traditional servers with the OS running directly on the hardware. ACI then uses several constructs as implemented via the Application Policy Infrastructure Controller (APIC), the software that serves as the centralized controller for ACI.

This section hopes to give you some insight into ACI, rather than touch on every feature. To do that, consider the application architecture of a typical enterprise web app for a moment. Most casual observers think of a web application as one entity, but one web app often exists as three separate servers:

- **Web server:** Users from outside the data center connect to a web server, which sends web page content to the user.
- **App (Application) server:** Because most web pages contain dynamic content, the app server does the processing to build the next web page for that particular user based on the user's profile and latest actions and input.
- **DB (Database) server:** Many of the app server's actions require data; the DB server retrieves and stores the data as requested by the app server.

To accommodate those ideas, ACI uses an intent-based networking (IBN) model. With that model, the engineer, or some automation program, defines the policies and intent for which endpoints should be allowed to communicate and which should not. Then the controller determines what that means for this network at this moment in time, depending on where the endpoints are right now.

For instance, when starting the VMs for this app, the virtualization software would create (via the APIC) several endpoint groups (EPGs) as shown in Figure 16-11. The controller must also be told the access policies, which define which EPGs should be able to communicate

(and which should not), as implied in the figure with arrowed lines. For example, the routers that connect to the network external to the data center should be able to send packets to all web servers, but not to the app servers or DB servers.

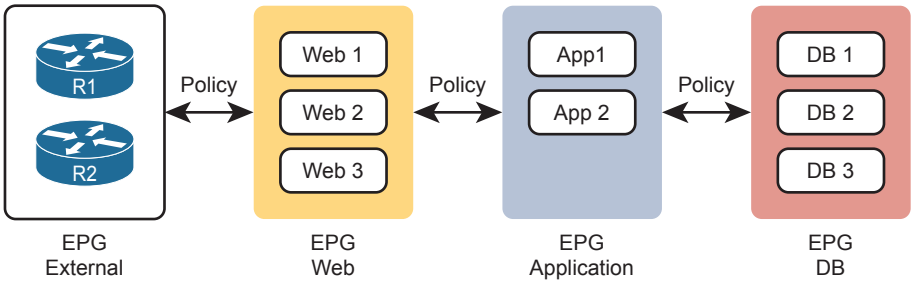


Figure 16-11 *Endpoint Groups (EPGs) and Policies*

Note that at no point did the previous paragraph talk about which physical switch interfaces should be assigned to which VLAN, or which ports are in an EtherChannel; the discussion moves to an application-centric view of what happens in the network. Once all the endpoints, policies, and related details are defined, the controller can then direct the network as to what needs to be in the forwarding tables to make it all happen—and to more easily react when the VMs start, stop, or move.

To make it all work, ACI uses a centralized controller called the *Application Policy Infrastructure Controller* (APIC), as shown in Figure 16-12. The name defines the function in this case: it is the controller that creates application policies for the data center infrastructure. The APIC takes the intent (EPGs, policies, and so on), which completely changes the operational model away from configuring VLANs, trunks, EtherChannels, ACLs, and so on.

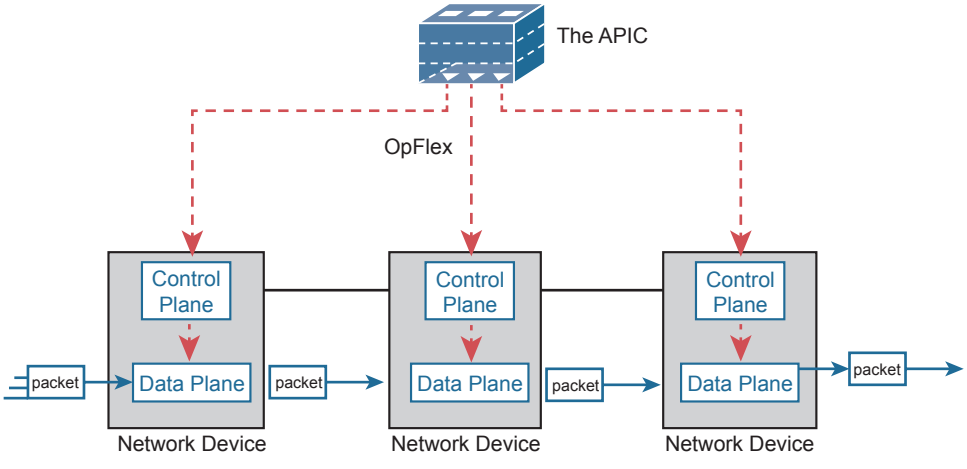


Figure 16-12 *Architectural View of ACI with APIC Pushing Intent to Switch Control Plane*

The APIC, of course, has a convenient GUI, but the power comes in software control—that is, network programmability. The same virtualization software, or cloud or automation software, even scripts written by the network engineer, can define the endpoint groups,

policies, and so on to the APIC. But all these players access the ACI system by interfacing to the APIC as depicted in Figure 16-13; the network engineer no longer needs to connect to each individual switch and configure CLI commands.

Key Topic

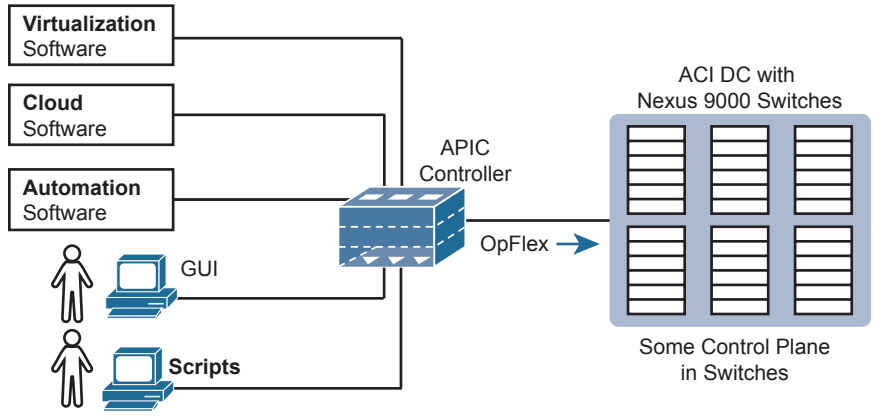


Figure 16-13 *Controlling the ACI Data Center Network Using the APIC*

For more information on Cisco ACI, go to www.cisco.com/go/aci.

Cisco APIC Enterprise Module

The next example of a Cisco SDN solution in this section, called *APIC Enterprise Module* (APIC-EM), solves a different problem. When Cisco began to reimagine networking in the enterprise, they saw a huge barrier: the installed base of their own products in most of their customer's networks. Any enterprise SDN solution that used new SBIs—SBIs that only some of the existing devices and software levels supported—would create a huge barrier to adoption.

APIC-EM Basics

Cisco came up with a couple of approaches, with one of those being APIC-EM, which Cisco released to the public around 2015.

APIC-EM assumes the use of the same traditional switches and routers with their familiar distributed data and control planes. Cisco rejected the idea that its initial enterprise-wide SDN (network programmability) solution could begin by requiring customers to replace all hardware. Instead, Cisco looked for ways to add the benefits of network programmability with a centralized controller while keeping the same traditional switches and routers in place. That approach could certainly change over time (and it has), but Cisco APIC-EM does just that: offer enterprise SDN using the same switches and routers already installed in networks.

NOTE Even though APIC-EM uses the same APIC acronym used for the controller with the Cisco ACI offering, the details of how it works differ significantly.

What advantages can a controller-based architecture bring if the devices in the network have no new features? In short, adding a centralized controller does nothing in comparison with old network management offerings. Adding a centralized controller with powerful

northbound APIs opens many possibilities for customers/operators, while also creating a world in which Cisco and its partners can bring new and interesting management applications to market. It includes these applications, as depicted in Figure 16-14:

- **Topology map:** The application discovers and displays the topology of the network.
- **Path Trace:** The user supplies a source and destination device, and the application shows the path through the network, along with details about the forwarding decision at each step.
- **Plug and Play:** This application provides Day 0 installation support so that you can unbox a new device and make it IP reachable through automation in the controller.
- **Easy QoS:** With a few simple decisions at the controller, you can configure complex QoS features at each device.

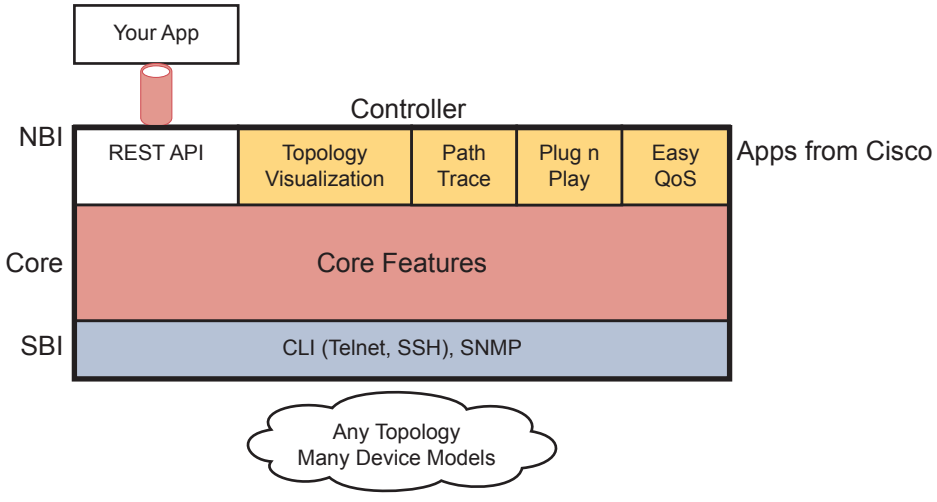


Figure 16-14 *APIC-EM Controller Model*

APIC-EM does not directly program the data or control planes, but it does interact with the management plane via Telnet, SSH, and/or SNMP; consequently, it can indirectly impact the data and control planes. The APIC-EM controller does not program flows into tables or ask the control plane in the devices to change how it operates. But it can interrogate and learn the configuration state and operational state of each device, and it can reconfigure each device, therefore changing how the distributed control and data plane operates.

APIC-EM Replacement

Cisco announced the current CCNA exam (200-301) in 2019, and around the same time Cisco announced the end of marketing for the APIC-EM product. That timing left us with a decision to make about whether to include APIC-EM in this book, and if so, to what extent. I decided to keep this small section about APIC-EM for several reasons, one reason being to give you these few closing comments about the product.

First, during the early 2020s—the years that CCNA 200-301 will likely still be the current exam—you will still see many references to APIC-EM. Cisco DevNet will likely still have many useful labs that reference and use APIC-EM, at least for a few years. Furthermore,

APIC-EM gives us a great tool to see how a controller can be used, even if the networking devices do not change their normal operation. So I think it's worth the few pages to introduces APIC-EM as done in this section.

Second, many of the functions of APIC-EM have become core features of the Cisco DNA Center (DNAC) product, which is discussed in some detail in Chapter 17. The list of applications just above this chapter's Figure 16-14 also exist as part of DNAC, for instance. So, do not look for APIC-EM version 2, but rather look for opportunities to use DNAC.

Summary of the SDN Examples

The three sample SDN architectures in this section of the book were chosen to provide a wide variety for the sake of learning. However, they differ to some degree in how much of the control plane work is centralized. Table 16-2 lists those and other comparison points taken from this section, for easy review and study.

Key Topic

Table 16-2 Points of Comparison: OpenFlow, ACI, and APIC Enterprise

Criteria	OpenFlow	ACI	APIC Enterprise
Changes how the device control plane works versus traditional networking	Yes	Yes	No
Creates a centralized point from which humans and automation control the network	Yes	Yes	Yes
Degree to which the architecture centralizes the control plane	Mostly	Partially	None
SBIs used	OpenFlow	OpFlex	CLI, SNMP
Controllers mentioned in this chapter	OpenDaylight	APIC	APIC-EM
Organization that is the primary definer/owner	ONF	Cisco	Cisco

If you want to learn more about the Cisco solutions, consider using both Cisco DevNet (the Cisco Developer Network) and dCloud (Demo cloud). Cisco provides its DevNet site (https://developer.cisco.com) for anyone interested in network programming, and the Demo Cloud site (https://dcloud.cisco.com) for anyone to experience or demo Cisco products. At the time this book went to press, DevNet had many APIC-EM labs, while both sites had a variety of ACI-based labs.

Comparing Traditional Versus Controller-Based Networks

Before finishing the chapter, this final topic turns directly toward the CCNA 200-301 exam. Three of the CCNA 200-301 exam topics in domain 6.0, "Automation and Programmability," ask us to compare some aspect of traditional networks versus new networking using controllers and automation. Those exam topics include

6.1: Explain how automation impacts network management

6.2: Compare traditional networks with controller-based networking

6.4: Compare traditional campus device management with Cisco DNA Center enabled device management

First, the wording in all three exam topics can be reduced to "compare and contrast." Two use the word *compare*. The other uses a longer phrase "explain how automation impacts…,"

which asks us to compare what was before to what happens now that automation has been added to the network.

Two exam topics (6.1 and 6.4) center on network management, so what might Cisco mean by "network management" in these exam topics? You could break that down into two aspects of network management: configuration management and operational management.

Configuration management refers to any feature that changes device configuration, with automated configuration management doing so with software (program) control. For instance, Cisco's ACI uses the APIC controller. You do not configure the devices directly, but the APIC pushes configuration down to the ACI switches that it builds based on its interpretation of the policies configured by the engineer. With ACI, the configuration management occurs as a part of the overall system. Other configuration management tools can be more focused on automating traditional configuration processes, with tools like NETCONF/RESTCONF, Ansible, Puppet, and Chef, as discussed in Chapter 18, "Understanding REST and JSON," and Chapter 19, "Understanding Ansible, Puppet, and Chef."

Operational network management includes monitoring, gathering operational data, reporting, and alerting humans to possible issues. For instance, the APIC-EM and DNA Center both have an app that checks the IOS images on Cisco devices to make sure only approved versions are used and that no changes have occurred to the images in comparison to the images created by Cisco.

The other exam topic (6.2) described in this section focuses on controller-based networking instead of network management. That exam topic includes any SDN network as characterized by the use of a controller. Today people might use that term or these other synonyms to describe some of the newer networking options that happen to use controllers:

- Software Defined Networking
- Software Defined Architecture
- Programmable Networks
- Controller-Based Networks

Table 16-3 summarizes the chapters that have content related to these three exam topics.

Table 16-3 Exam Topics and Most Relevant Chapters

Exam Topic	Exam Topic Text	Most Relevant Chapter(s)
6.1	Explain how automation impacts network management	16–19
6.2	Compare traditional networks with controller-based networking	16, 17
6.4	Compare traditional campus device management with Cisco DNA Center–enabled device management	17

How Automation Impacts Network Management

This chapter introduces many of the features that enable automation in SDNs, but so far it has not made any overt statements about how automation impacts network management. This next topic works through a couple of examples that show the power of automation as enabled through controller-based networks.

First, centralized controllers formalize and define data models for the configuration and operational data about networks. We humans might be comfortable with visually scanning the output of **show** commands to find the tidbit of information we need. Programs need to be able to identify the specific fact. To build a controller-based network with APIs, all the data about the network needs to be defined in a data model so programs can use that data via API calls. Before using controllers, automation scripts often had to begin by processing the text output of a **show** command, but with controllers and the data models behind the APIs, the data can be readily available to any automation script or vendor application through a northbound API.

For instance, Example 16-1 shows some output from a command on a switch. With a northbound API on a controller, and the data model it supplies, an automation program could issue this command and begin by parsing this text. The goal: find the configuration setting on the **switchport mode** command and the current trunking state.

Example 16-1 *Small Output from a Switch Command*

```
SW1# show interfaces gigabit 0/1 switchport
Name: Gi0/1
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
Negotiation of Trunking: On
```

Example 16-2 shows a simple example of the starting point for a program using a controller's northbound API. Instead of asking for the text from a **show** command, the API call will result in the program having a series of variables set. In this case, there are variables for that same interface that list the trunk configuration setting and the trunk operational state.

Example 16-2 *Python Dictionary with Variables Set to Needed Values*

```
>>> interface1
{'trunk-config': 'dynamic auto', 'trunk-status': 'static access'}
>>>
```

Using a controller-based model not only supplies APIs that give us the exact same data a human could see in **show** commands, but often they also supply much more useful information. A controller collects data from the entire network, so the controller can be written so that it analyzes and presents more useful data via the API. As a result, software that uses the APIs—whether automation written by local engineers or applications written by vendors—can be written more quickly and can often create features that would have been much more difficult without a controller.

For instance, both APIC-EM and its successor DNA Center provide a path trace feature. The applications show the path of a packet from source to destination, with the forwarding logic used at each node.

Now imagine writing that application with either of these two approaches.

- One API call that returns a list of all devices and their running configuration, with other API calls to collect each device's MAC address tables and/or their IP routing tables. Then you have to process that data to find the end-to-end path.
- One API call to which you pass the source and destination IP addresses and TCP/UDP ports, and the API returns variables that describe the end-to-end path, including device hostnames and interfaces. The variables spell out the path the packet takes through the network.

The second option does most of the work, while the first option leaves most of the work to you and your program. But that second option becomes possible because of the centralized controller. The controller has the data if it at least collects configuration and forwarding table information. Going beyond that, these Cisco controllers analyze the data to provide much more useful data. The power of these kinds of APIs is amazing, and this is just one example.

The following list summarizes a few of the comparison points for this particular exam topic:

- Northbound APIs and their underlying data models make it much easier to automate functions versus traditional networks.
- The robust data created by controllers makes it possible to automate functions that were not easily automated without controllers.
- The new reimagined software defined networks that use new operational models simplify operations, with automation resulting in more consistent configuration and less errors.
- Centralized collection of operational data at controllers allows the application of modern data analytics to networking operational data, providing actionable insights that were likely not noticeable with the former model.
- Time required to complete projects is reduced.
- New operational models use external inputs, like considering time-of-day, day-of-week, and network load.

Comparing Traditional Networks with Controller-Based Networks

As for exam topic 6.2, this entire chapter begins to show the advantages created by using controller-based networks. However, this chapter only begins to describe the possibilities. By centralizing some of the functions in the network and providing robust APIs, controllers enable a large number of new operational models. Those models include the three most likely to be seen from Cisco in an enterprise: Software-Defined Access (SDA), Software-Defined WAN (SD-WAN), and Application Centric Infrastructure (ACI). (Chapter 17 introduces SDA.)

This changes the operating paradigm in many cases, with the controller determining many device-specific details:

- The network engineer does not need to think about every command on every device.
- The controller configures the devices with consistent and streamlined settings.
- The result: faster and more consistent changes with fewer issues.

As another example, just consider the ACI example from earlier in the chapter. Instead of configuring each port with an access VLAN, or making it a trunk, adding routing protocol configuration, and possibly updating IP ACLs, all you had to do was create some end-point groups (EPGs) and policies. In that case, the orchestration software that started the VMs could automatically create the EPGs and policies. The new paradigm of intent-based networking was enabled through the controller-based architecture. Then the automation features enabled by the controller's northbound APIs allowed third-party applications to automatically configure the network to support the necessary changes.

Some of the advantages include the following:

Key Topic

- Uses new and improved operational models that allow the configuration of the network rather than per-device configuration
- Enables automation through northbound APIs that provide robust methods and model-driven data
- Configures the network devices through southbound APIs, resulting in more consistent device configuration, fewer errors, and less time spent troubleshooting the network
- Enables a DevOps approach to networks

Chapter 17 goes into some depth comparing traditional networking with controller-based networks with descriptions of Cisco Software-Defined Access (SDA). Look throughout that chapter for some of the reasons and motivations for SDA and the features enabled by using the DNA Center controller.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 16-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 16-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Book, website
Watch video		Website

Review All the Key Topics

Key Topic

Table 16-5 Key Topics for Chapter 16

Key Topic Element	Description	Page Number
List	Sample actions of the networking device data plane	359
List	Sample actions of the networking device control plane	360
Figure 16-4	Switch internals with ASIC and TCAM	362
Figure 16-5	Basic SDN architecture, with the centralized controller programming device data planes directly	364
Paragraph	Description of the role and purpose of the NBI	365
Figure 16-7	REST API basic concepts	366
List	Spine-leaf topology requirements	370
Figure 16-10	Spine-leaf design	371
Figure 16-13	Controlling the ACI data center network using APIC	373
Table 16-2	Comparisons of Open SDN, Cisco ACI, and Cisco APIC Enterprise options	375
List	Comparisons of how automation improves network management	378
List	Comparisons of how controller-based networking works versus traditional networking	379

Key Terms You Should Know

application programming interface (API), Application Policy Infrastructure Controller (APIC), APIC Enterprise Module (APIC-EM), Application Centric Infrastructure (ACI), northbound API, southbound API, control plane, data plane, management plane, application-specific integrated circuit (ASIC), ternary content-addressable memory (TCAM), OpenFlow, Software Defined Networking (SDN), distributed control plane, centralized control plane, northbound interface (NBI), southbound interface (SBI), controller-based networking, intent-based networking (IBN), spine, leaf

This page intentionally left blank

CHAPTER 17

Cisco Software-Defined Access (SDA)

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.1 Explain the role and function of network components

1.1.e Controllers (Cisco DNA Center and WLC)

6.0 Automation and Programmability

6.1 Explain how automation impacts network management

6.2 Compare traditional networks with controller-based networking

6.3 Describe controller-based and software defined architectures (overlay, underlay, and fabric)

6.3.a Separation of control plane and data plane

6.3.b Northbound and southbound APIs

6.4 Compare traditional campus device management with Cisco DNA Center enabled device management

Cisco Software-Defined Access (SDA) uses a software defined networking approach to build a converged wired and wireless campus LAN. The word *access* in the name refers to the endpoint devices that access the network, while *software-defined* refers to many of the usual software-defined architectural features discussed in Chapter 16, "Introduction to Controller-Based Networking." Those features include a centralized controller—DNA Center—with southbound and northbound protocols. It also includes a completely different operational model inside SDA, with a network fabric composed of an underlay network and an overlay network.

SDA fills the position as Cisco's campus offering within Cisco Digital Network Architecture (DNA). Cisco DNA defines the entire architecture for the new world of software defined networks, digitization, and Cisco's reimagining of how networks should be operated in the future. This chapter introduces SDA, which exists as one implementation of Cisco DNA.

The discussion of SDA and DNA provides a great backdrop to discuss a few other topics from the CCNA blueprint: the DNA Center controller and network management. SDA uses the DNA Center controller to configure and operate SDA. However, DNA Center also acts as a complete network management platform. To understand DNA Center, you also need to understand traditional network management as well as the new management models using controllers.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 17-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
SDA Fabric, Underlay, and Overlay	1–3
DNA Center and SDA Operation	4, 5
DNA Center as a Network Management Platform	6

1. In Cisco Software-Defined Access (SDA), which term refers to the devices and cabling, along with configuration that allows the network device nodes enough IP connectivity to send IP packets to each other?
   - **a.** Fabric
   - **b.** Overlay
   - **c.** Underlay
   - **d.** VXLAN
2. In Cisco Software-Defined Access (SDA), which term refers to the functions that deliver endpoint packets across the network using tunnels between the ingress and egress fabric nodes?
   - **a.** Fabric
   - **b.** Overlay
   - **c.** Underlay
   - **d.** VXLAN
3. In Software-Defined Access (SDA), which of the answers are part of the overlay data plane?
   - **a.** LISP
   - **b.** GRE
   - **c.** OSPF
   - **d.** VXLAN
4. Which answers best describe options of how to implement security with scalable groups using DNA Center and SDA? (Choose two answers.)
   - **a.** A human user from the DNA Center GUI
   - **b.** An automation application using NETCONF
   - **c.** A human user using the CLI of an SDA fabric edge node
   - **d.** An automation application using REST

5. Which of the following protocols or tools could be used as part of the Cisco DNA Center southbound interface? (Choose three answers.)
   a. Ansible
   b. SSH
   c. NETCONF
   d. SNMP
   e. Puppet
6. Which of the following are network management features performed by both traditional network management software as well as by DNA Center? (Choose two answers.)
   a. Network device discovery
   b. Software-Defined Access configuration
   c. End-to-end path discovery with ACL analysis
   d. Device installation (day 0), configuration (day 1), and monitoring (day n) operations

Foundation Topics

SDA Fabric, Underlay, and Overlay

Cisco Software-Defined Access (SDA) creates an entirely new way to build campus LANs as compared with the traditional methods of networking discussed in most chapters of this book. In the mid 2010s, Cisco set about to reimagine campus networking, with SDA as the result.

SDA uses the software-defined architectural model introduced in Chapter 16, with a controller and various APIs. It still uses a physical network with switches and routers, cables, and various endpoints. At the center sits the Digital Network Architecture (DNA) Center controller, as shown in Figure 17-1, with human users making use of a graphical user interface (GUI) and automation using APIs. In short, DNA Center is the controller for SDA networks.

Architecturally, the southbound side of the controller contains the fabric, underlay, and overlay. By design in SDN implementations, most of the interesting new capabilities occur on the northbound side, which are examined in the second half of this chapter. This first half of the chapter examines the details south of the controller—namely, the fabric, underlay network, and overlay network.

Key Topic

Overlay: The mechanisms to create VXLAN tunnels between SDA switches, which are then used to transport traffic from one fabric endpoint to another over the fabric.

Underlay: The network of devices and connections (cables and wireless) to provide IP connectivity to all nodes in the fabric, with a goal to support the dynamic discovery of all SDA devices and endpoints as a part of the process to create overlay VXLAN tunnels.

Fabric: The combination of overlay and underlay, which together provide all features to deliver data across the network with the desired features and attributes.

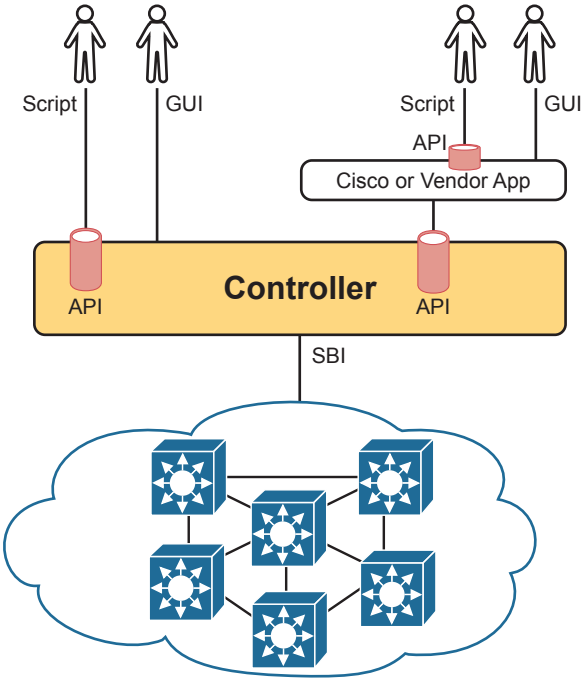


Figure 17-1 *SDA Architectural Model with DNA Center*

In less formal terms, the underlay exists as multilayer switches and their links, with IP connectivity—but for a special purpose. The underlay supports some new concepts with a tunneling method called VXLAN. Traffic sent by the endpoint devices flows through VXLAN tunnels in the overlay—a completely different process than traditional LAN switching and IP routing.

For instance, think about the idea of sending packets from hosts on the left of a network, over SDA, to hosts on the right. For instance, imagine a packet enters on the left side of the physical network at the bottom of Figure 17-2 and eventually exits the campus out switch SW2 on the far right. This underlay network looks like a more traditional network drawing, with several devices and links.

The overlay drawing at the top of the figure shows only two switches—called fabric edge nodes, because they happen to be at the edges of the SDA fabric—with a tunnel labeled VXLAN connecting the two. Both concepts (underlay and overlay) together create the SDA fabric.

The next few pages explain both the underlay and overlay in a little more depth.

Key Topic

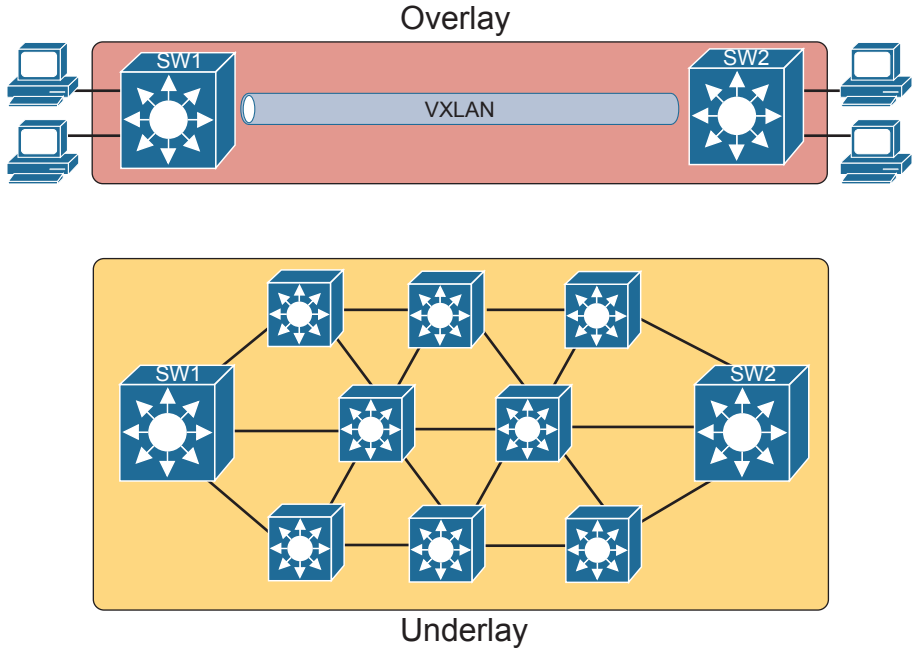


Figure 17-2 *Fabric, Underlay, and Overlay Concepts*

The SDA Underlay

With SDA, the underlay exists to provide connectivity between the nodes in the SDA environment for the purpose of supporting VXLAN tunnels in the overlay network. To do that, the underlay includes the switches, routers, cables, and wireless links used to create the physical network. It also includes the configuration and operation of the underlay so it can support the work of the overlay network.

Using Existing Gear for the SDA Underlay

To build an SDA underlay network, companies have two basic choices. They can use their existing campus network and add new configuration to create an underlay network, while still supporting their existing production traffic with traditional routing and switching. Alternately, the company can purchase some new switches and build the SDA network without concern for harming existing traffic, and migrate endpoints to the new SDA network over time.

To build SDA into an existing network, it helps to think for a moment about some typical campus network designs. The larger campus site may use either a two-tier or three-tier design as discussed in Chapter 13, "LAN Architecture." It has a cluster of wireless LAN controllers (WLCs) to support a number of lightweight APs (LWAPs). Engineers have configured VLANs, VLAN trunks, IP routing, IP routing protocols, ACLs, and so on. And the LAN connects to WAN routers.

Answers to the "Do I Know This Already?" quiz:

1 C **2** B **3** D **4** A, D **5** B, C, D **6** A, D

SDA can be added into an existing campus LAN, but doing so has some risks and restrictions. First and foremost, you have to be careful not to disrupt the current network while adding the new SDA features to the network. The issues include

- Because of the possibility of harming the existing production configuration, DNA Center should not be used to configure the underlay if the devices are currently used in production. (DNA Center will be used to configure the underlay with deployments that use all new hardware.)
- The existing hardware must be from the SDA compatibility list, with different models supported depending on their different SDA roles (see a link at www.cisco.com/go/sda).
- The device software levels must meet the requirements, based on their roles, as detailed in that same compatibility list.

For instance, imagine an enterprise happened to have an existing campus network that uses SDA-compatible hardware. That company might need to update the IOS versions in a few cases. Additionally, the engineers would need to configure the underlay part of the SDA devices manually rather than with DNA Center because Cisco assumes that the existing network already supports production traffic, so they want the customer directly involved in making those changes.

The SDA underlay configuration requires you to think about and choose the different SDA roles filled by each device before you can decide which devices to use and which minimum software levels each requires. If you look for the hardware compatibility list linked from www.cisco.com/go/sda, you will see different lists of supported hardware and software depending on the roles. These roles include

Key Topic

Fabric edge node: A switch that connects to endpoint devices (similar to traditional access switches)

Fabric border node: A switch that connects to devices outside SDA's control, for example, switches that connect to the WAN routers or to an ACI data center

Fabric control node: A switch that performs special control plane functions for the underlay (LISP), requiring more CPU and memory

For example, when I was writing this chapter back in 2019, Cisco's compatibility list included many Catalyst 9300, 9400, and 9500 switches, but also some smaller Catalyst 3850 and 3650 switches, as fabric edge nodes. However, the Catalyst 2960X or 2960XR products did not make the list as fabric edge nodes. For fabric control nodes, the list included more higher-end Catalyst switch models (which typically have more CPU and RAM), plus several router models (routers typically have much more RAM for control plane protocol storage—for instance, for routing protocols).

The beginning of an SDA project will require you to look at the existing hardware and software to begin to decide whether the existing campus might be a good candidate to build the fabric with existing gear or to upgrade hardware when building the new campus LAN.

Using New Gear for the SDA Underlay

When buying new hardware for the SDA fabric—that is, a greenfield design—you remove many of the challenges that exist when deploying SDA on existing gear. You can simply order compatible hardware and software. Once it arrives, DNA Center can then configure all the underlay features automatically.

At the same time, the usual campus LAN design decisions still need to be made. Enterprises use SDA as a better way to build and operate a campus network, but SDA is still a campus network. It needs to provide access and connectivity to all types of user devices. When planning a greenfield SDA design, plan to use SDA-compatible hardware, but also think about these traditional LAN design points:

- The number of ports needed in switches in each wiring closet
- The port speeds required
- The benefit of a switch stack in each wiring closet
- The cable length and types of cabling already installed
- The need for power (PoE/PoE+)
- The power available in each new switch versus the PoE power requirements
- Link capacity (speed and number of links) for links between switches

As far as the topology, traditional campus design does tell us how to connect devices, but SDA does not have to follow those traditional rules. To review, traditional campus LAN Layer 2 design (as discussed back in Chapter 13) tells us to connect each access switch to two different distribution layer switches, but not to other access layer switches, as shown in Figure 17-3. The access layer switch acts as a Layer 2 switch, with a VLAN limited to those three switches.

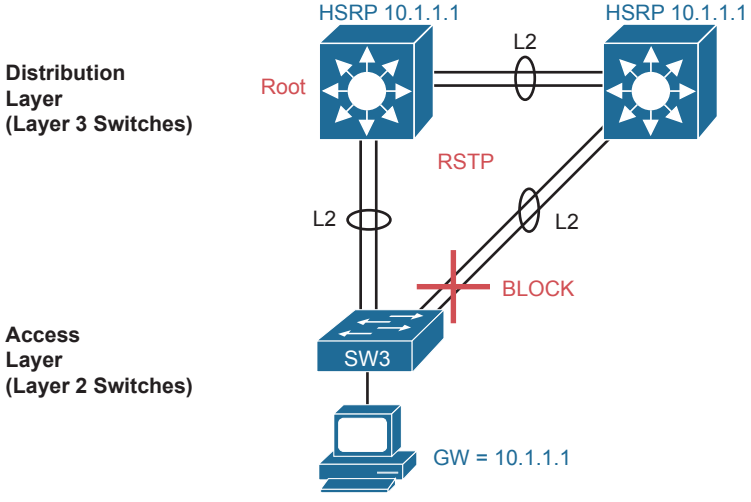


Figure 17-3 *Traditional Access Layer Design: Three Switches in STP Triangle*

Take a moment to reflect about the traditional features shown in the figure. The distribution layer switches—Layer 3 switches—act as the default gateway used by hosts and often implement HSRP for better availability. The design uses more than one uplink from the access to distribution layer switches, with Layer 2 EtherChannels, to allow balancing in addition to redundancy. And STP/RSTP manages the small amount of Layer 2 redundancy in the campus, preventing loops by blocking on some ports.

In comparison, a greenfield SDA fabric uses a *routed access layer* design. Routed access layer designs have been around long before SDA, but SDA makes good use of the design,

and it works very well for the underlay with its goal to support VXLAN tunnels in the overlay network. A routed access layer design simply means that all the LAN switches are Layer 3 switches, with routing enabled, so all the links between switches operate as Layer 3 links.

With a greenfield SDA deployment—that is, all new gear that you can allow to be configured by DNA Center—DNA Center will configure the devices' underlay configuration to use a *routed access layer*. Because DNA Center knows it can configure the switches without concern of harming a production network, it chooses the best underlay configuration to support SDA. That best configuration happens to use a design called a routed access layer design, which has these features:

Key Topic

- All switches act as Layer 3 switches.
- The switches use the IS-IS routing protocol.
- All links between switches (single links, EtherChannels) are routed Layer 3 links (not Layer 2 links).
- As a result, STP/RSTP is not needed, with the routing protocol instead choosing which links to use based on the IP routing tables.
- The equivalent of a traditional access layer switch—an SDA edge node—acts as the default gateway for the endpoint devices, rather than distribution switches.
- As a result, HSRP (or any FHRP) is no longer needed.

Figure 17-4 repeats the same physical design as in Figure 17-3 but shows the different features with the routed access design as configured using DNA Center.

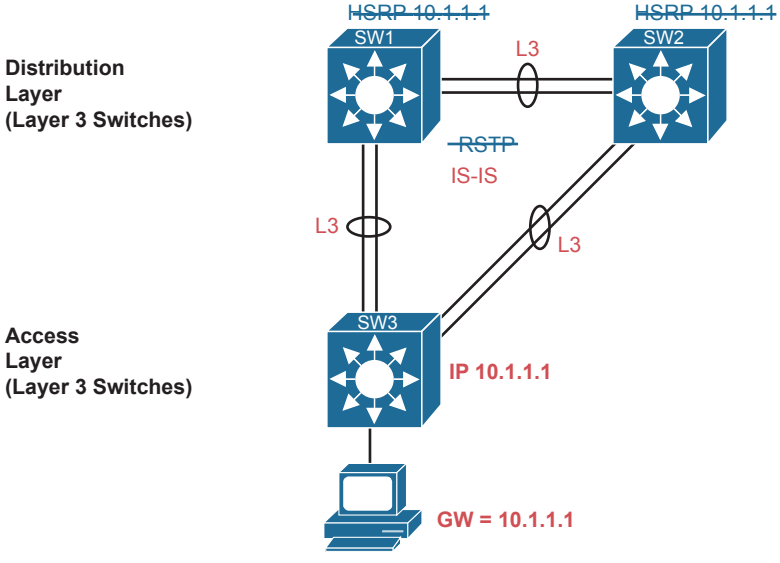


Figure 17-4 *SDA Fabric Layer 3 Access Benefits*

NOTE DNA Center configures the underlay with consistent settings for each instance of DNA across an enterprise. This convention simplifies operation as an enterprise completes a migration to SDA.

The SDA Overlay

When you first think of the SDA overlay, think of this kind of sequence. First, an endpoint sends a frame that will be delivered across the SDA network. The first SDA node to receive the frame encapsulates the frame in a new message—using a tunneling specification called VXLAN—and forwards the frame into the fabric. Once the ingress node has encapsulated the original frame in VXLAN, the other SDA nodes forward the frame based on the VXLAN tunnel details. The last SDA node removes the VXLAN details, leaving the original frame, and forwards the original frame on toward the destination endpoint.

While the summary of some of SDA's overlay work in the previous paragraph may sound like a lot of work, all that work happens in each switch's ASIC. So, while it is more complex to understand, there is no performance penalty for the switches to perform the extra work.

When Cisco set about to create SDA, they saw an opportunity. Making use of VXLAN tunnels opened up the possibilities for a number of new networking features that did not exist without VXLAN. This next topic begins with a closer look at the VXLAN tunnels in the overlay, followed by a discussion of how SDA uses LISP for endpoint discovery and location needed to create the VXLAN tunnels.

VXLAN Tunnels in the Overlay (Data Plane)

SDA has many additional needs beyond the simple message delivery—needs that let it provide improved functions. To that end, SDA does not only route IP packets or switch Ethernet frames. Instead, it encapsulates incoming data link frames in a tunneling technology for delivery across the SDA network, with these goals in mind:

- The VXLAN tunneling (the encapsulation and de-encapsulation) must be performed by the ASIC on each switch so that there is no performance penalty. (That is one reason for the SDA hardware compatibility list: the switches must have ASICs that can perform the work.)
- The VXLAN encapsulation must supply header fields that SDA needs for its features, so the tunneling protocol should be flexible and extensible, while still being supported by the switch ASICs.
- The tunneling encapsulation needs to encapsulate the entire data link frame instead of encapsulating the IP packet. That allows SDA to support Layer 2 forwarding features as well as Layer 3 forwarding features.

To achieve those goals, when creating SDA, Cisco chose the *Virtual Extensible LAN* (VXLAN) protocol to create the tunnels used by SDA. When an SDA endpoint (for example, an end-user computer) sends a data link frame into an SDA edge node, the ingress edge node encapsulates the frame and sends it across a VXLAN tunnel to the egress edge node, as shown in Figure 17-5.

To support the VXLAN encapsulation, the underlay uses a separate IP address space as compared with the rest of the enterprise, including the endpoint devices that send data over the SDA network. The overlay tunnels use addresses from the enterprise address space. For instance, imagine an enterprise used these address spaces:

- 10.0.0.0/8: Entire enterprise
- 172.16.0.0/16: SDA underlay

Key Topic

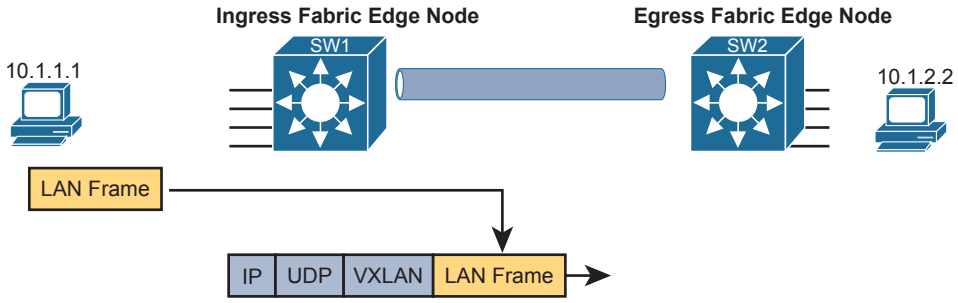


Figure 17-5 *Fundamentals of VXLAN Encapsulation in SDA*

To make that work, first the underlay would be built using the 172.16.0.0/16 IPv4 address space, with all links using addresses from that address space. As an example, Figure 17-6 shows a small SDA design, with four switches, each with one underlay IP address shown (from the 172.16.0.0/16 address space).

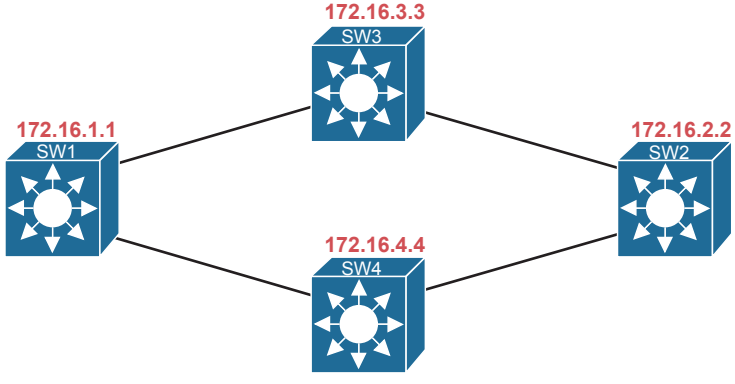


Figure 17-6 *SDA Underlay Using 172.16.0.0*

The overlay tunnel creates a path between two fabric edge nodes in the overlay IP address space—that is, in the same address space used by all the endpoints in the enterprise. Figure 17-7 emphasizes that point by showing the endpoints (PCs) on the left and right, with IP addresses in network 10.0.0.0/8, with the VXLAN overlay tunnel shown with addresses also from 10.0.0.0/8.

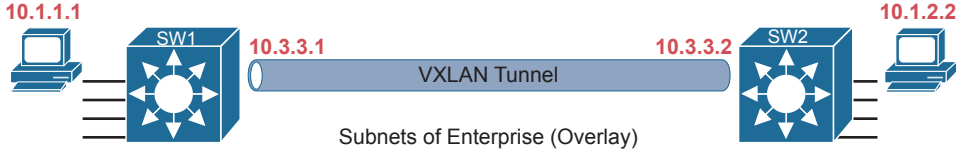


Figure 17-7 *VXLAN Tunnel and Endpoints with IPv4 Addresses in the Same IPv4 Space*

LISP for Overlay Discovery and Location (Control Plane)

Ignore SDA for a moment, and think about traditional Layer 2 switching and Layer 3 routing. How do their control planes work? In other words, how do these devices discover the possible destinations in the network, store those destinations, so that the data plane has all the data it needs when making a forwarding decision? To summarize:

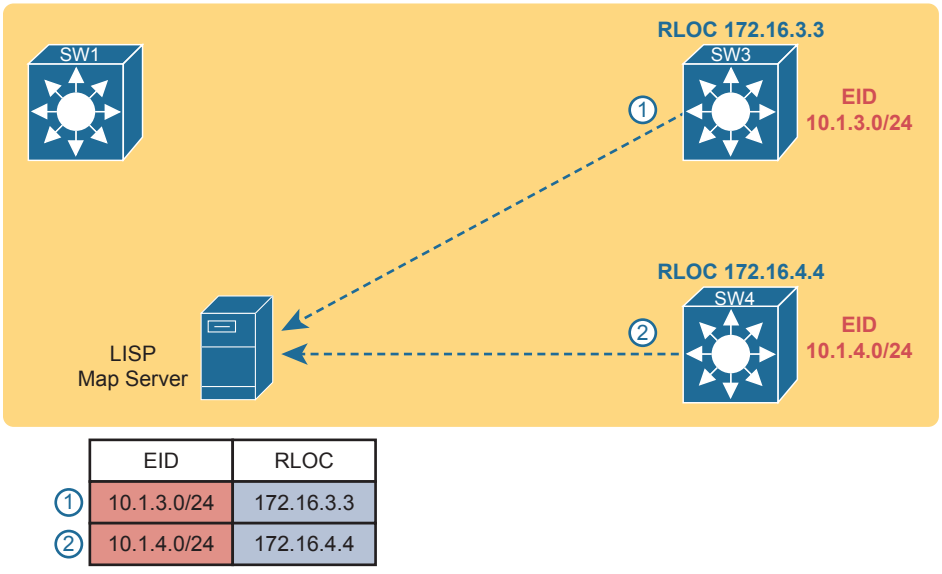
- Traditional Layer 2 switches learn possible destinations by examining the source MAC addresses of incoming frames, storing those MAC addresses as possible future destinations in the switch's MAC address table. When new frames arrive, the Layer 2 switch data plane then attempts to match the Ethernet frame's destination MAC address to an entry in its MAC address table.
- Traditional Layer 3 routers learn destination IP subnets using routing protocols, storing routes to reach each subnet in their routing tables. When new packets arrive, the Layer 3 data plane attempts to match the IP packet's destination IP address to some entry in the IP routing table.

Nodes in the SDA network do not do these same control plane actions to support endpoint traffic. Just to provide a glimpse into the process for the purposes of CCNA, consider this sequence, which describes one scenario:

- Fabric edge nodes—SDA nodes that connect to the edge of the SDA fabric—learn the location of possible endpoints using traditional means, based on their MAC address, individual IP address, and by subnet, identifying each endpoint with an endpoint identifier (EID).
- The fabric edge nodes register the fact that the node can reach a given endpoint (EID) into a database called the LISP map server.
- The LISP map server keeps the list of endpoint identifiers (EIDs) and matching routing locators (RLOCs) (which identify the fabric edge node that can reach the EID).
- In the future, when the fabric data plane needs to forward a message, it will look for and find the destination in the LISP map server's database.

For instance, switches SW3 and SW4 in Figure 17-8 each just learned about different subnets external to the SDA fabric. As noted at step 1 in the figure, switch SW3 sent a message to the LISP map server, registering the information about subnet 10.1.3.0/24 (an EID), with its RLOC setting to identify itself as the node that can reach that subnet. Step 2 shows an equivalent registration process, this time for SW4, with EID 10.1.4.0/24, and with R4's RLOC of 172.16.4.4. Note that the table at the bottom of the figure represents that data held by the LISP map server.

Key Topic



	EID	RLOC
1	10.1.3.0/24	172.16.3.3
2	10.1.4.0/24	172.16.4.4

Figure 17-8 *Edge Nodes Register IPv4 Prefixes (Endpoint IDs) with LISP Map Server*

When new incoming frames arrive, the ingress tunnel router (ITR)—the SDA node that receives the new frame from outside the SDA fabric—needs some help from the control plane. To where should the ITR forward this frame? And because SDA always forwards frames in the fabric over some VXLAN tunnel, what tunnel should the ITR use when forwarding the frame? For the first frame sent to a destination, the ITR has to follow a process like the following steps. The steps begin at step 3, as a continuation of Figure 17-8, with the action referenced in Figure 17-9:

3. An Ethernet frame to a new destination arrives at ingress edge node SW1 (upper left), and the switch does not know where to forward the frame.
4. The ingress node sends a message to the LISP map server asking if the LISP server knows how to reach IP address 10.1.3.1.
5. The LISP map server looks in its database and finds the entry it built back at step 1 in the previous figure, listing SW3's RLOC of 172.16.3.3.
6. The LISP map server contacts SW3—the node listed as the RLOC—to confirm that the entry is correct.
7. SW3 completes the process of informing the ingress node (SW1) that 10.1.3.1 can be reached through SW3.

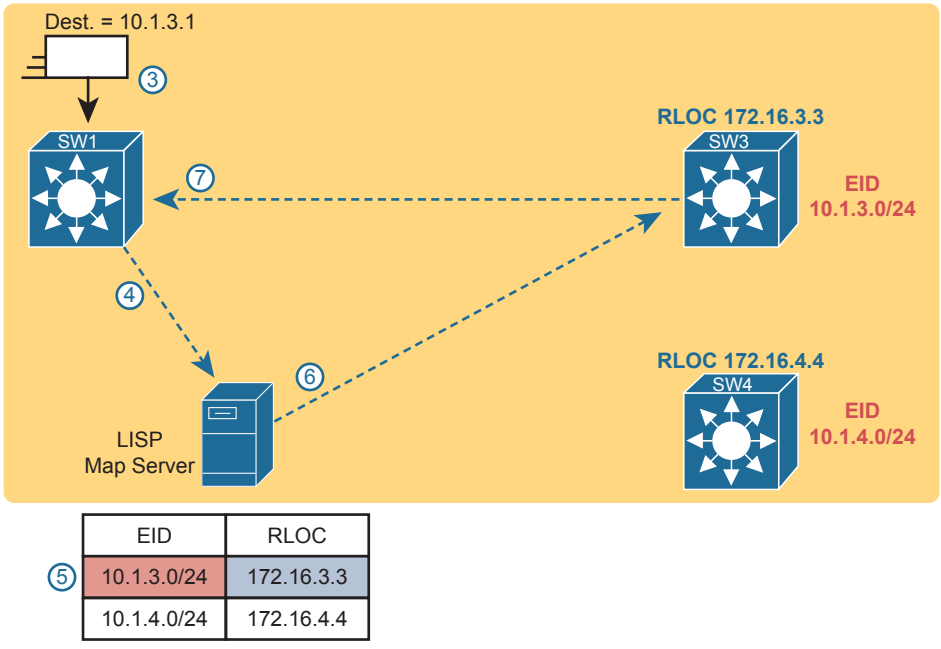
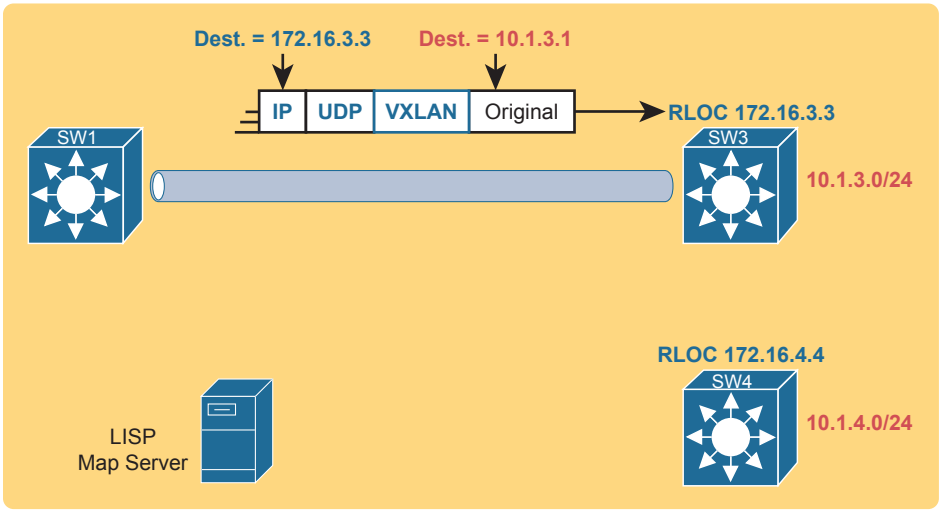


Figure 17-9 *Ingress Tunnel Router SW1 Discovers Egress Tunnel Router SW3 Using LISP*

To complete the story, now that ingress node SW1 knows that it can forward packets sent to endpoint 10.1.3.1 to the edge node with RLOC 172.16.3.3 (that is, SW3), SW1 encapsulates the original Ethernet frame as shown in Figure 17-9, with the original destination IP address of 10.1.3.1. It adds the IP, UDP, and VXLAN headers shown so it can deliver the message over the SDA network, with that outer IP header listing a destination IP address of the RLOC IP address, so that the message will arrive through the SDA fabric at SW3, as shown in Figure 17-10.

At this point, you should have a basic understanding of how the SDA fabric works. The underlay includes all the switches and links, along with IP connectivity, as a basis for forwarding data across the fabric. The overlay adds a different level of logic, with endpoint traffic flowing through VXLAN tunnels. This chapter has not mentioned any reasons that SDA might want to use these tunnels, but you will see one example by the end of the chapter. Suffice it to say that with the flexible VXLAN tunnels, SDA can encode header fields that let SDA create new networking features, all without suffering a performance penalty, as all the VXLAN processing happens in an ASIC.

This chapter next focuses on DNA Center and its role in managing and controlling SDA fabrics.



	EID	RLOC
①	10.1.3.0/24	172.16.3.3
②	10.1.4.0/24	172.16.4.4

Figure 17-10 *Ingress Tunnel Router (ITR) SW1 Forwards Based on LISP Mapping to SW3*

DNA Center and SDA Operation

Cisco DNA Center (www.cisco.com/go/dnacenter) has two notable roles:

- As the controller in a network that uses Cisco SDA
- As a network management platform for traditional (non-SDA) network devices, with an expectation that one day DNA Center may become Cisco's primary enterprise network management platform

The first role as SDA network controller gets most of the attention and is the topic of discussion in this second of the three major sections of this chapter. SDA and DNA Center go together, work closely together, and any serious use of SDA requires the use of DNA Center. At the same time, DNA Center can manage traditional network devices; the final major section of the chapter works through some comparisons.

Cisco DNA Center

Cisco DNA Center exists as a software application that Cisco delivers pre-installed on a Cisco DNA Center appliance. The software follows the same general controller architecture concepts as described in Chapter 16. Figure 17-11 shows the general ideas.

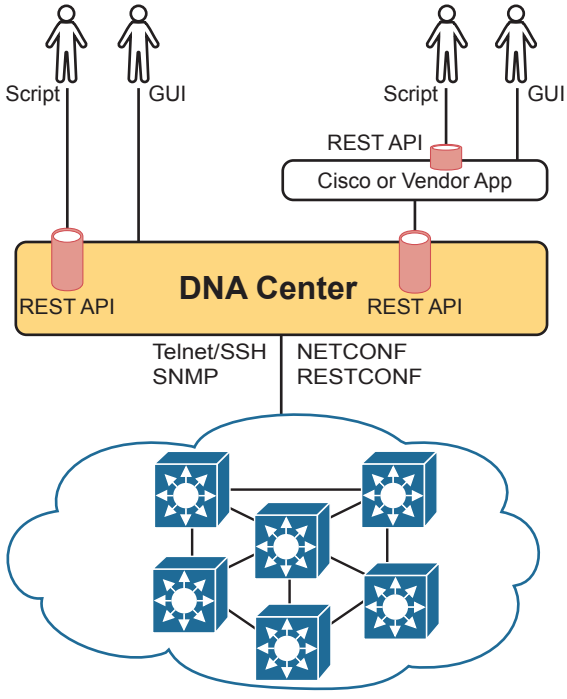


Figure 17-11 *Cisco DNA Center with Northbound and Southbound Interfaces*

Cisco DNA Center includes a robust northbound REST API along with a series of southbound APIs. For most of us, the northbound API matters most, because as the user of SDA networks, you interact with SDA using Cisco DNA Center's northbound REST API or the GUI interface. (Chapter 18, "Understanding REST and JSON," discusses the concepts behind REST APIs in more detail.)

Cisco DNA Center supports several southbound APIs so that the controller can communicate with the devices it manages. You can think of these as two categories:

- Protocols to support traditional networking devices/software versions: Telnet, SSH, SNMP
- Protocols to support more recent networking devices/software versions: NETCONF, RESTCONF

Cisco DNA Center needs the older protocols to be able to support the vast array of older Cisco devices and OS versions. Over time, Cisco has been adding support for NETCONF and RESTCONF to their more current hardware and software.

Cisco DNA Center and Scalable Groups

SDA creates many interesting new and powerful features beyond how traditional campus networks work. Cisco DNA Center not only enables an easier way to configure and operate those features, but it also completely changes the operational model. While the scope of CCNA does not allow us enough space to explore all of the features of SDA and DNA Center, this next topic looks at one feature as an example: scalable groups.

Issues with Traditional IP-Based Security

Imagine the life of one traditional IP ACL in an enterprise. Some requirements occurred, and an engineer built the first version of an ACL with three Access Control Entries (ACEs)—that is, **access-list** commands—with a **permit any** at the end of the list. Months later, the engineer added two more lines to the ACL, so the ACL has the number of ACEs shown in Figure 17-12. The figure notes the lines added for requests one and two with the circled numbers in the figure.

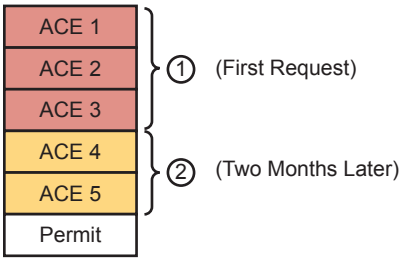


Figure 17-12 *Lines (ACEs) in an ACL after Two Changes*

Now think about that same ACL after four more requirements caused changes to the ACL, as noted in Figure 17-13. Some of the movement includes

- The ACEs for requirement two are now at the bottom of the ACL.
- Some ACEs, like ACE 5, apply to more than one of the implemented requirements.
- Some requirements, like requirement number five, required ACEs that overlap with multiple other requirements.

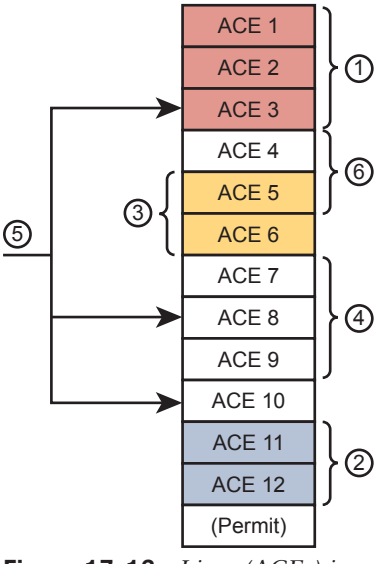


Figure 17-13 *Lines (ACEs) in an ACL after Six Changes*

Now imagine your next job is to add more ACEs for the next requirement (7). However, your boss also told you to reduce the length of the ACL, removing the ACEs from that one change made last August—you remember it, right? Such tasks are problematic at best.

With the scenario in Figure 17-13, no engineer could tell from looking at the ACL whether any lines in the ACL could be safely removed. You never know if an ACE was useful for one requirement or for many. If a requirement was removed, and you were even told which old project caused the original requirement so that you could look at your notes, you would not know if removing the ACEs would harm other requirements. Most of the time, ACL management suffers with these kinds of issues:

- ACEs cannot be removed from ACLs because of the risk of causing failures to the logic for some other past requirement.
- New changes become more and more challenging due to the length of the ACLs.
- Troubleshooting ACLs as a system—determining whether a packet would be delivered from end-to-end—becomes an even greater challenge.

SDA Security Based on User Groups

Imagine you could instead enforce security without even thinking about IP address ranges and ACLs. SDA does just that, with simple configuration, and the capability to add and remove the security policies at will.

First, for the big ideas. Imagine that over time, using SDA, six different security requirements occurred. For each project, the engineer would define the policy with DNA Center, either with the GUI or with the API. Then, as needed, DNA Center would configure the devices in the fabric to enforce the security, as shown in Figure 17-14.

Key Topic

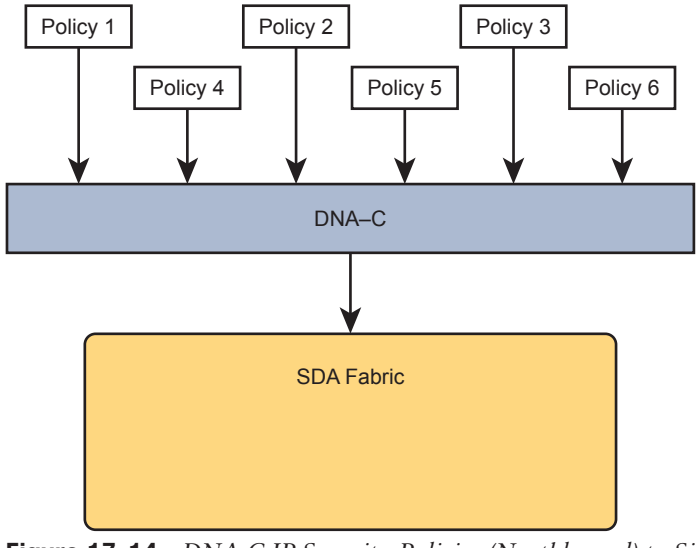


Figure 17-14 *DNA-C IP Security Policies (Northbound) to Simplify Operations*

NOTE The model in Figure 17-14 helps demonstrate the concept of intent-based networking (IBN). The engineer configures the intent or outcome desired from the network—in this case, a set of security policies. The controller communicates with the devices in the network, with the devices determining exactly what configuration and behavior are necessary to achieve those intended policies.

The SDA policy model solves the configuration and operational challenges with traditional ACLs. In fact, all those real issues with managing IP ACLs on each device are no longer issues with SDA's group-based security model. For instance:

- The engineer can consider each new security requirement separately, without analysis of an existing (possibly lengthy) ACL.
- Each new requirement can be considered without searching for all the ACLs in the likely paths between endpoints and analyzing each and every ACL.
- DNA Center (and related software) keeps the policies separate, with space to keep notes about the reason for the policy.
- Each policy can be removed without fear of impacting the logic of the other policies.

SDA and Cisco DNA achieve this particular feature by tying security to groups of users, called scalable groups, with each group assigned a scalable group tag (SGT). Then the engineer configures a grid that identifies which SGTs can send packets to which other SGTs. For instance, the grid might include SGTs for an employee group, the Internet (for the Enterprise's WAN routers that lead to the Internet), partner employees, and guests, with a grid like the one shown in Table 17-2.

Table 17-2 Access Table for SDA Scalable Group Access

Source \ Dest.	Employee	Internet	Partner	Guest
Employee	N/A	Permit	Permit	Deny
Internet	Permit	N/A	Permit	Permit
Partner	Permit	Permit	N/A	Deny
Guest	Deny	Permit	Deny	N/A

To link this security feature back to packet forwarding, consider when a new endpoint tries to send its first packet to a new destination. The ingress SDA node starts a process by sending messages to DNA Center. DNA Center then works with security tools in the network, like Cisco's Identity Services Engine (ISE), to identify the users and then match them to their respective SGTs. DNA Center then checks the logic similar to Table 17-2. If DNA Center sees a permit action between the source/destination pair of SGTs, DNA Center directs the edge nodes to create the VXLAN tunnel, as shown in Figure 17-15. If the security policies state that the two SGTs should not be allowed to communicate, DNA Center does not direct the fabric to create the tunnel, and the packets do not flow.

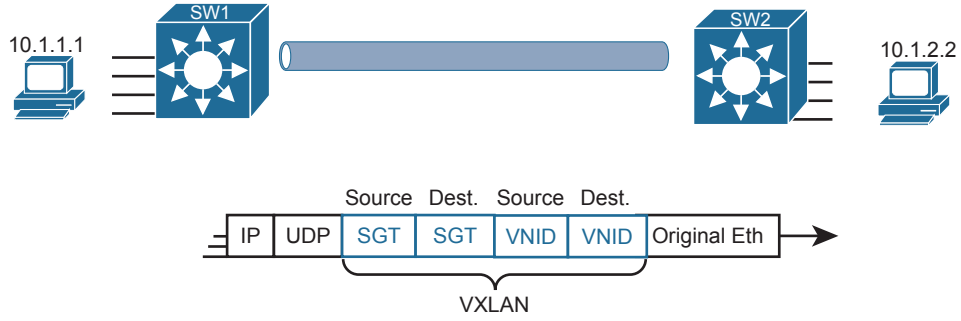


Figure 17-15 *VXLAN Header with Source and Destination SGTs and VNIDs Revealed*

NOTE The figure gives a brief insight into why SDA goes to the trouble of using VXLAN encapsulation for its data plane, rather than performing traditional Layer 2 switching or Layer 3 routing. The VXLAN header has great flexibility—in this case, used to define both a source and destination SGT, matching SDA's desired logic of allowing a subset of source/destination SGTs in the SDA fabric.

The operational model with scalable groups greatly simplifies security configuration and ongoing maintenance of the security policy, while focusing on the real goal: controlling access based on user. From a controller perspective, the fact that Cisco DNA Center acts as much more than a management platform, and instead as a controller of the activities in the network, makes for a much more powerful set of features and capabilities.

DNA Center as a Network Management Platform

CCNA Exam topic 6.4 asks you to compare traditional network management with DNA Center:

> Compare traditional campus device management with Cisco DNA Center enabled device management

Note that the exam topic does not identify which traditional management product. In fact, Cisco tends to shy away from product details in most of its career certifications. So, to think through this exam topic, you need to think in general about network management products. But it also helps to think about specific products—but temper that by focusing on the more prominent features and major functions.

This section uses Cisco Prime Infrastructure (PI) (www.cisco.com/go/primeinfrastructure) as an example of a traditional enterprise network management product. For many years, Cisco Prime Infrastructure has been Cisco's primary network management product for the enterprise. It includes the following features:

Key Topic

- **Single-pane-of-glass:** Provides one GUI from which to launch all PI functions and features
- **Discovery, inventory, and topology:** Discovers network devices, builds an inventory, and arranges them in a topology map
- **Entire enterprise:** Provides support for traditional enterprise LAN, WAN, and data center management functions
- **Methods and protocols:** Uses SNMP, SSH, and Telnet, as well as CDP and LLDP, to discover and learn information about the devices in the network
- **Lifecycle management:** Supports different tasks to install a new device (day 0), configure it to be working in production (day 1), and perform ongoing monitoring and make changes (day *n*)
- **Application visibility:** Simplifies QoS configuration deployment to each device
- **Converged wired and wireless:** Enables you to manage both the wired and wireless LAN from the same management platform

- **Software Image Management (SWIM):** Manages software images on network devices and automates updates
- **Plug-and-Play:** Performs initial installation tasks for new network devices after you physically install the new device, connect a network cable, and power on

PI itself runs as an application on a server platform with GUI access via a web browser. The PI server can be purchased from Cisco as a software package to be installed and run on your servers, or as a physical appliance.

The next few pages now compare and contrast DNA Center to traditional management tools like PI.

DNA Center Similarities to Traditional Management

If you read the user's guide for DNA Center and look through all the features, you will find all the features just listed here as traditional management features. For instance, both can discover network devices and create a network topology map. Human operators (rather than automated processes) often start with the topology map, expecting notices (flashing lights, red colors) to denote issues in the network.

As an example, Figure 17-16 shows a topology map from DNA Center. Both PI and DNA Center can perform a discover process to find all the devices in the network and then build topology maps to show the devices. (Interestingly, DNA Center can work with PI, using the data discovered by PI rather than performing the discovery work again.)

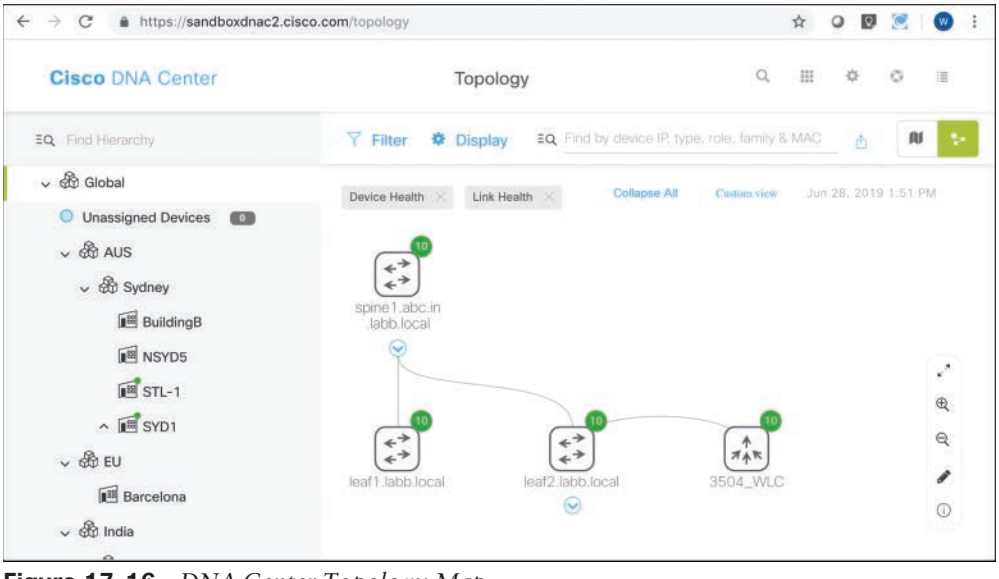


Figure 17-16 *DNA Center Topology Map*

The GUI mechanisms are relatively intuitive, with the ability to click into additional or less detail. Figure 17-17 shows a little more detail after hovering over and clicking on one of the nodes in the topology from Figure 17-16, typical actions and results in many management products.

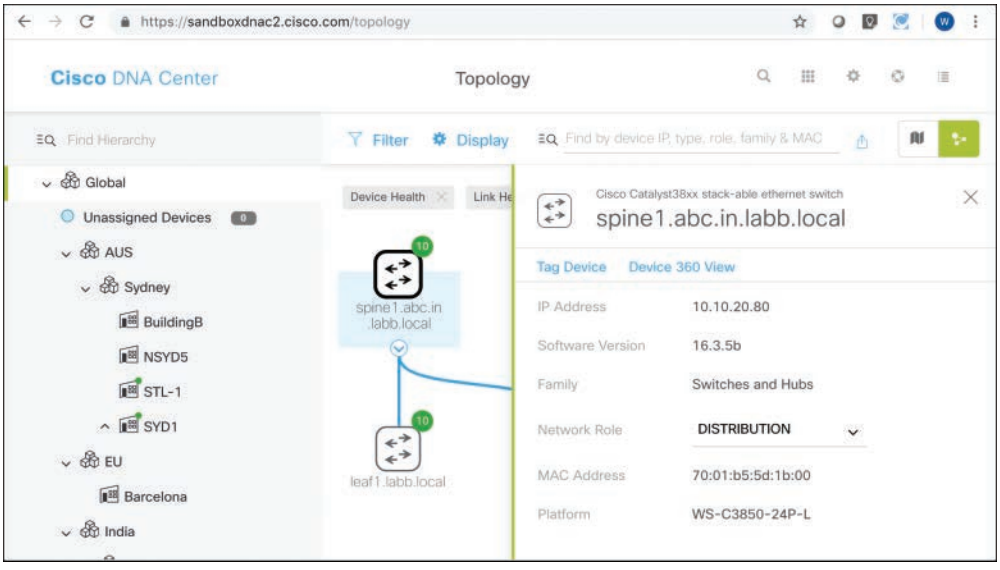


Figure 17-17 *Hover and Click Details About One Cisco 9300 Switch from DNA Center*

I encourage you to take some time to use and watch some videos about Cisco DNA Center. The "Chapter Review" section for this chapter on the companion website lists some links for good videos. Also, start at https://developer.cisco.com and look for Cisco DNA Center sandbox labs to find a place to experiment with Cisco DNA Center.

DNA Center Differences with Traditional Management

In a broad sense, there are several fundamental differences between Cisco DNA Center and traditional network management platforms like Cisco PI. The largest difference: Cisco DNA Center supports SDA, whereas other management apps do not. At the same time, given its long history, as of the time this chapter was written, Cisco PI still had some traditional management features not found in Cisco DNA Center. So think of PI as comprehensive to traditional device management, with Cisco DNA Center having many of those features, while focusing on future features like SDA support.

NOTE Cisco hopes to continue to update Cisco DNA Center's traditional network management features to be equivalent compared to Cisco PI, to the point at which DNA Center could replace PI.

In terms of intent and strategy, Cisco focuses their development of Cisco DNA Center features toward simplifying the work done by enterprises, with resulting reduced costs and much faster deployment of changes. Cisco DNA Center features help make initial installation easier, simplify the work to implement features that traditionally have challenging configuration, and use tools to help you notice issues more quickly. Some of the features unique to Cisco DNA Center include

Key Topic

- **EasyQoS:** Deploys QoS, one of the most complicated features to configure manually, with just a few simple choices from Cisco DNA Center
- **Encrypted traffic analysis:** Enables Cisco DNA to use algorithms to recognize security threats even in encrypted traffic

- **Device 360 and Client 360:** Gives a comprehensive (360-degree) view of the health of the device
- **Network time travel:** Shows past client performance in a timeline for comparison to current behavior
- **Path trace:** Discovers the actual path packets would take from source to destination based on current forwarding tables

Just to expound on one feature as an example, Cisco DNA Center's Path Trace feature goes far beyond a traditional management application. A typical network management app might show a map of the network and let you click through to find the configuration on each device, including ACLs. The path trace feature goes much further. The DNA user (from the GUI or the API) specifies a source and destination host and optionally transport protocol and ports. Then the path trace feature shows a map of the path through the network and shows which ACLs are in the path, and whether they would permit or deny the packet.

All of Cisco Digital Network Architecture sets about to help customers reach some big goals: reduced costs, reduced risks, better security and compliance, faster deployment of services through automation and simplified processes, and the list goes on. Cisco DNA Center plays an important role, with all the functions available through its robust northbound API, and with its intent-based networking approach for SDA. Cisco DNA Center represents the future of network management for Cisco enterprises.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 17-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 17-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP

Review All the Key Topics

Key Topic

Table 17-4 Key Topics for Chapter 17

Key Topic Element	Description	Page Number
List	Definitions for overlay, underlay, and fabric	384
Figure 17-2	SDA overlay and underlay	386
List	SDA fabric edge, fabric border, and fabric control node roles	387
List	Attributes of the SDA underlay	389
List	SDA VXLAN tunneling benefits	390
Figure 17-5	VXLAN encapsulation process with SDA	391
Figure 17-8	Registering SDA endpoint IDs (EIDs) with the map server	393
Figure 17-14	DNA Center shown controlling the fabric to implement group-based security	398
List	DNA Center features that go beyond traditional network management	400
List	Features unique to DNA Center	402

Key Terms You Should Know

Software-Defined Access, overlay, underlay, fabric, DNA Center, fabric edge node, VXLAN, LISP, scalable group tag (SGT), Cisco Prime Infrastructure (PI)

This page intentionally left blank

CHAPTER 18

Understanding REST and JSON

This chapter covers the following exam topics:

6.0 Automation and Programmability

6.5 Describe characteristics of REST-based APIs (CRUD, HTTP verbs, and data encoding)

6.7 Interpret JSON encoded data

To automate and program networks, some automation software does several tasks. The software analyzes data in the form of variables, makes decisions based on that analysis, and then may take action to change the configuration of network devices or report facts about the state of the network.

The different automation functions reside on different devices: the network engineer's device, a server, a controller, and the various network devices themselves. For these related automation processes to work well, all these software components need useful well-defined conventions to allow easy communication between software components.

This chapter focuses on two conventions that allow automation software to communicate. The first major section discusses application programming interfaces (APIs), specifically APIs that follow a style called REpresentational State Transfer (REST). APIs of any kind create a way for software applications to communicate, while RESTful APIs (APIs that use REST conventions) follow a particular set of software rules. Many APIs used in network automation today use REST-based APIs.

The second half of the chapter focuses on the conventions and standards for the data variables exchanged over APIs, with a focus on one: JavaScript Object Notation (JSON). If REST provides one standard method of how two automation programs should communicate over a network, JSON then defines how to communicate the variables used by a program: the variable names, their values, and the data structures of those variables.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 18-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
REST-based APIs	1–3
Data Models and JSON	4–6

1. Which of the following are required attributes of a REST-based API? (Choose two answers.)
   a. Uses HTTP
   b. Objects noted as to whether they can be cached
   c. Classful operation
   d. Client/server architecture

2. Which answers list a matching software development CRUD action to an HTTP verb that performs that action? (Choose two answers.)
   a. CRUD create and HTTP PATCH
   b. CRUD update and HTTP PATCH
   c. CRUD delete and HTTP PUT
   d. CRUD read and HTTP GET

3. Examine the following URI that works with a Cisco DNA Controller:

```
https://dnac.example.com/dna/intent/api/v1/network-device?managementIPAddress=10.10.22.74
```

   Which part of the URI, per the API documentation, is considered to identify the resource but not any parameters?
   a. https://
   b. dnac.example.com
   c. dna/intent/api/v1/network-device
   d. managementIPAddress=10.10.22.74

4. Which of the following data serialization and data modeling languages would be most likely to be used in a response from a REST-based server API used for networking applications? (Choose two answers.)
   a. JSON
   b. YAML
   c. JavaScript
   d. XML

5. Which answers correctly describe the format of the JSON text below? (Choose two answers.)

```
{ "myvariable":[1,2,3] }
```

   a. One JSON object that has one key:value pair
   b. One JSON object that has three key:value pairs
   c. A JSON object whose value is a second JSON object
   d. A JSON object whose value is a JSON array

6. Which answers refer to JSON values rather than JSON keys as found in the sample JSON data? (Choose two answers.)

```
{
    "response": {
        "type": "Cisco Catalyst 9300 Switch",
        "family": "Switches and Hubs",
        "role": "ACCESS",
        "managementIpAddress": "10.10.22.66"
    }
}
```

   a. "response"
   b. "type"
   c. "ACCESS"
   d. The entire gray area

Foundation Topics

REST-Based APIs

Applications use *application programming interfaces* (APIs) to communicate. To do so, one program can learn the variables and data structures used by another program, making logic choices based on those values, changing the values of those variables, creating new variables, and deleting variables. APIs allow programs running on different computers to work cooperatively, exchanging data to achieve some goal.

In an API software world, some applications create an API, with many other applications using (consuming) the API. Software developers add APIs to their software so other application software can make use of the first application's features.

When writing an application, the developer will write some code, but often the developer may do a lot of work by looking for APIs that can provide the data and functions, reducing the amount of new code that must be written. As a result, much of modern software development centers on understanding and learning new APIs, along with the available libraries (prebuilt software that can be used to accomplish tasks rather than writing the equivalent from scratch).

Several types of APIs exist, each with a different set of conventions to meet a different set of needs. The CCNA blueprint mentions one type of API—REpresentational State Transfer (REST)—because of its popularity as a type of API in networking automation applications. This first major section of the chapter takes a closer look at REST-based APIs.

REST-Based (RESTful) APIs

REST APIs follow a set of foundational rules about what makes a REST API and what does not. First, from a literal perspective, REST APIs include the six attributes defined a few decades

back by its creator, Roy Fielding. (You can find a good summary at https://restfulapi.net). Those six attributes are

Key Topic

- Client/server architecture
- Stateless operation
- Clear statement of cacheable/uncacheable
- Uniform interface
- Layered
- Code-on-demand

The first three of these attributes get at the heart of how a REST API works. You can more easily see those first three features at work with networking REST APIs, so the next few paragraphs give a little more explanation about those first three points.

Client/Server Architecture

Like many applications, REST applications use a client/server architectural model. First, an application developer creates a REST API, and that application, when executing, acts as a REST server. Any other application can make a REST API call (the REST client) by executing some code that causes a request to flow from the client to the server. For instance, in Figure 18-1

1. The REST client on the left executes a REST API call, which generates a message sent to the REST server.
2. The REST server on the right has API code that considers the request and decides how to reply.
3. The REST server sends back the reply message with the appropriate data variables in the reply message.

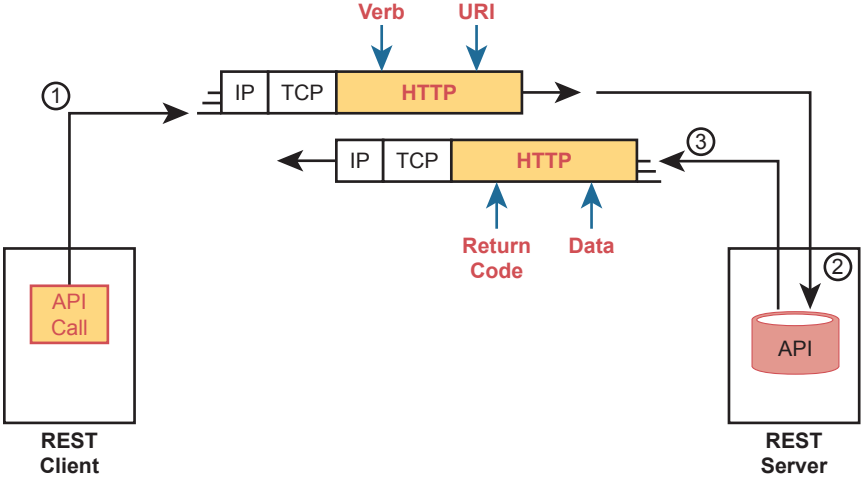


Figure 18-1 *Client/Server Operation with REST*

NOTE Figure 18-1 shows the use of HTTP. While many REST APIs use HTTP, the use of HTTP is not a requirement for an API to be considered RESTful.

Stateless Operation

The stateless attribute of REST APIs means that REST does not record and use information about one API exchange for the purpose of how subsequent API exchanges are processed. In other words, each API request and reply does not use any other past history considered when processing the request.

For comparison, the TCP protocol uses a stateful approach, whereas UDP uses stateless operation. A TCP connection requires the endpoints to initialize variables on each end, with those variables updating over time, and with those variables being used for subsequent TCP messages. For instance, TCP uses sequence numbers and acknowledgment numbers to manage the flow of data in a TCP connection.

Cacheable (or Not)

To appreciate what is meant by *cacheable*, consider what happens when you browse a website. When your browser loads a new web page, the page itself contains a variety of objects (text, images, videos, audio). Some objects seldom change, so it would be better to download the object once and not download it again; in that case, the server marks that object as cacheable. For instance, a logo or other image shown on many pages of a website would almost never change and would likely be cacheable. However, the product list returned in your most recent search of the website would not be cacheable because the server would want to update and supply a new list each time you request the page.

REST APIs require that any resource requested via an API call have a clear method by which to mark the resource as cacheable or not. The goals remain the same: improve performance by retrieving resources less often (cacheable). Note that cacheable resources are marked with a timeframe so that the client knows when to ask for a new copy of the resource again.

Background: Data and Variables

To appreciate a few of the upcoming topics, it helps to have a basic idea about how programming languages use variables. Anyone who has done even a small amount of programming should have enough background, but for those who have not written programs before, this next topic gives you enough background about data and variables inside programs to understand the next topic.

If you have some programming experience and already know about simple variables, list variables, and dictionary variables, then feel free to skip ahead to the section "REST APIs and HTTP."

Simple Variables

Applications all process data with the same general actions, starting with some kind of input. The program needs data to process, so the input process reads files, sends database queries to a database server, or makes API calls to retrieve data from another application's API. The goal: gather the data that the program needs to process to do its work.

Answers to the "Do I Know This Already?" quiz:

1 B, D **2** B, D **3** C **4** A, D **5** A, D **6** C, D

Programs then process data by making comparisons, making decisions, creating new variables, and performing mathematical formulas to analyze the data. All that logic uses variables. For instance, a program might process data with the following logic:

> If the router's G0/0 interface has a configuration setting of **switchport mode dynamic auto**, then gather more data to ensure that interface currently operates as a trunk rather than as an access port.

In programming, a variable is a name or label that has an assigned value. To get a general sense for programming variables, you can think of variables much like variables from algebra equations back in school. Example 18-1 shows some samples of variables of different types in a Python program (the Python language is the most popular language today for writing network automation applications). This program begins with a comment (the top three lines with triple single quotes) and then creates four variables, assigning them to different values, and prints a line of output: "The product is -12."

Example 18-1 *Simple Python Program That Shows a Product*

```
'''
Sample program to multiply two numbers and display the result
'''
x = 3
y = -4
z = 1.247
heading = "The product is "
print(heading,x*y)
```


The variables in Example 18-1 can be called *simple variables* because each variable name has a single value associated with it. Simple variables have one variable name and one associated value, so they have a simple structure.

The values of simple variables can have a variety of formats, as shown in Example 18-1. The example includes variables that contain

- Unsigned integers (x)
- Signed integers (y)
- Floating-point numbers (z)
- Text (heading)

List and Dictionary Variables

While simple variables have many great uses, programs need variables with more complex *data structures*. In programming, a data structure defines a related set of variables and values. For instance, Python uses list variables so that one variable name is assigned a value that is a list of values rather than a single value. You could imagine that a network automation program might want to have lists, such as a list of devices being managed, a list of interfaces on a device, or list of configuration settings on an interface.

First, consider the variable named list1 in Example 18-2; note that the lines that begin with a # are comment lines.

Example 18-2 *Sample List and Dictionary Variables in Python*

```
# Variable list1 is a list in Python (called an array in Java)
list1 = ["g0/0", "g0/1", "g0/2"]

# Variable dict1 is a dictionary (called an associative array in Java)
dict1 = {"config_speed":'auto', "config_duplex":"auto", "config_ip":"10.1.1.1"}
```

Even if you have never seen Python code before, you can guess at some of the meaning of the list1 variable. The code assigns variable list1 to a value that itself is a list of three text strings. Note that the list could include text, unsigned integers, signed integers, and so on.

Figure 18-2 shows the data structure behind variable list1 in Example 18-2. The variable is assigned to the list, with the list having three list elements.

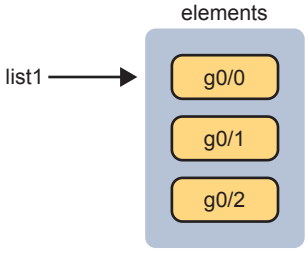


Figure 18-2 *The List Data Structure in Python*

Python supports a similar data structure called a *dictionary*. If you think of the contents of a dictionary for the English language, that dictionary lists a series of paired items: a term and a matching definition. With programming languages like Python, the dictionary data structure lists paired items as well: *keys* (like terms) and *values* (like definitions). Figure 18-3 shows the structure of that dictionary value matching the dict1 variable at the bottom of Example 18-2. Note that each key and its value is called a *key:value pair*.

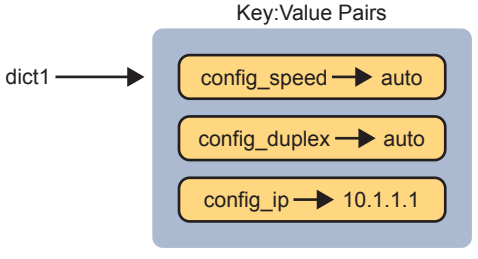


Figure 18-3 *Dictionary Data Structures in Python*

Data structures can get more complex. Additionally, the data structures can be nested. For instance, a single variable's value could be a list, with each list element being a dictionary, with the values in some key:value pairs being other lists, and so on. For now, be aware of the fact that programs use simple variables but also use list and dictionary variables to make it easier to perform different kinds of logic.

REST APIs and HTTP

APIs exist to allow two programs to exchange data. Some APIs may be designed as an interface between programs running on the same computer, so the communication between programs happens within a single operating system. Many APIs need to be available to programs that run on other computers, so the API must define the type of networking protocols supported by the API—and many REST-based APIs use the HTTP protocol.

The creators of REST-based APIs often choose HTTP because HTTP's logic matches some of the concepts defined more generally for REST APIs. HTTP uses the same principles as REST: it operates with a client/server model; it uses a stateless operational model; and it includes headers that clearly mark objects as cacheable or not cacheable. It also includes verbs—words that dictate the desired action for a pair HTTP Request and Reply—which matches how applications like to work.

This section breaks down the fundamentals of some programming terminology, how that matches HTTP verbs, and how REST APIs make use of Uniform Resource Identifiers (URIs) to specify the data desired from a RESTful API call.

Software CRUD Actions and HTTP Verbs

The software industry uses a memorable acronym—*CRUD*—for the four primary actions performed by an application. Those actions are

Create: Allows the client to create some new instances of variables and data structures at the server and initialize their values as kept at the server

Read: Allows the client to retrieve (read) the current value of variables that exist at the server, storing a copy of the variables, structures, and values at the client

Update: Allows the client to change (update) the value of variables that exist at the server

Delete: Allows the client to delete from the server different instances of data variables

For instance, if using the northbound REST API of a DNA controller, as discussed in Chapter 17, "Cisco Software-Defined Access (SDA)," you might want to create something new, like a new security policy. From a programming perspective, the security policy exists as a related set of configuration settings on the DNA controller, internally represented by variables. To do that, a REST client application would use a create action, using the DNA Center RESTful API, that created variables on the DNA Controller via the DNA Center REST API. The concept of creating new configuration at the controller is performed via the API using a create action per the CRUD generic acronym.

Other examples of CRUD actions include a check of the status of that new configuration (a read action), an update to change some specific setting in the new configuration (an update action), or an action to remove the security policy definition completely (a delete action).

HTTP uses verbs that mirror CRUD actions. HTTP defines the concept of an HTTP request and reply, with the client sending a request and with the server answering back with a reply. Each request/reply lists an action verb in the HTTP request header, which defines the HTTP action. The HTTP messages also include a URI, which identifies the resource being manipulated for this request. As always, the HTTP message is carried in IP and TCP, with headers and data, as represented in Figure 18-4.

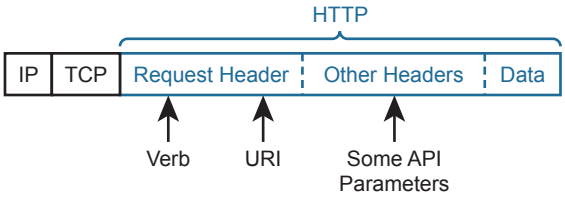


Figure 18-4 *HTTP Verb and URI in an HTTP Request Header*

To get some perspective about HTTP, ignore REST for a moment. Whenever you open a web browser and click a link, your browser generates an HTTP GET request message similar to Figure 18-4 in structure. The message includes an HTTP header with the GET verb and the URI. The resources returned in the reply are the components of a web page, like text files, image files, and video files.

HTTP works well with REST in part because HTTP has verbs that match the common program actions in the CRUD paradigm. Table 18-2 lists the HTTP verbs and CRUD terms for easy reference and study.

Key Topic

Table 18-2 Comparing CRUD Actions to REST Verbs

Action	CRUD Term	REST (HTTP) Verb
Create new data structures and variables	Create	POST
Read (retrieve) variable names, structures, and values	Read	GET
Update or replace values of some variable	Update	PATCH, PUT
Delete some variables and data structures	Delete	DELETE

NOTE While Table 18-2 lists HTTP POST as a create action and HTTP PATCH and PUT as CRUD update actions, all three of these HTTP verbs might be used both for create and for update actions in some cases.

Using URIs with HTTP to Specify the Resource

In addition to using HTTP verbs to perform the CRUD functions for an application, REST uses URIs to identify what resource the HTTP request acts on. For REST APIs, the resource can be any one of the many resources defined by the API. Each resource contains a set of related variables, defined by the API and identified by a URI.

For instance, imagine a user creates a REST-based API. When she does so, she creates a set of resources that she wants to make available via the API, and she also assigns a unique URI to each resource. In other words, the API creator creates a URI and a matching set of variables, and defines the actions that can be performed against those variables (read, update, and so on).

The API creator also creates API documentation that lists the resources and the URI that identifies each resource, among other details. The programmer for a REST client application can read the API documentation, build a REST API request, and ask for the specific resource, as shown in the example in Figure 18-5.

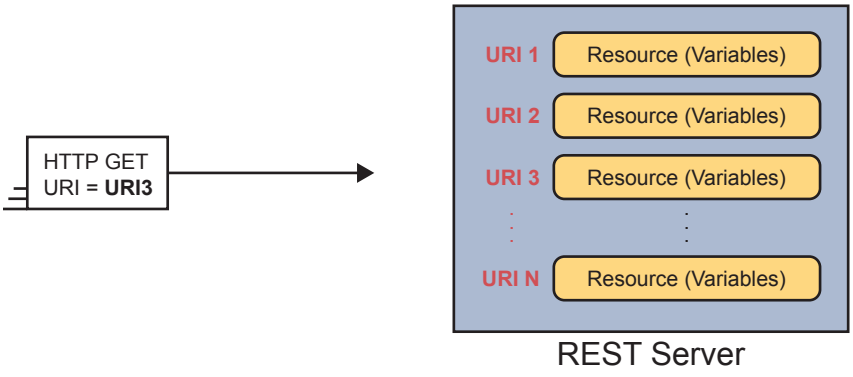


Figure 18-5 *One URI for Each API Resource—Conceptual View*

Figure 18-5 shows the URIs as generic values; however, today's network engineers need to be able to read API documentation, see URIs in that documentation, and understand the meaning of each part of the URI. Figure 18-6 shows a URI specific to the Cisco DNA Center northbound REST API as an example of some of the components of the URI.

Figure 18-6 *URI Structure for REST GET Request*

The figure shows these important values and concepts:

HTTPS: The letters before the :// identify the protocol used—in this case, HTTP Secure (which uses HTTP with SSL encryption).

Hostname or IP Address: This value sits between the // and first /, and identifies the host; if using a hostname, the REST client must perform name resolution to learn the IP address of the REST server.

Path (Resource): This value sits after the first / and finishes either at the end of the URI or before any additional fields (like a parameter query field). HTTP calls this field the path, but for use with REST, the field uniquely identifies the resource as defined by the API.

To drive home the connection between the API, URI, and resource part of the API, it can be helpful to just do a general tour of the API documentation for any REST-based API. For instance, when Cisco created DNA Center, it created the REST-based northbound interface and chose one URI as shown in Figure 18-6. Figure 18-7 shows a copy of the doc page for that particular resource for comparison. Go to https://developer.cisco.com and search for "Cisco DNA Center API documentation." Continue to search for yourself to see more examples of the resources defined by the Cisco DNA Center API.

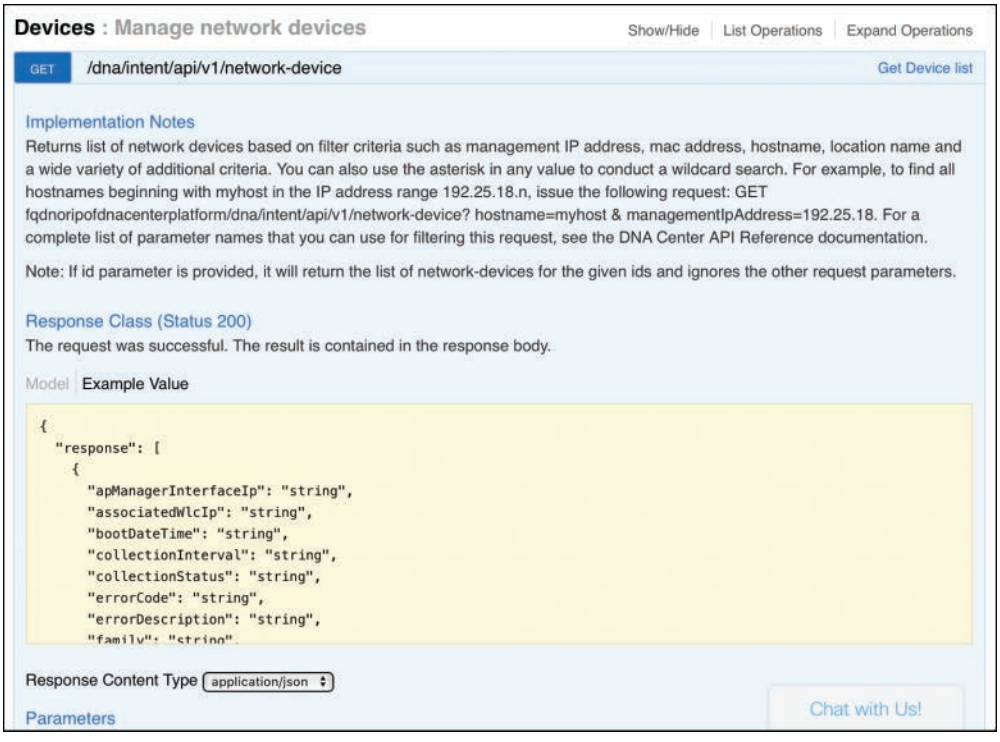


Figure 18-7 *DNA Center API Doc Page for the Network Device (List) Resource*

Many of the HTTP request messages need to pass information to the REST server beyond the API. Some of that data can be passed in header fields—for instance, REST APIs use HTTP header fields to encode much of the authentication information for REST calls. Additionally, parameters related to a REST call can be passed as parameters as part of the URI itself.

For instance, the URI in Figure 18-6 asks the Cisco DNA Center for a list of all known devices, with Cisco DNA Center returning a dictionary of values for each device. You might instead want that dictionary of values for only a single device. The Cisco DNA Center API allows for just that by tacking on the following to the end of the URI shown in Figure 18-6.

```
?managementIPAddress=10.10.22.66&macAddress=f8:7b:20:67:62:80
```

Figure 18-8 summarizes the major components of the URIs commonly used with a REST API, with the resource and parameter parts of the URI identifying specifically what the API should supply to the REST client.

Key Topic

Figure 18-8 *Example Components of a URI Used in a REST API Call*

Example of REST API Call to DNA Center

To pull some of the REST API concepts together, the next few pages work through a few sample API calls using a software application called an API development environment tool.

For a bit of development perspective, when working to automate some part of your network operation tasks, you would eventually use a program that made API calls. However, early in the process of developing an application, you might first focus on the data available from the API and ignore all the programming details at first. API development environments let you focus on the API calls. Later, that same tool can typically generate correct code that you can copy into your program to make the API calls.

The examples in this section use an app named Postman. Postman can be downloaded for free (www.postman.co) and used as shown in this section. Note that Cisco DevNet makes extensive use of Postman in its many labs and examples.

The first example shows a screenshot of a part of the Postman app after it sends a REST client GET request to a DNA Center REST API (see Figure 18-9). In particular, look for the following:

- The URI, near the top, lists a hostname of sandboxdnac2.cisco.com, which is an always-on DNA Center instance supplied by Cisco's DevNet site (which you can use).
- The resource part of the URI shows the same resource listed earlier in Figure 18-6, asking for a list of devices.
- The bottom center of the window shows the data returned by the DNA Center REST HTTP GET response.
- At the middle right, it lists the GET response's status code of 200, meaning "OK."

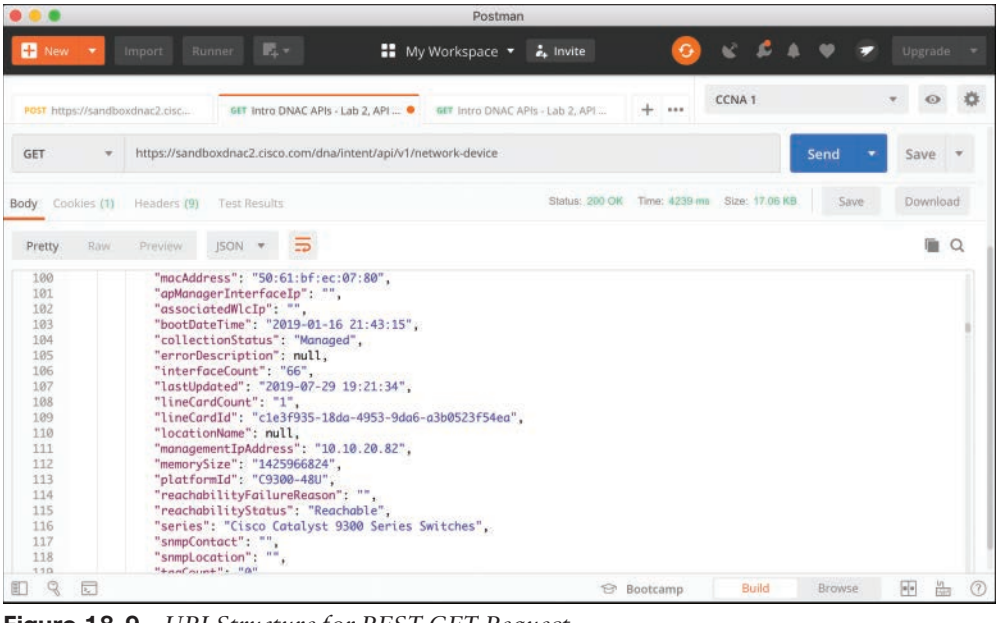


Figure 18-9 *URI Structure for REST GET Request*

Take a moment to look through the data at the bottom of the Postman window in Figure 18-9. The text follows a data modeling format called JavaScript Object Notation (JSON), which is one of the main topics for the remainder of the chapter. However, armed with just a knowledge of routers, you can find a few facts that look familiar. To help you see the text, Example 18-3 shows an edited (shortened to reduce the length) view of some of the JSON output in that window, just so you can see the format and some of the data returned in this single API call.

Example 18-3 *JSON Output from a REST API Call*

```
{
    "response": {
        "type": "Cisco Catalyst 9300 Switch",
        "family": "Switches and Hubs",
        "role": "ACCESS",
        "macAddress": "f8:7b:20:67:62:80",
        "hostname": "cat_9k_1",
        "serialNumber": "FCW2136L0AK",
        "softwareVersion": "16.6.1",
        "upTime": "17 days, 22:51:04.26",
        "interfaceCount": "41",
        "lineCardCount": "2",
        "managementIpAddress": "10.10.22.66",
        "series": "Cisco Catalyst 9300 Series Switches",
        "softwareType": "IOS-XE"
    }
}
```

API development tools like Postman help you work out the particulars of each API call, save the details, and share with other engineers and developers. Eventually, you will be ready to make the API call from a program. With a simple click from the Postman UI, Postman supplies the code to copy/paste into your program so that it returns all the output shown in the center/bottom of the window back as a variable to your program.

By now, you have a good foundational knowledge of the mechanics of REST APIs. By learning some skills, and using the API documentation for any REST API, you could now experiment with and try to make REST API calls. For many of those, the data will return to you as text, often in JSON format, so the second half of the chapter examines the meaning of that text.

Data Serialization and JSON

In your journey to become a modern network engineer with network automation skills, you will learn to understand several data serialization languages. Each data serialization language provides methods of using text to describe variables, with a goal of being able to send that text over a network or to store that text in a file. Data serialization languages give us a way to represent variables with text rather than in the internal representation used by any particular programming language.

Each data serialization language enables API servers to return data so that the API client can replicate the same variable names as well as data structures as found on the API server. To describe the data structures, the data serialization languages include special characters and conventions that communicate ideas about list variables, dictionary variables, and other more complex data structures.

This second major section of the chapter examines the concept of a data serialization language, with a focus on the one data modeling language as mentioned in the current CCNA blueprint: JavaScript Object Notation (JSON).

The Need for a Data Model with APIs

This section shows some ideas of how to move variables in a program on a server to a client program. First, Figure 18-10 and surrounding text show a nonworking example as a way to identify some of the challenges with copying variable values from one device to another. Then Figure 18-11 and its related text show how to use a data serialization language to solve the problems shown around Figure 18-10.

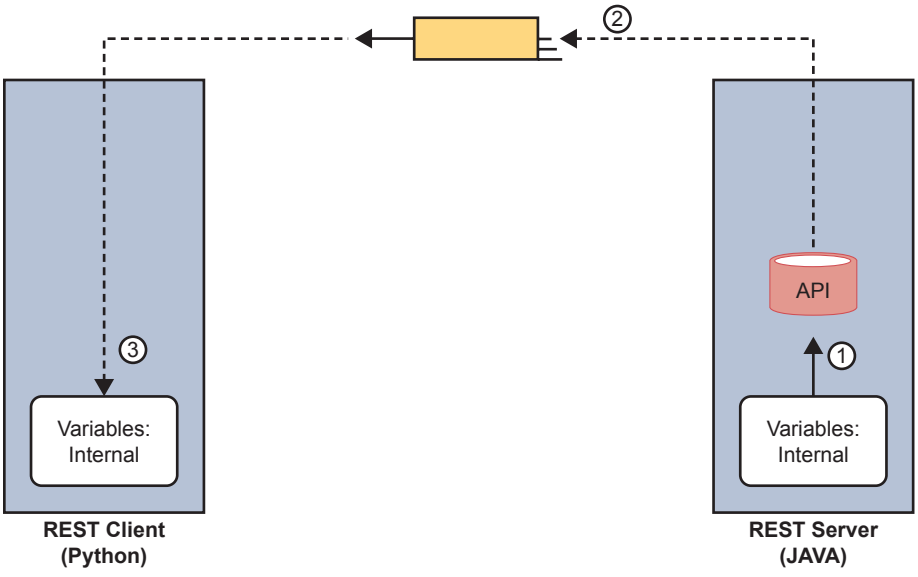


Figure 18-10 *Broken Concept: Exchanging Internal Representations of Variables*

First, for the nonworking example, consider the flow and numbered steps in Figure 18-10. A REST client sits on the left. The REST client asks for a resource, and the server needs to reply. In REST, a resource is a set of variables as defined by the API, so the REST server needs to return a set of variables to the REST client on the left. The steps in the figure run as follows:

1. The REST server (a JAVA application) takes a copy of the stored variables in RAM (step 1) in response to the REST request.
2. The REST API code creates the REST reply and sends it over the network, placing an exact replica of what the REST server had in RAM to represent the variables in that resource.

3. The REST client (a Python application) receives the REST reply message, storing the exact same bits and bytes into its RAM, in an attempt to have a copy of the variables, data, and data structures on the server.

The process shown in Figure 18-10 does not work (and is not attempted) because the REST client programs may not store variables in the same ways. First, programs written in different languages use different conventions to store their variables internally because there is no standard for internal variable storage across languages. In fact, programs written in the same language but with different versions of that language may not store all their variables with the same internal conventions.

To overcome these issues, applications need a standard method to represent variables for transmission and storage of those variables outside the program. *Data serialization languages* provide that function.

Figure 18-11 shows the correct process flow in comparison to Figure 18-10 with the data serialization process included:

1. The server collects the internally represented data and gives it to the API code.
2. The API converts the internal representation to a data model representing those variables (with JSON shown in the figure).
3. The server sends the data model in JSON format via messages across the network.
4. The REST client takes the received data and converts the JSON-formatted data into variables in the native format of the client application.

Key Topic

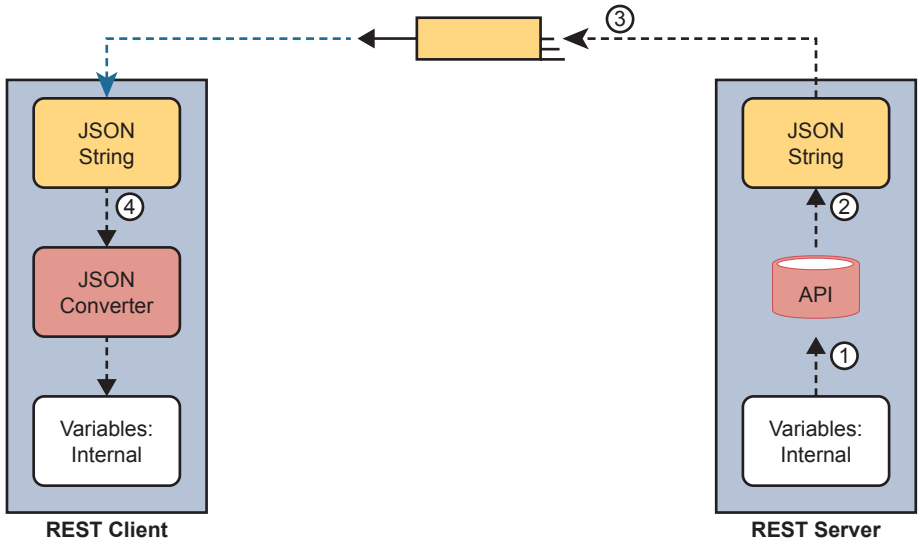


Figure 18-11 *Correct Concept: Exchanging Internal Representations of Variables*

At the end of the process, the REST client application now has equivalent variables to the ones it requested from the server in the API call. Note that the final step—to convert from the data serialization language to the native format—can be as little as a single line of code!

Finally, note that while data serialization languages like JSON enable applications to exchange variables over a network, applications can also store data in JSON format.

Data Serialization Languages

You will hear about and eventually use several data serialization and data modeling languages the more you learn about network automation. While the current CCNA blueprint mentions only JSON, learning a few facts about some of the alternatives can be helpful to add a little context to your new knowledge of JSON. These different data serialization languages exist to meet different needs that have arisen over the years. This next short section highlights four such languages.

NOTE The terms *data serialization language* and *data modeling language* should be considered equivalent for the purposes of this section.

JSON

JavaScript Object Notation attempts to strike a balance between human and machine readability. Armed with a few JSON rules, most humans can read JSON data, move past simply guessing at what it means, and confidently interpret the data structures defined by the JSON data. At the same time, JSON data makes it easy for programs to convert JSON text into variables, making it very useful for data exchange between applications using APIs.

You can find the details of JSON in IETF RFC 8259 and in a number of sites found with Internet searches, including www.json.org.

XML

Back in the 1990s, when web browsers and the World Wide Web (WWW) were first created, web pages primarily used Hypertext Markup Language (HTML) to define web pages. As a markup language, HTML defined how to add the text or a web page to a file and then add "markup"—additional text to denote formatting details for the text that should be displayed. For instance, the markup included codes for headings, font types, sizes, colors, hyperlinks, and so on.

The eXtensible Markup Language (XML) came later to make some improvements for earlier markup languages. In particular, over time web pages became more and more dynamic, and to make the pages dynamic, the files needed to store variables whose values could be changed and replaced over time by the web server. To define variables to be substituted into a web page, the world needed a markup language that could define data variables. XML defines a markup language that has many features to define variables, values, and data structures.

Over time, XML has grown beyond its original use as a markup language. XML's features also make it a useful general data serialization language, and it is used as such today.

Comparing XML to JSON, both attempt to be human readable, but with XML being a little more challenging to read for the average person. For instance, like HTML, XML uses beginning and ending tags for each variable, as seen in Example 18-4. In the highlighted line in the example, the <macAddress> and </macAddress> tags denote a variable name, with the value sitting between the tags.

Example 18-4 *JSON Output from a REST API Call*

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <response>
      <family>Switches and Hubs</family>
      <hostname>cat_9k_1</hostname>
      <interfaceCount>41</interfaceCount>
      <lineCardCount>2</lineCardCount>
      <macAddress>f8:7b:20:67:62:80</macAddress>
      <managementIpAddress>10.10.22.66</managementIpAddress>
      <role>ACCESS</role>
      <serialNumber>FCW2136L0AK</serialNumber>
      <series>Cisco Catalyst 9300 Series Switches</series>
      <softwareType>IOS-XE</softwareType>
      <softwareVersion>16.6.1</softwareVersion>
      <type>Cisco Catalyst 9300 Switch</type>
      <upTime>17 days, 22:51:04.26</upTime>
   </response>
</root>
```

YAML

YAML Ain't Markup Language (YAML) has a clever recursive name, but the name does tell us something. YAML does not attempt to define markup details (while XML does). Instead, YAML focuses on the data model (structure) details. YAML also strives to be clean and simple: of the data serialization/modeling languages listed here, YAML is easily the easiest to read for anyone new to data models.

Ansible, one of the topics in Chapter 19, "Understanding Ansible, Puppet, and Chef," makes extensive use of YAML files. Example 18-5 shows a brief sample. And to make the point about readability, even if you have no idea what Ansible does, you can guess at some of the functions just reading the file. (Note that YAML denotes variables in double curly brackets: {{ }}.)

Example 18-5 *YML File Used by Ansible*

```
---
# This comment line is a place to document this Playbook
- name: Get IOS Facts
  hosts: mylab
  vars:
    cli:
      host: "{{ ansible_host }}"
      username: "{{ username }}"
      password: "{{ password }}"

  tasks:
    - ios_facts:
        gather_subset: all
        provider: "{{ cli }}"
```

Summary of Data Serialization

As an easy reference, Table 18-3 summarizes the data serialization languages mentioned in this section, along with some key facts.

Key Topic

Table 18-3 Comparing Data Modeling Languages

Acronym	Name	Origin/Definition	Central Purpose	Common Use
JSON	JavaScript Object Notation	JavaScript (JS) language; RFC 8259	General data modeling and serialization	REST APIs
XML	eXtensible Markup Language	World Wide Web Consortium (W3C.org)	Data-focused text markup that allows data modeling	REST APIs, Web pages
YAML	YAML Ain't Markup Language	YAML.org	General data modeling	Ansible

Interpreting JSON

Cisco includes one exam topic in the current CCNA 200-301 blueprint that mentions JSON:

6.7 Interpret JSON encoded data

You can think of that skill and task with two major branches. First, even ignoring the syntax and special characters, anyone who knows the topic can probably make intelligent guesses about the meaning of many of the key:value pairs. For example, without knowing anything about JSON syntax, you could probably determine from your prior knowledge of Cisco routers and switches that the JSON in Example 18-6 lists two devices (maybe their host-names) and a list of interfaces on each device.

Example 18-6 *Simple JSON That Lists a Router's Interfaces*

```
{
    "R1": ["GigabitEthernet0/0", "GigabitEthernet0/1", "GigabitEthernet0/2/0"],
    "R2": ["GigabitEthernet1/0", "GigabitEthernet1/1", "GigabitEthernet0/3/0"]
}
```

Honestly, you probably already know everything needed to do this kind of intelligent guessing. However, to perform the second type of task, where you analyze the JSON data to find the data structures, including objects, lists, and key:value pairs, you need to know a bit more about JSON syntax. This final topic in the chapter gives you the basic rules, with some advice on how to break down JSON data.

Interpreting JSON Key:Value Pairs

First, consider these rules about key:value pairs in JSON, which you can think of as individual variable names and their values:

Key Topic

- **Key:Value Pair:** Each and every colon identifies one key:value pair, with the key before the colon and the value after the colon.
- **Key:** Text, inside double quotes, before the colon, used as the name that references a value.

- **Value:** The item after the colon that represents the value of the key, which can be
  - **Text:** Listed in double quotes.
  - **Numeric:** Listed without quotes.
  - **Array:** A special value (more details later).
  - **Object:** A special value (more details later)
- **Multiple Pairs:** When listing multiple key:value pairs, separate the pairs with a comma at the end of each pair (except the last pair).

To work through some of these rules, consider Example 18-7's JSON data, focusing on the three key:value pairs. The text after the example will analyze the example.

Example 18-7 *One JSON Object (Dictionary) with Three Key:Value Pairs*

```
{
    "1stbest": "Messi",
    "2ndbest": "Ronaldo",
    "3rdbest": "Pele"
}
```

As an approach, just find each colon, and look for the quoted string just before each colon. Those are the keys ("1stbest", "2ndbest", and "3rdbest".) Then look to the right of each colon to find their matching values. You can know all three values are text values because JSON lists the values within double quotes.

As for other special characters, note the commas and the curly brackets. The first two key:value pairs end with a comma, meaning that another key:value pair should follow. The curly brackets that begin and end the JSON data denote a single JSON object (one pair of curly brackets, so one object). JSON files, and JSON data exchanged over an API, exist first as a JSON object, with an opening (left) and closing (right) curly bracket as shown.

Interpreting JSON Objects and Arrays

To communicate data structures beyond a key:value pair with a simple value, JSON uses JSON objects and JSON arrays. Objects can be somewhat flexible, but in most uses, they act like a dictionary. Arrays list a series of values.

NOTE Python, the most common language to use for network automation, converts JSON objects to Python dictionaries, and JSON arrays to Python lists. For general conversation, many people refer to the JSON structures as dictionaries and lists rather than as objects and arrays.

To begin, consider this set of rules about how to interpret the syntax for JSON objects and arrays:

Key Topic

- **{ } - Object:** A series of key:value pairs enclosed in a matched pair of curly brackets, with an opening left curly bracket and its matching right curly bracket.
- **[ ] - Array:** A series of values (not key:value pairs) enclosed in a matched pair of square brackets, with an opening left square bracket and its matching right square bracket.

- **Key:value pairs inside objects:** All key:value pairs inside an object conform to the earlier rules for key:value pairs.
- **Values inside arrays:** All values conform to the earlier rules for formatting values (for example, double quotes around text, no quotes around numbers).

Example 18-8 shows a single array in JSON format. Notice the JSON data begins with a [and then lists three text values (the values could have been a mix of values). It then ends with a].

Example 18-8 *A JSON Snippet Showing a Single JSON Array (List)*

```
[
     "Messi",
     "Ronaldo",
     "Dybala"
]
```


While Example 18-8 shows only the array itself, JSON arrays can be used as a value in any key:value pair. Figure 18-12 does just that, shown in a graphic to allow easier highlighting of the arrays and object. The JSON text in the figure includes two arrays (lists) as values (each found just after a colon, indicating they are values).

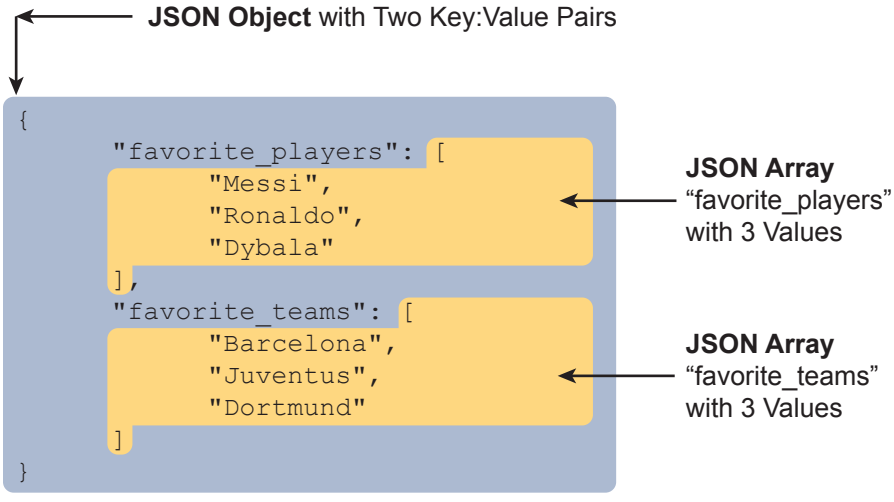


Figure 18-12 *Accurate/Complete JSON Data with One Object, Two Keys, Two JSON List Values*

Now think about the entire structure of the JSON data in Figure 18-12. It has a matched pair of curly brackets to begin and end the text, encapsulating one object. That object contains two colons, so there are two key:value pairs inside the object. When you think about the broader structure, as depicted in Figure 18-13, this JSON file has one JSON object, itself with two key:value pairs. (Note that Figure 18-13 does NOT show correct JSON syntax for the lists; it instead is intended to make sure you see the structure of the one object and its two key:value pairs.)

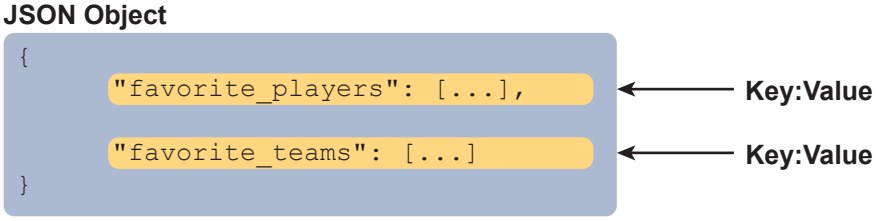


Figure 18-13 *Structural Representation of Figure 18-13's Primary Object and Two Key:Value Pairs*

To drive home the idea of how to find JSON objects, consider the example shown in Figure 18-14. This figure shows correct JSON syntax. It has the following:

- There is one object for the entire set because it begins and ends with curly braces.
- The outer object has two keys (Wendells_favorites and interface_config).
- The value of each key:value pair is another object (each with curly braces and three key:value pairs).

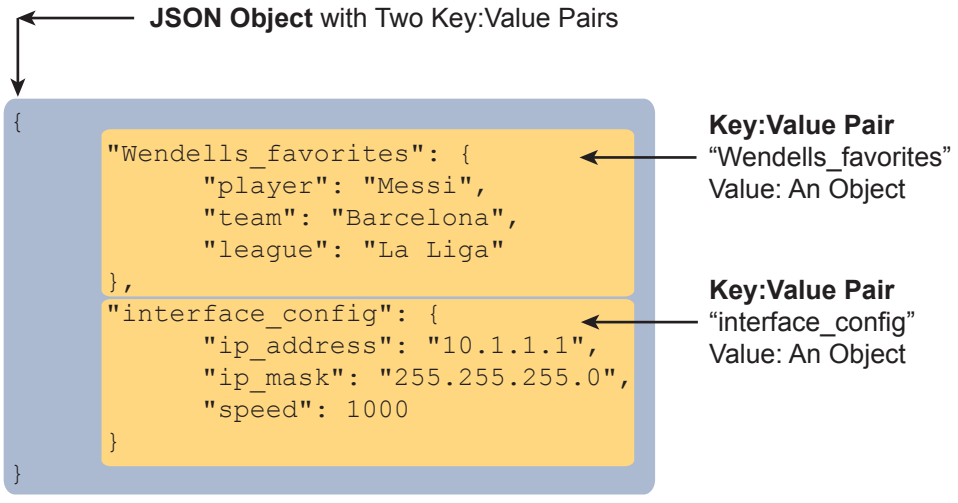


Figure 18-14 *A JSON Object, with Two Key:Value Pairs, Each Value Another Object*

The JSON example in Figure 18-14 shows how JSON can nest objects and arrays; that is, JSON puts one object or array inside another. Much of the JSON output you will see as you learn more and more about network automation will include JSON data with nested arrays and objects.

Minified and Beautified JSON

So far, all the JSON examples show lots of empty space. JSON allows for whitespace, or not, depending on your needs. For humans, reading JSON can be a lot easier with the text organized with space and aligned. For instance, having the matched opening and closing brackets sit at the same left-offset makes it much easier to find which brackets go with which.

When stored in a file or sent in a network, JSON does not use whitespace. For instance, earlier in this section, Example 18-7 showed one JSON object with three key:value pairs, with

whitespace, taking five lines. However, stored in a file, or sent over a network, the JSON would look like the following:

```
{"1stbest": "Messi", "2ndbest": "Ronaldo", "3rdbest": "Pele"}
```

Most of the tools you might use when working with JSON will let you toggle from a pretty format (good for humans) to a raw format (good for computers). You might see the pretty version literally called *pretty*, or *beautified*, or *spaced*, while the version with no extra whitespace might be called *minified* or *raw*.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 18-4 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 18-4 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review memory tables		Website
Practice Editing JSON		Website

Review All the Key Topics

Key Topic

Table 18-5 Key Topics for Chapter 18

Key Topic Element	Description	Page Number
List	Attributes of REST APIs	409
List	The meaning of the CRUD acronym	413
Table 18-2	A comparison of CRUD actions and HTTP verbs	414
Figure 18-8	Components of a URI	416
Figure 18-11	The process of sending JSON data over a REST API	420
Table 18-3	A comparison of JSON, XML, and YAML	423
List	JSON rules related to key:value pairs	423
List	JSON rules for arrays and objects	424

Key Terms You Should Know

REpresentational State Transfer (REST), REST API, stateless, cacheable, CRUD, list variable, dictionary variable, URI path (resource), URI query (parameters), key:value pair, data serialization language, JSON (JavaScript Object Notation), XML (eXtensible Markup Language), YAML (YAML Ain't Markup Language), JSON object, JSON array

CHAPTER 19

Understanding Ansible, Puppet, and Chef

This chapter covers the following exam topics:

6.0 Automation and Programmability

6.6 Recognize the capabilities of configuration mechanisms Puppet, Chef, and Ansible

By now, you have seen how to use the IOS CLI to configure routers and switches. To configure using the CLI, you get into configuration mode, issue configuration commands (which change the running-config file), and eventually leave configuration mode. If you decide to keep those changes, you save the configuration to the startup-config file using the **copy running-config startup-config** command. Next time the router or switch boots, the device loads the startup-config file into RAM as the running-config. Simple enough.

This chapter discusses tools for configuration management that replaces that per-device configuration process. To even imagine what these tools do first requires you to make a leap of imagination to the everyday world of a network engineer at a medium to large enterprise. In a real working network, managing the configuration of the many networking devices creates challenges. Those challenges can be addressed using that same old "use configuration mode on each device" process, plus with hard work, attention to detail, and good operational practices. However, that manual per-device process becomes more and more difficult for a variety of reasons, so at some point, enterprises turn to automated configuration management tools to provide better results.

The first section of this chapter takes a generalized look at the issues of configuration management at scale along with some of the solutions to those problems. The second major section then details each of three configuration management tools—Ansible, Puppet, and Chef—to define some of the features and terms used with each. By the end of the chapter, you should be able to see some of the reasons why these automated configuration management tools have a role in modern networks and enough context to understand as you pick one to investigate for further reading.

"Do I Know This Already?" Quiz

Take the quiz (either here or use the PTP software) if you want to use the score to help you decide how much time to spend on this chapter. The letter answers are listed at the bottom of the page following the quiz. Appendix C, found both at the end of the book as well as on the companion website, includes both the answers and explanations. You can also find both answers and explanations in the PTP testing software.

Table 19-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Device Configuration Challenges and Solutions	1–3
Ansible, Puppet, and Chef Basics	4, 5

1. Which answer best describes the meaning of the term *configuration drift*?
   a. Changes to a single device's configuration over time versus that single device's original configuration
   b. Larger and larger sections of unnecessary configuration in a device
   c. Changes to a single device's configuration over time versus other devices that have the same role
   d. Differences in device configuration versus a centralized backup copy
2. An enterprise moves away from manual configuration methods, making changes by editing centralized configuration files. Which answers list an issue solved by using a version control system with those centralized files? (Choose two answers.)
   a. The ability to find which engineer changed the central configuration file on a date/time
   b. The ability to find the details of what changed in the configuration file over time
   c. The ability to use a template with per-device variables to create configurations
   d. The ability to recognize configuration drift in a device and notify the staff
3. Configuration monitoring (also called configuration enforcement) by a configuration management tool generally solves which problem?
   a. Tracking the identity of individuals who changed files, along with which files they changed
   b. Listing differences between a former and current configuration
   c. Testing a configuration change to determine whether it will be rejected or not when implemented
   d. Finding instances of configuration drift
4. Which of the following configuration management tools uses a push model to configure network devices?
   a. Ansible
   b. Puppet
   c. Chef
   d. None use a push model

5. Which of the following answers list a correct combination of configuration management tool and the term used for one of its primary configuration files? (Choose two answers.)

   a. Ansible manifest
   b. Puppet manifest
   c. Chef recipe
   d. Ansible recipe

Foundation Topics

Device Configuration Challenges and Solutions

Think about any production network. What defines the exact intended configuration of each device in a production network? Is it the running-config as it exists right now or the startup-config before any recent changes were made or the startup-config from last month? Could one engineer change the device configuration so that it drifts away from that ideal, with the rest of the staff not knowing? What process, if any, might discover the configuration drift? And even with changes agreed upon by all, how do you know who changed the configuration, when, and specifically what changed?

Traditionally, CCNA teaches us how to configure one device using the **configure terminal** command to reach configuration mode, which changes the running-config file, and how to save that running-config file to the startup-config file. That manual process provides no means to answer any of the legitimate questions posed in the first paragraph; however, for many enterprises, those questions (and others) need answers, both consistent and accurate.

Not every company reaches the size to want to do something more with configuration management. Companies with one network engineer might do well enough managing device configurations, especially if the network device configurations do not change often. However, as a company moves to multiple network engineers and grows the numbers of devices and types of devices, with higher rates of configuration change, manual configuration management has problems.

This section begins by discussing a few of these kinds of configuration management issues so that you begin to understand why enterprises need more than good people and good practices to deal with device configuration. The rest of the section then details some of the features you can find in automated configuration management tools.

Configuration Drift

Consider the story of an enterprise of a size to need two network engineers, Alice and Bob. They both have experience and work well together. But the network configurations have grown beyond what any one person can know from memory, and with two network engineers, they may remember different details or even disagree on occasion.

One night at 1 a.m., Bob gets a call about an issue. He gets into the network from his laptop and resolves the problem with a small configuration change to branch office router BR22. Alice, the senior engineer, gets a different 4 a.m. call about another issue and makes a change to branch office router BR33.

The next day gets busy, and neither Alice nor Bob mentions the changes they made. They both follow procedures and document the changes in their change management system, which lists the details of every change. But they both get busy, the topic never comes up, and neither mentions the changes to each for months.

The story shows how configuration drift can occur—an effect in which the configuration drifts away from the intended configuration over time. Alice and Bob probably agree to what a standard branch office router configuration ought to look like, but they both made an exception to that configuration to fix a problem, causing configuration drift. Figure 19-1 shows the basic idea, with those two branch routers now with slightly different configurations than the other branch routers.

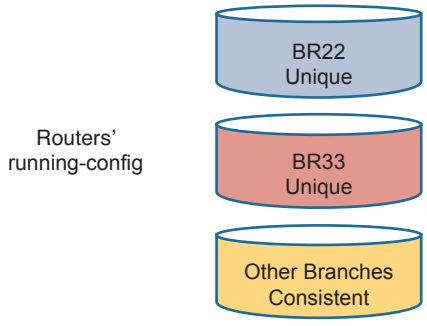


Figure 19-1 *Configuration Drift in Branch Routers BR22 and BR33*

Configuration drift becomes a much bigger problem if using only traditional manual configuration tools. For instance:

Key Topic

- The on-device manual configuration process does not track change history: which lines changed, what changed on each line, what old configuration was removed, who changed the configuration, when each change was made.
- External systems used by good systems management processes, like trouble ticketing and change management software, may record details. However, those sit outside the configuration and require analysis to figure out what changed. They also rely on humans to follow the operational processes consistently and correctly; otherwise, an engineer cannot find the entire history of changes to a configuration.
- Referring to historical data in change management systems works poorly if a device has gone through multiple configuration changes over a period of time.

Centralized Configuration Files and Version Control

The manual per-device configuration model makes great sense for one person managing one device. With that model, the one network engineer can use the on-device startup-config as the intended ideal configuration. If a change is needed, the engineer gets into configuration mode and updates the running-config until happy with the change. Then the engineer saves a copy to startup-config as the now-current ideal config for the device.

The per-device manual configuration model does not work as well for larger networks, with hundreds or even thousands of network devices, with multiple network engineers. For instance, if the team thinks of the startup-config of each device as the ideal configuration, if one team member changes the configuration (like Alice and Bob each did in the earlier

story), no records exist about the change. The config file does not show what changed, when it changed, or who changed it, and the process does not notify the entire team about the change.

As a first step toward better configuration management, many medium to large enterprises store configurations in a central location. At first, storing files centrally may be a simple effort to keep backup copies of each device's configuration. They would, of course, place the files in a shared folder accessible to the entire network team, as shown in Figure 19-2.

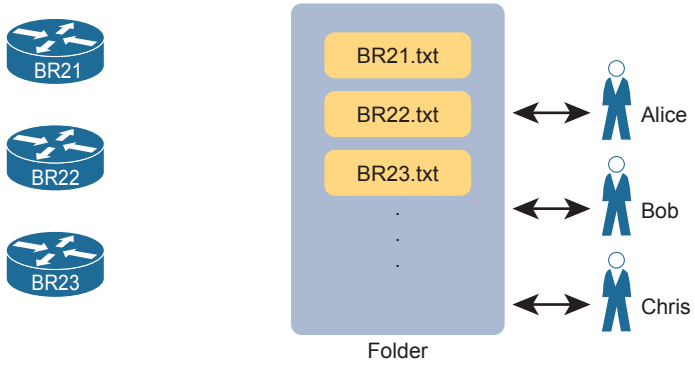


Figure 19-2 *Copying Device Configurations to a Central Location*

Which configuration file is the single source of truth in this model? The configuration files still exist on each device, but now they also exist on a centralized server, and engineers could change the on-device configuration as well as the text files on the server. For instance, if the copy of BR21's configuration on the device differs from the file on the centralized server, which should be considered as correct, ideal, the truth about what the team intends for this device?

In practice, companies take both approaches. In some cases, companies continue to use the on-device configuration files as the source of truth, with the centralized configuration files treated as backup copies in case the device fails and must be replaced. However, other enterprises make the transition to treat the files on the server as the single source of truth about each device's configuration. When using the centralized file as the source of truth, the engineers can take advantage of many configuration management tools and actually manage the configurations more easily and with more accuracy.

For example, configuration management tools use version control software to track the changes to centralized configuration files, noting who changes a file, what lines and specific characters changed, when the change occurred, and so on. The tools also allow you to compare the differences between versions of the files over time, as shown in Figure 19-3.

Answers to the "Do I Know This Already?" quiz:

1 C **2** A, B **3** D **4** A **5** B, C

Key Topic

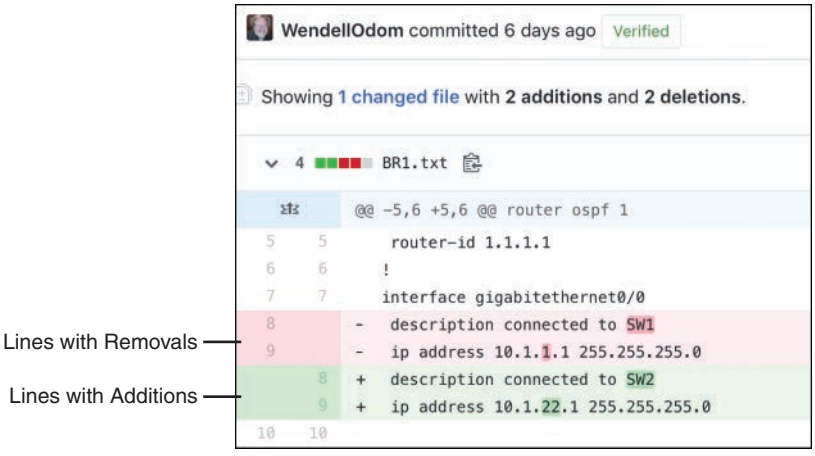


Figure 19-3 *Showing File Differences in GitHub*

The figure shows a sample of a comparison between two versions of a configuration file. The upper two highlighted lines, with the minus signs, show the lines that were changed, while the two lower highlighted lines, with the plus signs, show the new versions of each line.

Version control software solves many of the problems with the lack of change tracking within the devices themselves. Figure 19-3 shows output from a popular software-as-a-service site called GitHub (www.github.com). GitHub offers free and paid accounts, and it uses open-source software (Git) to perform the version control functions.

Configuration Monitoring and Enforcement

With a version control system and a convention of storing the configuration files in a central location, a network team can do a much better job of tracking changes and answering the who, what, and when of knowing what changed in every device's configuration. However, using that model then introduces other challenges—challenges that can be best solved by also using an automated configuration management tool.

With this new model, engineers should make changes by editing the associated configuration files in the centralized repository. The configuration management tool can then be directed to copy or apply the configuration to the device, as shown in Figure 19-4. After that process completes, the central config file and the device's running-config (and startup-config) should be identical.

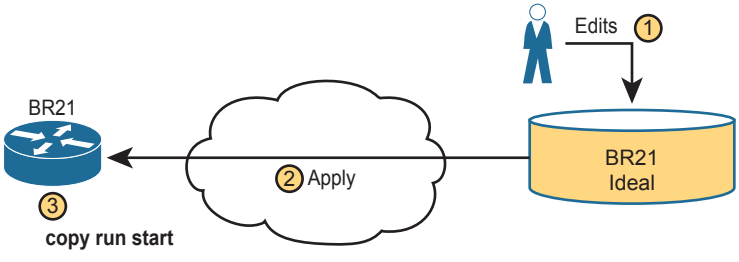


Figure 19-4 *Pushing Centralized Configuration to a Remote Device*

Using the model shown in Figure 19-4 still has dangers. For instance, the network engineers should make changes by using the configuration management tools, but they still have the ability to log in to each device and make manual changes on each device. So, while the idea of using a configuration management tool with a centralized repository of config files sounds appealing, eventually someone will change the devices directly. Former correct configuration changes might be overwritten, and made incorrect, by future changes. In other words, eventually, some configuration drift can occur.

Configuration management tools can monitor device configurations to discover when the device configuration differs from the intended ideal configuration, and then either reconfigure the device or notify the network engineering staff to make the change. This feature might be called *configuration monitoring* or *configuration enforcement*, particularly if the tool automatically changes the device configuration.

Figure 19-5 shows the general idea behind configuration monitoring. The automated configuration management software asks for a copy of the device's running-config file, as shown in steps 1 and 2. At step 3, the config management software compares the ideal config file with the just-arrived running-config file to check whether they have any differences (configuration drift). Per the configuration of the tool, it either fixes the configuration or notifies the staff about the configuration drift.

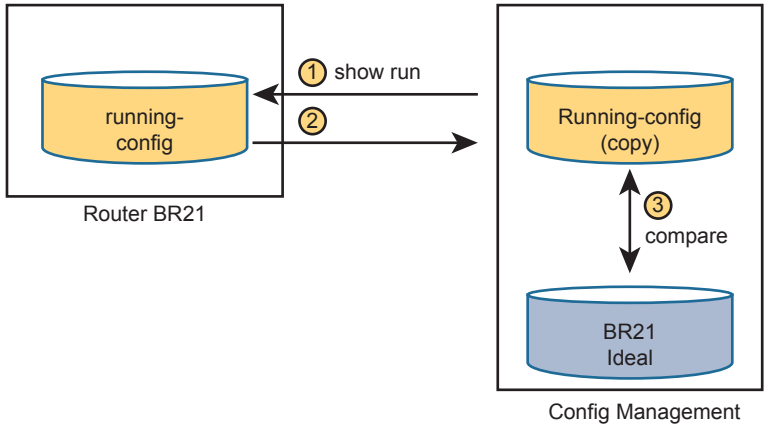


Figure 19-5 *Configuration Monitoring*

Configuration Provisioning

Configuration provisioning refers to how to provision or deploy changes to the configuration once made by changing files in the configuration management system. As one of the primary functions of a configuration management tool, you would likely see features like these:

- The core function to implement configuration changes in one device after someone has edited the device's centralized configuration file
- The ability to choose which subset of devices to configure: all devices, types with a given attribute (such as those of a particular role), or just one device, based on attributes and logic

- The ability to determine if each change was accepted or rejected, and to use logic to react differently in each case depending on the result
- For each change, the ability to revert to the original configuration if even one configuration command is rejected on a device
- The ability to validate the change now (without actually making the change) to determine whether the change will work or not when attempted
- The ability to check the configuration after the process completes to confirm that the configuration management tool's intended configuration does match the device's configuration
- The ability to use logic to choose whether to save the running-config to startup-config or not
- The ability to represent configuration files as templates and variables so that devices with similar roles can use the same template but with different values
- The ability to store the logic steps in a file, scheduled to execute, so that the changes can be implemented by the automation tool without the engineer being present

The list could go further, but these features outline some of the major features included in all of the configuration management tools discussed in this chapter. Most of the items in the list revolve around editing the central configuration file for a device. However, the tools have many more features, so you have more work to do to plan and implement how they work. The next few pages focus on giving a few more details about the last two items in the list.

Configuration Templates and Variables

Think about the roles filled by networking devices in an enterprise. Focusing on routers for a moment, routers often connect to both the WAN and one or more LANs. You might have a small number of larger routers connected to the WAN at large sites, with enough power to handle larger packet rates. Smaller sites, like branch offices, might have small routers, maybe with a single WAN interface and a single LAN interface; however, you might have a large number of those small branch routers in the network.

For any set of devices in the same role, the configurations are likely similar. For instance, a set of branch office routers might all have the exact same configuration for some IP services, like NTP or SNMP. If using OSPF interface configuration, routers in the same OSPF area and with identical interface IDs could have identical OSPF configuration.

For instance, Example 19-1 shows a configuration excerpt from a branch router, with the unique parts of the configuration highlighted. All the unhighlighted portions could be the same on all the other branch office routers of the same model (with the same interface numbers). An enterprise might have dozens or hundreds of branch routers with nearly identical configuration.

Example 19-1 *Router BR1 Configuration, with Unique Values Highlighted*

```
hostname BR1
!
interface GigabitEthernet0/0
 ip address 10.1.1.1 255.255.255.0
 ip ospf 1 area 11
```

```
!
interface GigabitEthernet0/1
!
interface GigabitEthernet0/1/0
 ip address 10.1.12.1 255.255.255.0
 ip ospf 1 area 11
!
router ospf 1
 router-id 1.1.1.1
```

Configuration management tools can separate the components of a configuration into the parts in common to all devices in that role (the template) versus the parts unique to any one device (the variables). Engineers can then edit the standard template file for a device role as a separate file than each device's variable file. The configuration management tool can then process the template and variables to create the ideal configuration file for each device, as shown in Figure 19-6, which shows the configuration files being built for branch routers BR21, BR22, and BR23.

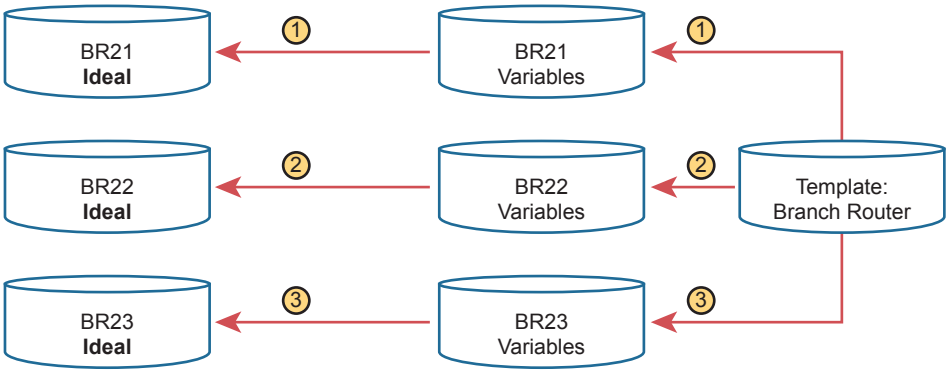


Figure 19-6 *Concept: Configuration Templates and Variables*

To give a little more insight, Example 19-2 shows a template file that could be used by Ansible for the configuration shown in Example 19-1. Each tool specifies what language to use for each type of file, with Ansible using the Jinja2 language for templates. The template mimics the configuration in Example 19-1, except for placing variable names inside sets of double curly brackets.

Key Topic

Example 19-2 *Jinja2 Template with Variables Based on Example 19-1*

```
hostname {{hostname}}
!
interface GigabitEthernet0/0
 ip address {{address1}} {{mask1}}
 ip ospf {{OSPF_PID}} area {{area}}
!
interface GigabitEthernet0/1
!
interface GigabitEthernet0/1/0
```

```
 ip address {{address2}} {{mask2}}
 ip ospf {{OSPF_PID}} area {{area}}
!
router ospf {{OSPF_PID}}
 router-id {{RID}}
```

To supply the values for a device, Ansible calls for defining variable files using YAML, as shown in Example 19-3. The file shows the syntax for defining the variables shown in the complete configuration in Example 19-1, but now defined as variables.

Example 19-3 *YAML Variables File Based on Example 19-2*

```
---
hostname: BR1
address1: 10.1.1.1
mask1: 255.255.255.0
address2: 10.1.12.1
mask2: 255.255.255.0
RID: 1.1.1.1
area: '11'
OSPF_PID: '1'
```


The configuration management system processes a template plus all related variables to produce the intended configuration for a device. For instance, the engineer would create and edit one template file that looks like Example 19-2 and then create and edit one variable file like Example 19-3 for each branch office router. Ansible would process the files to create complete configuration files like the text shown in Example 19-1.

It might seem like extra work to separate configurations into a template and variables, but using templates has some big advantages. In particular:

Key Topic

- Templates increase the focus on having a standard configuration for each device role, helping to avoid snowflakes (uniquely configured devices).
- New devices with an existing role can be deployed easily by simply copying an existing per-device variable file and changing the values.
- Templates allow for easier troubleshooting because troubleshooting issues with one standard template should find and fix issues with all devices that use the same template.
- Tracking the file versions for the template versus the variables files allows for easier troubleshooting as well. Issues with a device can be investigated to find changes in the device's settings separately from the standard configuration template.

Files That Control Configuration Automation

Configuration management tools also provide different methods to define logic and processes that tell the tool what changes to make, to which devices, and when. For instance, an engineer could direct a tool to make changes during a weekend change window. That same logic could specify a subset of the devices. It could also detail steps to verify the change before and after the change is attempted, and how to notify the engineers if an issue occurs.

Interestingly, you can do a lot of the logic without knowing how to program. Each tool uses a language of some kind that engineers use to define the action steps, often a language

defined by that company (a domain-specific language). But they make the languages to be straightforward, and they are generally mush easier to learn than programming languages. Configuration management tools also enable you to extend the action steps beyond what can be done in the toolset by using a general programming language. Figure 19-7 summarizes the files you could see in any of the configuration management tools.

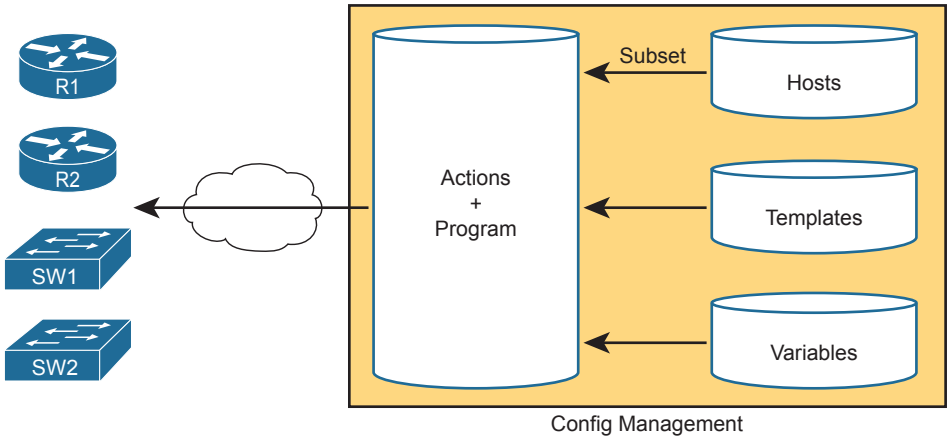


Figure 19-7 *Important Files Used by Configuration Management Tools*

Ansible, Puppet, and Chef Basics

This chapter focuses on one exam topic that asks about the capabilities of three configuration management tools: Ansible, Puppet, and Chef. The first major section of the chapter describes the capabilities of all three (and other) configuration management tools. This second major section examines a few of the features of each tool, focusing on terminology and major capabilities.

Ansible, Puppet, and Chef are software packages. You can purchase each tool, with variations on which package. However, they all also have different free options that allow you to download and learn about the tools, although you might need to run a Linux guest because some of the tools do not run in a Windows OS.

As for the names, most people use the words *Ansible*, *Puppet*, and *Chef* to refer to the companies as well as their primary configuration management products. All three emerged as part of the transition from hardware-based servers to virtualized servers, which greatly increased the number of servers and created the need for software automation to create, configure, and remove VMs. All three produce one or more configuration management software products that have become synonymous with their companies in many ways. (This chapter follows that convention, for the most part ignoring exact product names, and referring to products and software simply as Ansible, Puppet, and Chef.)

Next, on to some details about each.

Ansible

To use Ansible (www.ansible.com), you need to install Ansible on some computer: Mac, Linux, or a Linux VM on a Windows host. You can use the free open-source version or use the paid Ansible Tower server version.

Once it is installed, you create several text files, such as the following:

- **Playbooks:** These files provide actions and logic about what Ansible should do.
- **Inventory:** These files provide device hostnames along with information about each device, like device roles, so Ansible can perform functions for subsets of the inventory
- **Templates:** Using Jinja2 language, the templates represent a device's configuration but with variables (see Example 19-2).
- **Variables:** Using YAML, a file can list variables that Ansible will substitute into templates (see Example 19-3).

As far as how Ansible works for managing network devices, it uses an agentless architecture. That means Ansible does not rely on any code (agent) running on the network device. Instead, Ansible relies on features typical in network devices, namely SSH and/or NETCONF, to make changes and extract information. When using SSH, the Ansible control node actually makes changes to the device like any other SSH user would do, but doing the work with Ansible code, rather than with a human.

Ansible can be described as using a push model (per Figure 19-8) rather than a pull model (like Puppet and Chef). After installing Ansible, an engineer needs to create and edit all the various Ansible files, including an Ansible playbook. Then the engineer runs the playbook, which tells Ansible to perform the steps. Those steps can include configuring one or more devices per the various files (step 3), with the control node seen as pushing the configuration to the device.

Key Topic

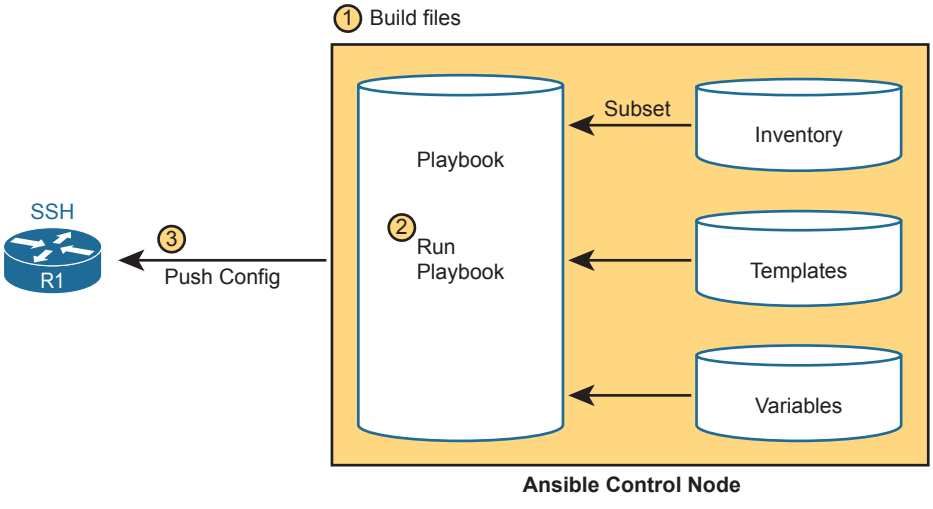


Figure 19-8 *Ansible Push Model*

As with all the tools, Ansible can do both configuration provisioning (configuring devices after changes are made in the files) and configuration monitoring (checking to find out whether the device config matches the ideal configuration on the control node). However, Ansible's architecture more naturally fits with configuration provisioning, as seen in the figure. To do configuration monitoring, Ansible uses logic modules that detect and list configuration differences, after which the playbook defines what action to take (reconfigure or notify).

Puppet

To use Puppet (www.puppet.com), like Ansible, begin by installing Puppet on a Linux host. You can install it on your own Linux host, but for production purposes, you will normally install it on a Linux server called a Puppet master. As with Ansible, you can use a free open-source version with paid versions available. You can get started learning Puppet without a separate server for learning and testing.

Once installed, Puppet also uses several important text files with different components, such as the following:

- **Manifest:** This is a human-readable text file on the Puppet master, using a language defined by Puppet, used to define the desired configuration state of a device.
- **Resource, Class, Module:** These terms refer to components of the manifest, with the largest component (module) being composed of smaller classes, which are in turn composed of resources.
- **Templates:** Using a Puppet domain-specific language, these files allow Puppet to generate manifests (and modules, classes, and resources) by substituting variables into the template.

One way to think about the differences between Ansible's versus Puppet's approach is that Ansible's playbooks use an imperative language, whereas Puppet uses a declarative language. For instance, with Ansible, the playbook will list tasks and choices based on those results, like "Configure all branch routers in these locations, and if errors occur for any device, do these extra tasks for that device." Puppet manifests instead declare the end state that a device should have: "This branch router should have the configuration in this file by the end of the process." The manifest, built by the engineer, defines the end state, and Puppet has the job to cause the device to have that configuration, without being told the specific set of steps to take.

Puppet typically uses an agent-based architecture for network device support. Some network devices enable Puppet support via an on-device agent—think of it as another feature configurable on the device. However, not every Cisco OS supports Puppet agents, so Puppet solves that problem using a proxy agent running on some external host (called agentless operation). The external agent then uses SSH to communicate with the network device, as shown in Figure 19-9.

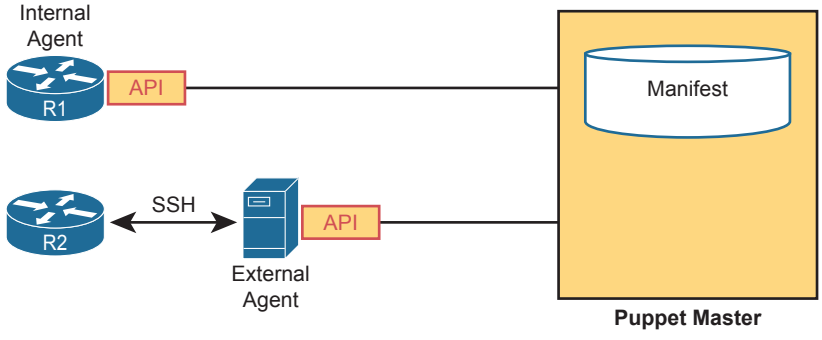


Figure 19-9 *Agent-based and Agentless Operation for Puppet*

NOTE Per Puppet's website, Puppet supports both an agent-based and agentless architecture, with the agentless architecture being the case of using an agent external to the network device, as shown in the lower part of Figure 19-9.

Armed with a manifest that declares something like "This device should have this configuration state," Puppet uses a pull model to make that configuration appear in the device, as shown in Figure 19-10. Once installed, these steps occur:

Step 1. The engineer creates and edits all the files on the Puppet server.

Step 2. The engineer configures and enables the on-device agent or a proxy agent for each device.

Step 3. The agent pulls manifest details from the server, which tells the agent what its configuration should be.

Step 4. If the agent device's configuration should be updated, the Puppet agent performs additional pulls to get all required detail, with the agent updating the device configuration.

Key Topic

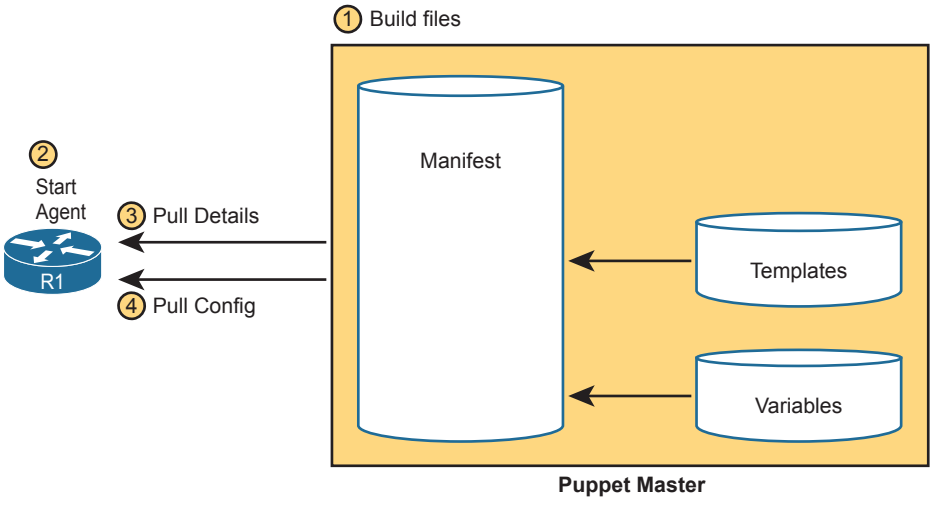


Figure 19-10 *Pull Model with Puppet*

Chef

Chef (www.chef.io), as with Ansible and Puppet, exists as software packages you install and run. Chef (the company) offers several products, with Chef Automate being the product that most people refer to simply as Chef. As with Puppet, in production you probably run Chef as a server (called server-client mode), with multiple Chef workstations used by the engineering staff to build Chef files that are stored on the Chef server. However, you can also run Chef in standalone mode (called Chef Zero), which is helpful when you're just getting started and learning in the lab.

Once Chef is installed, you create several text files with different components, like the following:

- **Resource:** The configuration objects whose state is managed by Chef; for instance, a set of configuration commands for a network device—analogous to the ingredients in a recipe in a cookbook
- **Recipe:** The Chef logic applied to resources to determine when, how, and whether to act against the resources—analogous to a recipe in a cookbook
- **Cookbooks:** A set of recipes about the same kinds of work, grouped together for easier management and sharing
- **Runlist:** An ordered list of recipes that should be run against a given device

Chef uses an architecture similar to Puppet. For network devices, each managed device (called a Chef node or Chef client) runs an agent. The agent performs configuration monitoring in that the client pulls recipes and resources from the Chef server and then adjusts its configuration to stay in sync with the details in those recipes and runlists. Note however that Chef requires on-device Chef client code, and many Cisco devices do not support a Chef client, so you will likely see more use of Ansible and Puppet for Cisco device configuration management.

Summary of Configuration Management Tools

All three of the configuration management tools listed here have a good base of users and different strengths. As for their use for managing network device configuration, Ansible appears to have the most interest, then Puppet, and then Chef. Ansible's agentless architecture and the use of SSH provides support for a wide range of Cisco devices. Puppet's agentless model also creates wide support for Cisco devices.

Table 19-2 summarizes a few of the most common ideas about each of the three automated configuration management tools. Note that the column for Puppet assumes an on-device agent.

Key Topic

Table 19-2 *Comparing Ansible, Puppet, and Chef*

Action	Ansible	Puppet	Chef
Term for the file that lists actions	Playbook	Manifest	Recipe, Runlist
Protocol to network device	SSH, NETCONF	HTTP (REST)	HTTP (REST)
Uses agent or agentless model	Agentless	Agent*	Agent
Push or pull model	Push	Pull	Pull

* Puppet can use an in-device agent or an external proxy agent for network devices.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 19-3 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 19-3 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Repeat DIKTA questions		Book, PTP
Review memory table		Book, website
Do DevNet Labs		DevNet

Review All the Key Topics

Key Topic

Table 19-4 Key Topics for Chapter 19

Key Topic Element	Description	Page Number
List	Issues that arise from configuration drift	431
Figure 19-3	Sample of showing router configuration file differences with GitHub	433
Figure 19-5	Basic configuration monitoring concepts.	434
List	Primary functions of a configuration management tool	434
Example 19-2	Sample Jinja2 Ansible template	436
List	Advantages of using configuration templates	437
Figure 19-8	Ansible's push model and other features	439
Figure 19-10	Puppet's pull model and other features	441
Table 19-2	Summary of configuration management features and terms	442

Key Terms You Should Know

configuration monitoring, configuration provisioning, configuration drift, configuration management tool, Git, Ansible, Puppet, Chef, configuration template, push model, pull model, agent-based architecture, agentless architecture, Ansible playbook, Puppet manifest, Chef recipe

Do DevNet Labs

Cisco's DevNet site (https://developer.cisco.com)—a free site—includes lab environments and exercises. You can learn a lot about configuration management and Ansible in particular with a few of the lab tracks on the DevNet site (at the time this book was published). Refer to the "Chapter Review" section of the companion website for links to some good labs, or just go to https://developer.cisco.com and search for learning labs about Ansible.

Part V Review

Keep track of your part review progress with the checklist shown in Table P5-1. Details on each task follow the table.

Table P5-1 Part V Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		

Repeat All DIKTA Questions

For this task, use the PTP software to answer the "Do I Know This Already?" questions again for the chapters in this part of the book.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

This page intentionally left blank

Part VI

Final Review

CHAPTER 20

Final Review

Congratulations! You made it through the book, and now it's time to finish getting ready for the exam. This chapter helps you get ready to take and pass the exam in two ways.

First, this chapter focuses on the exam event. Now you need to think about what happens during the exam and what you need to do in these last few weeks before taking the exam. At this point, everything you do should be focused on getting ready to pass so that you can finish up this hefty task.

The second section of this chapter focuses on final content review. You should not just complete the previous chapter, which is the 48th technology chapter in the combined *CCNA 200-301 Official Cert Guide, Volume 1 and 2* books. Instead, you need to review, refine, deepen, and assess your skills. This second section of this chapter gives advice and suggestions on how to approach your final weeks of study before you take the CCNA 200-301 exam.

Advice About the Exam Event

Now that you have finished the bulk of this book, you could just register for your Cisco CCNA exam, show up, and take the exam. However, if you spend a little time thinking about the exam event itself, learning more about the user interface of the real Cisco exams and the environment at the Pearson VUE testing centers, you will be better prepared, particularly if this is your first Cisco exam.

This first of two major sections in this chapter gives some advice about the Cisco exams and the exam event itself, specifically about

- Question types
- Your time budget
- A sample time-check method
- The final week
- The 24 hours before the exam
- The final 30 minutes before the exam
- The hour after the exam

Exam Event: Learn About Question Types

In the weeks leading up to your exam, you should think more about the different types of exam questions and have a plan for how to approach those questions. One of the best ways to learn about the exam questions is to use some videos from the former Cisco Certification Exam Tutorial.

As for the backstory, Cisco formerly published a tool (the Cisco Certification Exam Tutorial) that gave anyone the ability to experience the Cisco exam user interface via an

interactive flash application. Cisco has updated the real exam interface; plus, Cisco removed the exam tutorial web pages with no equivalent replacement.

However, Cisco did make videos of the exam tutorial, with someone talking through the various question types. Cisco lists the videos in a post at the Cisco Learning Network (https://learningnetwork.cisco.com), so you can start by looking for those videos as follows:

- Go to the CLN (https://learningnetwork.cisco.com) and search for the post 34312.
- Use this direct link to the same page: https://learningnetwork.cisco.com/docs/DOC-34312?dtid=osscdc000283.
- Use https://blog.certskills.com/final-review, which links to a blog post of mine that lists the above link (as well as other links useful for final review).

While watching any of the videos about the exam tutorial, pay close attention to some important behaviors. For instance, for multichoice questions, the user interface

- Identifies single-answer questions with circles beside the answers versus multiple-answer questions showing squares before the answers.
- Prevents you from choosing too many answers.
- Supplies a popup window to tell you if you have selected too few answers if you try to move to the next question, so you can stop and go back and answer with the correct number of answers.
- Does not penalize you for guessing. You should always supply the number of answers that the question asks for. There is no penalty for guessing.

Note that because there is no penalty for guessing, you should always answer every question and answer with the exact number of correct answers.

For drag-and-drop questions, the user interface lets you change your mind while you are still working on the question. The draggable items begin in one location, and you drag and drop them to answer. You can just drag them back to where they were to begin the question.

For simulation questions:

- Pay close attention to the navigation to get to the command-line interface (CLI) on one of the routers. To do so, you have to click the PC icon for a PC connected to the router console; the console cable appears as a dashed line, whereas network cables are solid lines. (You should definitely look for this interaction in the exam tutorial videos.)
- Make sure that you look at the scroll areas at the top, at the side, and in the terminal emulator window. These scrollbars let you view the entire question and scenario.
- Make sure that you can toggle between the topology window and the terminal emulator window by clicking **Show topology** and **Hide topology**. The question window can be pretty crowded for sim questions, so the user interface gives you the means to toggle between seeing different parts of the question.

Both simlet and testlet questions give you one scenario with a group of related multichoice questions. However, the behavior with this small group (usually three or four) of multiple-choice questions differs from the flow of the more common standalone multiple-choice questions. In particular:

- You can move between the multiple-choice questions in a single simlet or testlet. You can answer one multiple-choice question, move to the second and answer it, and then move back to the first question, confirming that inside a testlet you can move around between questions.
- You can make a big mistake by not answering all questions or by not supplying enough answers, and the user interface does not prevent you from making that mistake.

On that second point, consider this scenario with a simlet question. You see the simlet question, answer the first three multichoice questions, but forget to look at the fourth multichoice question. If you click **Next**, you will see a generic popup window that Cisco uses as a prompt to ask whether you want to move on. However, it does not tell you that you did not answer a question at all, and it does not tell you if you answered with too few answers on a multi-answer question. **So be very careful when clicking *Next* when answering simlet and testlet questions.**

Exam Event: Think About Your Time Budget

On exam day, you need to keep an eye on your speed. Going too slowly hurts you because you might not have time to answer all the questions. Going too fast can be hurtful if you are rushing because you are fearful about running out of time. So, you need to be able to somehow know whether you are moving quickly enough to answer all the questions, while not rushing.

The exam user interface shows some useful information, namely a countdown timer and a question counter. The question counter shows a question number for the question you are answering, and it shows the total number of questions on your exam.

Unfortunately, some questions require lots more time than others, and for this and other reasons, time estimating can be a challenge.

First, before you show up to take the exam, you know only a range of the number of questions for the exam; for example, the Cisco website might list the CCNA exam as having from 50 to 60 questions (the Cisco website did not list a number of questions at the time this chapter was published). You will not know how many questions are on your exam until the exam begins, when you go through the screens that lead up to the point where you click **Start Exam**, which starts your timed exam.

Next, some questions (call them *time burners)* clearly take a lot more time to answer:

Normal-time questions: Multiple-choice and drag-and-drop, approximately one minute each

Time burners: Sims, simlets, and testlets, approximately six to eight minutes each

Finally, even though testlet and simlet questions contain several multiple-choice questions, the exam software counts each testlet and simlet question as one question in the question counter. For example, if a testlet question has four embedded multiple-choice questions, in the exam software's question counter, that counts as one question. So when you start

the exam, you might see that you will have 50 questions, but you don't know how many of those are time burners.

NOTE Cisco does not tell us why one person taking the exam might get 50 questions while someone else taking the same exam might get 60 questions, but it seems reasonable to think that the person with 50 questions might have a few more of the time burners, making the two exams equivalent.

You need a plan for how you will check your time, a plan that does not distract you from the exam. You can ponder the facts listed here and come up with your own plan. If you want a little more guidance, the next topic shows one way to check your time that uses some simple math so that it does not take much time away from the test.

Exam Event: A Sample Time-Check Method

As a suggestion, you can use the following math to do your time-check in a way that weights the time based on those time-burner questions. You do not have to use this method. But this math uses only addition of whole numbers, to keep it simple. It gives you a pretty close time estimate, in my opinion.

The concept is simple. Just do a simple calculation that estimates the time you should have used so far. Here's the math:

Number of questions answered so far + 7 per time burner answered so far

Then you check the timer to figure out how much time you have spent:

- **You have used exactly that much time or a little more time:** Your timing is perfect.
- **You have used less time:** You are ahead of schedule.
- **You have used noticeably more time:** You are behind schedule.

For example, if you have already finished 17 questions, two of which were time burners, your time estimate is l7 + 7 + 7 = 31 minutes. If your actual time is also 31 minutes, or maybe 32 or 33 minutes, you are right on schedule. If you have spent less than 31 minutes, you are ahead of schedule.

So, the math is pretty easy: questions answered, plus 7 per time burner, is the guesstimate of how long you should have taken so far if you are right on time.

NOTE This math is an estimate; I make no guarantees that the math will be an accurate predictor on every exam.

Exam Event: One Week Away

I have listed a variety of tips in the next few pages, broken down by timing versus the big exam event. First, this section discusses some items to consider when your exam is about a week away:

- **Get some earplugs:** Testing centers often have some, but if you do not want to chance it, come prepared with your own. (They will not let you bring your own noise-canceling headphones into the room if they follow the rules disallowing any user electronic

devices in the room, so think low-tech disposable earplugs, or even bring a cotton ball.) The testing center is typically one room within a building of a company that does something else as well, often a training center, and almost certainly you will share the room with other test takers coming and going. So, there are people talking in nearby rooms and other office noises. Earplugs can help.

- **Create an exam-event note-taking plan:** Some people like to spend the first minute of the exam writing down some notes for reference, before actually starting the exam. For example, maybe you want to write down the table of magic numbers for finding IPv4 subnet IDs. If you plan to do that, practice making those notes between now and exam day. Before each practice exam, transcribe those lists, just like you expect to do at the real exam.
- **Plan your travel to the testing center:** Leave enough time in your schedule so that you will not be rushing to make it just in time.
- **Practice your favorite relaxation techniques for a few minutes before each practice exam:** That way you can enter the exam event and be more relaxed and have more success.

Exam Event: 24 Hours Before the Exam

After you wake up on the big day, what should you be doing and thinking? Certainly, the better prepared you are, the better chances you have on the exam. But these small tips can help you do your best on exam day:

- Rest the night before the exam rather than staying up late to study. Clarity of thought is more important than one extra fact, especially because the exam requires so much analyzing and thinking rather than just remembering facts.
- Bring as few extra items with you as possible when leaving for the exam center. You may bring personal effects into the building and testing company's space, but not into the actual room in which you take the exam. So, save a little stress and bring as little extra stuff with you as possible. If you have a safe place to leave briefcases, purses, electronics, and so on, leave them there. However, the testing center should have a place to store your things as well. Simply put, the less you bring, the less you have to worry about storing. (For example, I have been asked to remove even my analog wristwatch on more than one occasion.)
- Plan time in your schedule for the day to not rush to get there and not rush when leaving either.
- Do not drink a 64-ounce caffeinated drink on the trip to the testing center. After the exam starts, the exam timer will not stop while you go to the restroom.
- Use any relaxation techniques that you have practiced to help get your mind focused while you wait for the exam.

Exam Event: The Last 30 Minutes

It's almost time! Here are a few tips for those last moments.

- Ask the testing center personnel for earplugs if you did not bring any—even if you cannot imagine using them. You never know whether using them might help.
- Ask for extra pens and laminated note sheets. The exam center will give you a laminated sheet and dry erase pen to take notes. (Test center personnel typically do not let you

bring paper and ink pen into the room, even if supplied by the testing center.) I always ask for a second pen as well.

- Test your pens and sheets before going into the room to take the exam. Better to get a replacement pen before the clock starts.
- Grab a few tissues from the box in the room, for two reasons. One, to avoid having to get up in the middle of the exam if you need to sneeze. Two, if you need to erase your laminated sheet, doing that with a tissue rather than your hand helps prevent the oil from your hand making the pen stop working well.
- Find a restroom to use before going into the testing center, or just ask where one is, to avoid needing to go during the approximately two-hour exam event. Note that the exam timer does not stop if you need to go to the restroom during the exam, and you first have to find the exam center contact before just heading to the restroom, so it can cost you a few minutes.

Exam Event: Reserve the Hour After the Exam

Some people pass these exams on the first attempt, and some do not. The exams are not easy. If you fail to pass the exam that day, you will likely be disappointed. And that is understandable. But it is not a reason to give up. In fact, I added this short topic to give you a big advantage in case you do fail.

The most important study hour for your next exam attempt is the hour just after your failed attempt.

Before you take the exam, prepare for how you will react if you do not pass. That is, prepare your schedule to give yourself an hour, or at least a half an hour, immediately after the exam attempt, in case you fail. Follow these suggestions to be ready for taking notes:

- Bring pen and paper, preferably a notebook you can write in if you have to write standing up or sitting somewhere inconvenient.
- Make sure you know where pen and paper are so that you can take notes immediately after the exam. Keep these items in your backpack if using the train or bus, or on your car seat.
- Install an audio recording app on your phone, and be prepared to start talking into your app when you leave the testing center.
- Before the exam, scout the testing center, and plan the place where you will sit and take your notes, preferably somewhere quiet.

Then, once you complete the exam, if you do not pass on this attempt, use the following process when taking notes:

- Write down anything in particular that you can recall from any question.
- Write down details of questions you know you got right as well, because doing so may help trigger a memory of another question.
- Draw the figures that you can remember.
- Most importantly, write down any tidbit that might have confused you: terms, configuration commands, **show** commands, scenarios, topology drawings, anything.
- Take at least three passes at remembering. That is, you will hit a wall where you do not remember more. So, start on your way back to the next place, and then find a place to pause and take more notes. And do it again.

- When you have sucked your memory dry, take one more pass while thinking of the major topics in the book, to see if that triggers any other memory of a question.

Once you have collected your notes, *you cannot share the information with anyone* because doing so would break the Cisco nondisclosure agreement (NDA). Cisco considers cheating a serious offense and strongly forbids sharing this kind of information publicly. But you can use your information to study for your next attempt. Remember, anything you can do to determine what you do not know is valuable when studying for your next attempt. See the section "Exam Review: Study Suggestions for Your Second Attempt" in this chapter for the rest of the story.

Exam Review

At this point, you should have read the other chapters in both the *CCNA 200-301 Official Cert Guide, Volumes 1 and 2*, and completed the Chapter Review and Part Review tasks. Now you need to do the final study and review activities before taking the exam, as detailed in this section.

This section suggests some new activities and repeats some activities that have been previously mentioned. However, whether the activities are new or old to you, they all focus on filling in your knowledge gaps, finishing off your skills, and completing the study process. While repeating some tasks you did at Chapter Review and Part Review can help, you need to be ready to take an exam, so the Exam Review asks you to spend a lot of time answering exam questions.

The Exam Review walks you through suggestions for several types of tasks and gives you some tracking tables for each activity. The main categories are

- Taking practice exams
- Finding what you do not know well yet (knowledge gaps)
- Configuring and verifying functions from the CLI
- Repeating the Chapter Review and Part Review tasks

Exam Review: Take Practice Exams

One day soon, you need to pass a real Cisco exam at a Pearson VUE testing center. So, it's time to practice the real event as much as possible.

A practice exam using the Pearson IT Certification Practice Test (PTP) exam software lets you experience many of the same issues as when taking a real Cisco exam. When you select *practice exam* mode, the PTP software (both desktop and web) gives you a number of questions, with a countdown timer shown in the window. When using this PTP mode, after you answer a question, you cannot go back to it (yes, that's true on Cisco exams). If you run out of time, the questions you did not answer count as incorrect.

The process of taking the timed practice exams helps you prepare in three key ways:

- To practice the exam event itself, including time pressure, the need to read carefully, and the need to concentrate for long periods
- To build your analysis and critical thinking skills when examining the network scenario built in to many questions
- To discover the gaps in your networking knowledge so that you can study those topics before the real exam

As much as possible, treat the practice exam events as if you were taking the real Cisco exam at a VUE testing center. The following list gives some advice on how to make your practice exam more meaningful, rather than just one more thing to do before exam day rolls around:

- Set aside two hours for taking a 90-minute timed practice exam.
- Make a list of what you expect to do for the 10 minutes before the real exam event. Then visualize yourself doing those things. Before taking each practice exam, practice those final 10 minutes before your exam timer starts. (The earlier section "Exam Event: The Last 30 Minutes" lists some suggestions about what to do in those last 10 minutes.)
- You cannot bring anything with you into the VUE exam room, so remove all notes and help materials from your work area before taking a practice exam. You can use blank paper, a pen, and your brain only. Do not use calculators, notes, web browsers, or any other app on your computer.
- Real life can get in the way, but if at all possible, ask anyone around you to leave you alone for the time you will practice. If you must do your practice exam in a distracting environment, wear headphones or earplugs to reduce distractions.
- Do not guess, hoping to improve your score. Answer only when you have confidence in the answer. Then, if you get the question wrong, you can go back and think more about the question in a later study session.

Using the Practice CCNA Exams

The PTP questions you can access as part of this book include exam banks labeled as follows:

- CCNA Volume 2 Exam 1
- CCNA Volume 2 Exam 2
- CCNA 200-301 Full Exam 1
- CCNA 200-301 Full Exam 2

The exams whose names begin "CCNA Volume 2" have questions from this Volume 2 book only, but no questions from Volume 1. The exams titled "CCNA 200-301" (without Volume 2 in the name) include questions from the entire breadth of CCNA topics, including topics covered in both the Volume 1 and Volume 2 books.

You should do your final review with the CCNA 200-301 exams. Just select those exams and deselect the others. Then you simply need to choose the **Practice Exam** option in the upper right and start the exam.

You should plan to take between one and three practice exams with the supplied CCNA exam databases. Even people who are already well prepared should do at least one practice exam, just to experience the time pressure and the need for prolonged concentration.

Table 20-1 gives you a checklist to record your different practice exam events. Note that recording both the date and the score is helpful for some other work you will do, so note both. Also, in the Time Notes section, if you finish on time, note how much extra time you had; if you run out of time, note how many questions you did not have time to answer.

Table 20-1 CCNA Practice Exam Checklist

Exam	Date	Score	Time Notes
CCNA			
CCNA			
CCNA			

Exam Review: Advice on How to Answer Exam Questions

Our everyday habits have changed how we all read and think in front of a screen. Unfortunately, those same habits often hurt our scores when taking computer-based exams.

For example, open a web browser. Yes, take a break and open a web browser on any device. Do a quick search on a fun topic. Then, before you click a link, get ready to think about what you just did. Where did your eyes go for the first 5 to 10 seconds after you opened that web page. Now, click a link and look at the page. Where did your eyes go?

Interestingly, web browsers and the content in web pages have trained us all to scan. Web page designers actually design content expecting certain scan patterns from viewers. Regardless of the pattern, when reading a web page, almost no one reads sequentially, and no one reads entire sentences. People scan for the interesting graphics and the big words, and then scan the space around those noticeable items.

Other parts of our electronic culture have also changed how the average person reads. For example, many of you grew up using texting and social media, sifting through hundreds or thousands of messages—but each message barely fills an entire sentence. Also, we find ourselves responding to texts, tweets, and emails and later realizing we did not really understand what the other person meant.

If you use those same habits when taking the exam, you will probably make some mistakes because you missed a key fact in the question, answer, or exhibits. It helps to start at the beginning and read all the words—a process that is amazingly unnatural for many people today.

NOTE I have talked to many college professors, in multiple disciplines, and Cisco Networking Academy instructors, and they consistently tell me that the number-one test-taking issue today is that people do not read the questions well enough to understand the details.

When you are taking the practice exams and answering individual questions, consider these two strategies. First, before the practice exam, think about your own personal strategy for how you will read a question. Make your approach to multiple-choice questions in particular be a conscious decision on your part. Second, if you want some suggestions on how to read an exam question, use the following strategy:

Step 1. Read the question itself, thoroughly, from start to finish.

Step 2. Scan any exhibit or figure.

Step 3. Scan the answers to look for the types of information. (Numeric? Terms? Single words? Phrases?)

Step 4. Reread the question thoroughly, from start to finish, to make sure that you understand it.

Step 5. Read each answer thoroughly, while referring to the figure/exhibit as needed. After reading each answer, before reading the next answer:

A. If correct, select as correct.

B. If for sure incorrect, mentally rule it out.

C. If unsure, mentally note it as a possible correct answer.

NOTE Cisco exams will tell you the number of correct answers. The exam software also helps you finish the question with the right number of answers noted. For example, for standalone multichoice questions, the software prevents you from selecting too many or too few answers. And you should guess the answer when unsure on the actual exam; there is no penalty for guessing.

Use the practice exams as a place to practice your approach to reading. Every time you click to the next question, try to read the question following your approach. If you are feeling time pressure, that is the perfect time to keep practicing your approach, to reduce and eliminate questions you miss because of scanning the question instead of reading thoroughly.

Exam Review: Additional Exams with the Premium Edition

Many people add other practice exams and questions other than those that come with this book. Frankly, using other practice exams in addition to the questions that come with this book can be a good idea, for many reasons. The other exam questions can use different terms in different ways, emphasize different topics, and show different scenarios that make you rethink some topics.

Note that Cisco Press does sell products that include additional test questions. The *CCNA 200-301 Official Cert Guide, Volume 2, Premium Edition eBook and Practice Test* product is basically the publisher's eBook version of this book. It includes a soft copy of the book in formats you can read on your computer and on the most common book readers and tablets. The product includes all the electronic content you would normally get with the print book, including all the question databases mentioned in this chapter. Additionally, this product includes two more CCNA exam databases (plus two more CCNA Volume 2 exam databases as well).

NOTE In addition to providing the extra questions, the Premium Editions have links to every test question, including those in the print book, to the specific section of the book for further reference. This is a great learning tool if you need more detail than what you find in the question explanations. You can purchase the eBooks and additional practice exams at 70 percent off the list price using the coupon on the back of the activation code card in the cardboard sleeve, making the Premium Editions the best and most cost-efficient way to get more practice questions.

Exam Review: Find Knowledge Gaps

One of the hardest things when doing your final exam preparation is to discover gaps in your knowledge and skills. In other words, what topics and skills do you need to know that you do not know? Or what topics do you think you know, but you misunderstand about some important fact? Finding gaps in your knowledge at this late stage requires more than just your gut feeling about your strengths and weaknesses.

This next task uses a feature of PTP to help you find those gaps. The PTP software tracks each practice exam you take, remembering your answer for every question and whether you got it wrong. You can view the results and move back and forth between seeing the question and seeing the results page. To find gaps in your knowledge, follow these steps:

Step 1. Pick and review one of your practice exams.

Step 2. Review each incorrect question until you are satisfied that you understand the question.

Step 3. When finished with your review for a question, mark the question.

Step 4. Review all incorrect questions from your exam until all are marked.

Step 5. Move on to the next practice exam.

Figure 20-1 shows a sample Question Review page, in which all the questions were answered incorrectly. The results list a Correct column, with no check mark, meaning that the answer was incorrect.

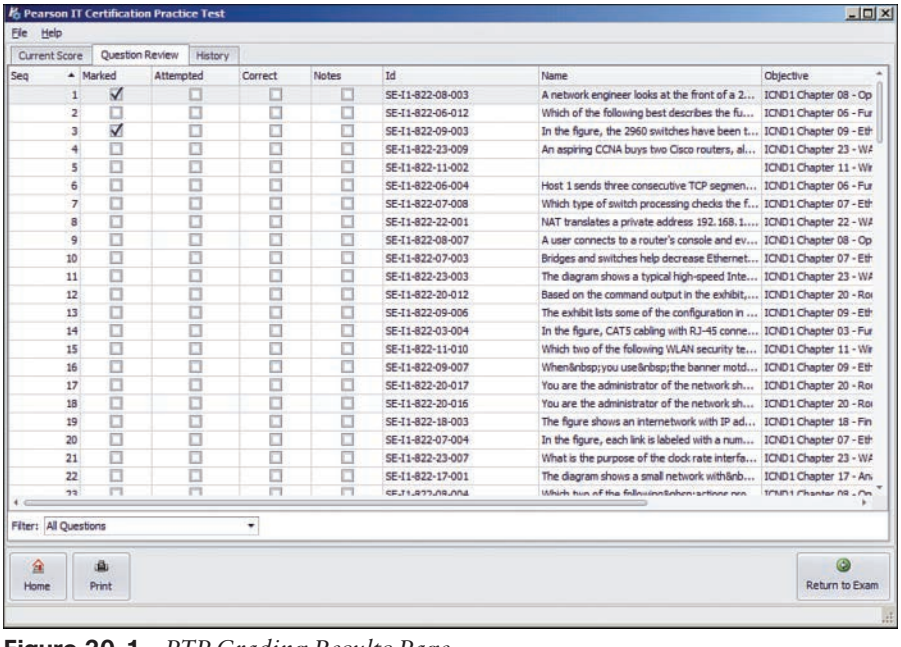


Figure 20-1 *PTP Grading Results Page*

To perform the process of reviewing questions and marking them as complete, you can move between this Question Review page and the individual questions. Just double-click a question to move back to that question. From the question, you can click **Grade Exam** to

move back to the grading results and to the Question Review page shown in Figure 20-1. The question window also shows the place to mark the question, in the upper left, as shown in Figure 20-2.

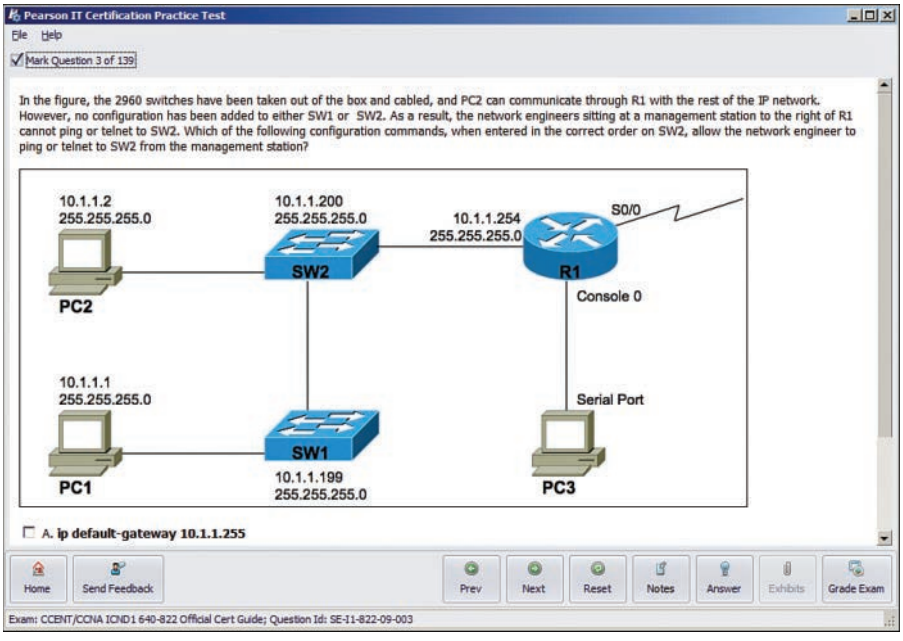


Figure 20-2 *Reviewing a Question, with the Mark Feature in the Upper Left*

If you want to come back later to look through the questions you missed from an earlier exam, start at the PTP home screen. From there, instead of clicking the Start button to start a new exam, click the **View Grade History** button to see your earlier exam attempts and work through any missed questions.

Track your progress through your gap review in Table 20-2. PTP lists your previous practice exams by date and score, so it helps to note those values in the table for comparison to the PTP menu.

Table 20-2 Tracking Checklist for Gap Review of Practice Exams

Original Practice Exam Date	Original Exam Score	Date Gap Review Was Completed

Exam Review: Practice Hands-On CLI Skills

To do well on sim and simlet questions, you need to be comfortable with many Cisco router and switch commands, and how to use them from a Cisco CLI. As described in the introduction to this book, sim questions require you to decide what configuration commands need to be configured to fix a problem or to complete a working configuration. Simlet questions require you to answer multiple-choice questions by first using the CLI to issue **show** commands to look at the status of routers and switches in a small network.

To be ready for the exam, you need to know the following kinds of information:

CLI navigation: Basic CLI mechanics of moving into and out of user, enable, and configuration modes

Individual configuration: The meaning of the parameters of each configuration command

Feature configuration: The set of configuration commands, both required and optional, for each feature

Verification of configuration: The **show** commands that directly identify the configuration settings

Verification of status: The **show** commands that list current status values and the ability to decide incorrect configuration or other problem causes of less-than-optimal status values

To help remember and review all this knowledge and skill, you can do the tasks listed in the next several pages.

CCNA Exam Topics with CLI Skill Requirements

Wondering about all the topics in CCNA 200-301 that specifically include configuration or verification skills? You can just scan the CCNA 200-301 exam topics. However, Table 20-3 and Table 20-4 summarize the topics for which you could consider practicing your CLI skills. The tables organize the topics into the same order used in the *CCNA 200-301 Official Cert Guides, Volume 1 and 2*, with chapter references.

Table 20-3 Topics with Configuration Skills in CCNA Volume 1

Topic	Volume 1 Chapter	Date You Finished Lab Review
Switch IPv4	6	
Verifying LAN switching	5	
Switch IPv4	6	
Switch passwords	6	
Switch interfaces	7	
VLANs	8	
VLAN trunking	8	
STP and RSTP	10	
Layer 2 EtherChannel	10	
Router interfaces	15	
Router IPv4 addresses and static routes	16	

Topic	Volume 1 Chapter	Date You Finished Lab Review
Router on a Stick	17	
Layer 3 switching with SVIs	17	
Layer 3 switching with routed interfaces and L3 EtherChannels	17	
OSPF fundamentals	20	
OSPF network types	21	
IPv6 addressing on routers	24	
IPv6 static routes	25	

Table 20-4 Topics with Configuration Skills in CCNA Volume 2

Topic	Volume 2 Chapter	Date You Finished Lab Review
Standard ACLs	2	
Extended ACLs	3	
Telnet and SSH Access ACLs	5	
Port Security	6	
DHCP client and DHCP relay	7	
DHCP snooping	8	
Dynamic ARP Inspection	8	
Syslog, NTP, CDP, and LLDP	9	
NAT, PAT	10	

You should research and choose your favorite methods and tools to get hands-on practice for CCNA. Those options include several that focus on giving you a specific activity to do. The options include the Pearson Network Simulator, Config Labs (on my blog), and Packet Tracer labs (on my blog).

First, one great way to practice is to use the Pearson Network Simulator (the sim) at www.pearsonitcertification.com/networksimulator. Pearson builds the sim to focus on lab exercises that help you learn and expand your skills with the topics in the CCNA exam. The sim also organizes the lab content so you can follow along with the books. You can get a sense for what the labs are like in the sim by going to the companion website for this book and downloading the Sim Lite, which uses the same core software but with a more limited number of labs compared to the full product.

Second, review the Config Checklist apps available from the book's companion website. For any configuration topics that require more than a few commands, the book collects the configuration commands into config checklists so that you can review and study in the days leading up to the exam. Take advantage of those checklists to review and remember all the required and optional configuration commands.

Finally, my blog site (https://blog.certskills.com) has informal lab exercises designed so that you can do the labs without any real gear or simulator. Config Labs list straightforward

configuration requirements. Your job: configure per the requirements, writing the configuration on paper or just typing into a text document. To learn more, go to

https://blog.certskills.com/config-labs

https://blog.certskills.com/packet-tracer-labs

Exam Review: Self-Assessment Pitfalls

When you take a practice exam with PTP, PTP gives you a score, on a scale from 300 to 1000. Why? Cisco gives a score of between 300 and 1000 as well. But the similarities end there.

With PTP, the score is a basic percentage but expressed as a number from 0 to 1000. For example, answer 80 percent correct, and the score is 800; get 90 percent correct, and the score is 900. If you start a practice exam and click through it without answering a single question, you get a 0.

However, Cisco does not score exams in the same way. The following is what we do know about Cisco exam scoring:

- Cisco uses a scoring scale from 300 to 1000.
- Cisco tells us that it gives partial credit but provides no further details.

So, what does an 800 or a 900 mean on the actual Cisco exams? Many people think those scores mean 80 percent or 90 percent, but we don't know. Cisco doesn't reveal the details of scoring to us. It doesn't reveal the details of partial credit. It seems reasonable to expect a sim question to be worth more points than a multiple-choice, single-answer question, but we do not know.

The reason I mention all these facts to you is this:

> Do not rely too much on your PTP practice exam scores to assess whether you are ready to pass. Those scores are a general indicator, in that if you make a 700 one time and a 900 a week later, you are probably now better prepared. But that 900 on your PTP practice exam does not mean you will likely make a 900 on the actual exam—because we do not know how Cisco scores the exam.

So, what can you use as a way to assess whether you are ready to pass? Unfortunately, the answer requires some extra effort, and the answer will not be some nice, convenient number that looks like an exam score. But you can self-assess your skills as follows:

1. When you do take an exam with PTP, you should understand the terms used in the questions and answers.
2. You should be able to look at the list of key topics from each chapter and explain a sentence or two about each topic to a friend.
3. You should be able to do subnetting math confidently with 100 percent accuracy at this point.
4. You should be able to do all the Config Labs, or labs of similar challenge level, and get them right consistently.
5. For chapters with **show** commands, you should understand the fields highlighted in gray in the examples spread throughout the book, and when looking at those

examples, you should know which values show configuration settings and which show status information.

6. For the key topics that list various troubleshooting root causes, when you review those lists, you should remember and understand the concept behind each item in the list without needing to look further at the chapter.

Exam Review: Adjustments for Your Second Attempt

None of us wants to take and fail any exam, but some of you will. And even if you pass the CCNA exam on your first try, if you keep going with Cisco certifications, you will probably fail some exams along the way. I mention failing an exam not to focus on the negative, but to help prepare you for how to pass the next attempt after failing an earlier attempt. This section collects some of the advice I have given to readers over the years who have contacted me after a failed attempt, asking for help about what to do next.

The single most important bit of advice is to change your mindset about Cisco exams. Cisco exams are not like high school or college exams where your failing grade matters. Instead, a Cisco exam is more like an event on the road to completing an impressive major accomplishment, one that most people have to try a few times to achieve.

For instance, achieving a Cisco certification is more like training to run a marathon in under four hours. The first time running a marathon, you may not even finish, or you may finish at 4:15 rather than under 4:00. But finishing a marathon in 4:15 means that you have prepared and are getting pretty close to your goal. Or maybe it is more like training to complete an obstacle course (for any *American Ninja Warrior* fans out there). Maybe you got past the first three obstacles today, but you couldn't climb over the 14-foot high warped wall. That just means you need to practice on that wall a little more.

So change your mindset. You're a marathon runner looking to improve your time or a Ninja Warrior looking to complete the obstacle course. And you are getting better skills every time you study, which helps you compete in the market.

With that attitude and analogy in mind, the rest of this section lists specific study steps that can help.

First, study the notes you took about your failed attempt. (See the earlier section "Exam Event: Reserve the Hour After the Exam.") Do not share that information with others, but use it to study. Before you take the exam again, you should be able to answer every actual exam question you can remember from the last attempt. Even if you never see the exact same question again, you will still get a good return for your effort.

Second, spend more time on activities that uncover your weaknesses. When doing that, you have to slow down and be more self-aware. For instance, answer practice questions in study mode, and *do not guess.* Do not click on to the next question, but pause and ask yourself if you are really sure about both the wrong and correct answers. If unsure, fantastic! You just discovered a topic for which to go back and dig in to learn it more deeply. Or when you do a lab, you may refer to your notes without thinking, so now think about it when you turn to your notes because that tells you where you are unsure. That might be a reminder that you have not mastered those commands yet.

Third, think about your time spent on the exam. Did you run out of time? Go too fast? Too slow? If too slow, were you slow on subnetting, or sims, or something else? Then make a

written plan as to how you will approach time on the next attempt and how you will track time use. And if you ran out of time, practice for the things that slowed you down.

Exam Review: Other Study Tasks

If you got to this point and still feel the need to prepare some more, this last topic gives you three suggestions.

First, the Chapter Review and Part Review sections give you some useful study tasks.

Second, use more exam questions from other sources. You can always get more questions in the Cisco Press Premium Edition eBook and Practice Test products, which include an eBook copy of this book plus additional questions in additional PTP exam banks. However, you can search the Internet for questions from many sources and review those questions as well.

NOTE Some vendors claim to sell practice exams that contain the literal exam questions from the official exam. These exams, called "brain dumps," are against the Cisco testing policies. Cisco strongly discourages using any such tools for study.

Finally, join in the discussions on the Cisco Learning Network. Try to answer questions asked by other learners; the process of answering makes you think much harder about the topic. When someone posts an answer with which you disagree, think about why and talk about it online. This is a great way to both learn more and build confidence.

Final Thoughts

You have studied quite a bit, worked hard, and sacrificed time and money to be ready for the exam. I hope your exam goes well, that you pass, and that you pass because you really know your stuff and will do well in your IT and networking career.

I encourage you to celebrate when you pass and ask advice when you do not. The Cisco Learning Network is a great place to make posts to celebrate and to ask advice for the next time around. I personally would love to hear about your progress through Twitter (@wendellodom) or my Facebook page (www.facebook.com/wendellodom). I wish you well, and congratulations for working through the entire book!

This page intentionally left blank

Part VII

Appendixes

Appendix A: Numeric Reference Tables

Appendix B: CCNA 200-301 Volume 2 Exam Updates

Appendix C: Answers to the “Do I Know This Already?” Quizzes

Glossary

This page intentionally left blank

APPENDIX A

Numeric Reference Tables

This appendix provides several useful reference tables that list numbers used throughout this book. Specifically:

Table A-1: A decimal-binary cross reference, useful when converting from decimal to binary and vice versa.

Table A-1 Decimal-Binary Cross Reference, Decimal Values 0–255

Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value
0	00000000	32	00100000	64	01000000	96	01100000
1	00000001	33	00100001	65	01000001	97	01100001
2	00000010	34	00100010	66	01000010	98	01100010
3	00000011	35	00100011	67	01000011	99	01100011
4	00000100	36	00100100	68	01000100	100	01100100
5	00000101	37	00100101	69	01000101	101	01100101
6	00000110	38	00100110	70	01000110	102	01100110
7	00000111	39	00100111	71	01000111	103	01100111
8	00001000	40	00101000	72	01001000	104	01101000
9	00001001	41	00101001	73	01001001	105	01101001
10	00001010	42	00101010	74	01001010	106	01101010
11	00001011	43	00101011	75	01001011	107	01101011
12	00001100	44	00101100	76	01001100	108	01101100
13	00001101	45	00101101	77	01001101	109	01101101
14	00001110	46	00101110	78	01001110	110	01101110
15	00001111	47	00101111	79	01001111	111	01101111
16	00010000	48	00110000	80	01010000	112	01110000
17	00010001	49	00110001	81	01010001	113	01110001
18	00010010	50	00110010	82	01010010	114	01110010
19	00010011	51	00110011	83	01010011	115	01110011
20	00010100	52	00110100	84	01010100	116	01110100
21	00010101	53	00110101	85	01010101	117	01110101
22	00010110	54	00110110	86	01010110	118	01110110
23	00010111	55	00110111	87	01010111	119	01110111
24	00011000	56	00111000	88	01011000	120	01111000
25	00011001	57	00111001	89	01011001	121	01111001
26	00011010	58	00111010	90	01011010	122	01111010
27	00011011	59	00111011	91	01011011	123	01111011
28	00011100	60	00111100	92	01011100	124	01111100
29	00011101	61	00111101	93	01011101	125	01111101
30	00011110	62	00111110	94	01011110	126	01111110
31	00011111	63	00111111	95	01011111	127	01111111

Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value	Decimal Value	Binary Value
128	10000000	160	10100000	192	11000000	224	11100000
129	10000001	161	10100001	193	11000001	225	11100001
130	10000010	162	10100010	194	11000010	226	11100010
131	10000011	163	10100011	195	11000011	227	11100011
132	10000100	164	10100100	196	11000100	228	11100100
133	10000101	165	10100101	197	11000101	229	11100101
134	10000110	166	10100110	198	11000110	230	11100110
135	10000111	167	10100111	199	11000111	231	11100111
136	10001000	168	10101000	200	11001000	232	11101000
137	10001001	169	10101001	201	11001001	233	11101001
138	10001010	170	10101010	202	11001010	234	11101010
139	10001011	171	10101011	203	11001011	235	11101011
140	10001100	172	10101100	204	11001100	236	11101100
141	10001101	173	10101101	205	11001101	237	11101101
142	10001110	174	10101110	206	11001110	238	11101110
143	10001111	175	10101111	207	11001111	239	11101111
144	10010000	176	10110000	208	11010000	240	11110000
145	10010001	177	10110001	209	11010001	241	11110001
146	10010010	178	10110010	210	11010010	242	11110010
147	10010011	179	10110011	211	11010011	243	11110011
148	10010100	180	10110100	212	11010100	244	11110100
149	10010101	181	10110101	213	11010101	245	11110101
150	10010110	182	10110110	214	11010110	246	11110110
151	10010111	183	10110111	215	11010111	247	11110111
152	10011000	184	10111000	216	11011000	248	11111000
153	10011001	185	10111001	217	11011001	249	11111001
154	10011010	186	10111010	218	11011010	250	11111010
155	10011011	187	10111011	219	11011011	251	11111011
156	10011100	188	10111100	220	11011100	252	11111100
157	10011101	189	10111101	221	11011101	253	11111101
158	10011110	190	10111110	222	11011110	254	11111110
159	10011111	191	10111111	223	11011111	255	11111111

Table A-2: A hexadecimal-binary cross reference, useful when converting from hex to binary and vice versa.

Table A-2 Hex-Binary Cross Reference

Hex	4-Bit Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Table A-3: Powers of 2, from 2^1 through 2^{32}.

Table A-3 Powers of 2

X	2^X	X	2^X
1	2	17	131,072
2	4	18	262,144
3	8	19	524,288
4	16	20	1,048,576
5	32	21	2,097,152
6	64	22	4,194,304
7	128	23	8,388,608
8	256	24	16,777,216
9	512	25	33,554,432
10	1024	26	67,108,864
11	2048	27	134,217,728
12	4096	28	268,435,456
13	8192	29	536,870,912
14	16,384	30	1,073,741,824
15	32,768	31	2,147,483,648
16	65,536	32	4,294,967,296

Table A-4: Table of all 33 possible subnet masks, in all three formats.

Table A-4 All Subnet Masks

Decimal	Prefix	Binary
0.0.0.0	/0	00000000 00000000 00000000 00000000
128.0.0.0	/1	10000000 00000000 00000000 00000000
192.0.0.0	/2	11000000 00000000 00000000 00000000
224.0.0.0	/3	11100000 00000000 00000000 00000000
240.0.0.0	/4	11110000 00000000 00000000 00000000
248.0.0.0	/5	11111000 00000000 00000000 00000000
252.0.0.0	/6	11111100 00000000 00000000 00000000
254.0.0.0	/7	11111110 00000000 00000000 00000000
255.0.0.0	/8	11111111 00000000 00000000 00000000
255.128.0.0	/9	11111111 10000000 00000000 00000000
255.192.0.0	/10	11111111 11000000 00000000 00000000
255.224.0.0	/11	11111111 11100000 00000000 00000000
255.240.0.0	/12	11111111 11110000 00000000 00000000
255.248.0.0	/13	11111111 11111000 00000000 00000000
255.252.0.0	/14	11111111 11111100 00000000 00000000
255.254.0.0	/15	11111111 11111110 00000000 00000000
255.255.0.0	/16	11111111 11111111 00000000 00000000
255.255.128.0	/17	11111111 11111111 10000000 00000000
255.255.192.0	/18	11111111 11111111 11000000 00000000
255.255.224.0	/19	11111111 11111111 11100000 00000000
255.255.240.0	/20	11111111 11111111 11110000 00000000
255.255.248.0	/21	11111111 11111111 11111000 00000000
255.255.252.0	/22	11111111 11111111 11111100 00000000
255.255.254.0	/23	11111111 11111111 11111110 00000000
255.255.255.0	/24	11111111 11111111 11111111 00000000
255.255.255.128	/25	11111111 11111111 11111111 10000000
255.255.255.192	/26	11111111 11111111 11111111 11000000
255.255.255.224	/27	11111111 11111111 11111111 11100000
255.255.255.240	/28	11111111 11111111 11111111 11110000
255.255.255.248	/29	11111111 11111111 11111111 11111000
255.255.255.252	/30	11111111 11111111 11111111 11111100
255.255.255.254	/31	11111111 11111111 11111111 11111110
255.255.255.255	/32	11111111 11111111 11111111 11111111

This page intentionally left blank

APPENDIX B

CCNA 200-301, Volume 2 Exam Updates

Over time, reader feedback allows Pearson to gauge which topics give our readers the most problems when taking the exams. To assist readers with those topics, the authors create new materials clarifying and expanding on those troublesome exam topics. As mentioned in the Introduction, the additional content about the exam is contained in a PDF on this book's companion website, at www.ciscopress.com/title/9781587147135.

This appendix provides you with updated information if Cisco makes minor modifications to the exam topics during the life of the 200-301 exam. In particular, this appendix does the following:

- Mentions technical items that might not have been mentioned elsewhere in the book
- Covers new topics if Cisco adds new content to the exam over time
- Provides a way to get up-to-the-minute current information about content for the exam

Note that this appendix shows updated information related to the subset of CCNA 200-301 exam topics covered in this book. Refer also to the *CCNA 200-301 Official Cert Guide, Volume 1*, for more details about the rest of the exam topics and for an Appendix B similar to that of this book.

Always Get the Latest at the Book's Product Page

Many of you are reading the version of this appendix that was available when your book was printed or when you downloaded the e-book. However, given that the main purpose of this appendix is to be a living, changing document, it is important that you look for the latest version online at the book's companion website. To do so, follow these steps:

Step 1. Browse to www.ciscopress.com/title/9781587147135.

Step 2. Click the **Updates** tab.

Step 3. If there is a new Appendix B document on the page, download the latest Appendix B document.

NOTE The downloaded document has a version number. Comparing the version of the print Appendix B (**Version 1.0**) with the latest downloadable version of this appendix, you should do the following:

- **Same version:** Ignore the PDF that you downloaded from the companion website.
- **Website has a later version:** Ignore this Appendix B in your book and read only the latest version that you downloaded from the companion website.

Technical Content

The current **Version 1.0** of this appendix does not contain additional technical coverage.

This page intentionally left blank

APPENDIX C

Answers to the "Do I Know This Already?" Quizzes

Chapter 1

1. D and E. Many headers include a field that identifies the next header that follows inside a message. Ethernet uses the Ethernet Type field, and the IP header uses the Protocol field. The TCP and UDP headers identify the application that should receive the data that follows the TCP or UDP header by using the port number field in the TCP and UDP headers, respectively.
2. A, B, C, and F. IP, not TCP, defines routing. Many other protocols define encryption, but TCP does not. The correct answers simply list various TCP features.
3. C. TCP, not UDP, performs windowing, error recovery, and ordered data transfer. Neither performs routing or encryption.
4. C and F. The terms *packet* and *L3PDU* refer to the header plus data encapsulated by Layer 3. *Frame* and *L2PDU* refer to the header (and trailer), plus the data encapsulated by Layer 2. *Segment* and *L4PDU* refer to the header and data encapsulated by the transport layer protocol.
5. B. Note that the hostname is all the text between the // and the /. The text before the // identifies the application layer protocol, and the text after the / represents the name of the web page.
6. C and D. Web traffic uses TCP as the transport protocol, with HTTP as the application protocol. As a result, the web server typically uses well-known TCP port 80, which is the well-known port for HTTP traffic. Messages flowing to the web server would have a destination TCP port of 80, and messages flowing from the server would have a source TCP port of 80.

Chapter 2

1. A and C. Standard ACLs check the source IP address. The address range 10.1.1.1–10.1.1.4 can be matched by an ACL, but it requires multiple **access-list** commands. Matching all hosts in Barney's subnet can be accomplished with the **access-list 1 permit 10.1.1.0 0.0.0.255** command.
2. A and D. The range of valid ACL numbers for standard numbered IP ACLs is 1–99 and 1300–1999, inclusive.
3. D. 0.0.0.255 matches all packets that have the same first three octets. This is useful when you want to match a subnet in which the subnet part comprises the first three octets, as in this case.
4. E. 0.0.15.255 matches all packets with the same first 20 bits. This is useful when you want to match a subnet in which the subnet part comprises the first 20 bits, as in this case.

5. A. The router always searches the ACL statements in order, and stops trying to match ACL statements after a statement is matched. In other words, it uses first-match logic. A packet with source IP address 1.1.1.1 would match any of the three explicitly configured commands described in the question. As a result, the first statement will be used.
6. B. One wrong answer, with wildcard mask 0.0.255.0, matches all packets that begin with 172.16, with a 5 in the last octet. One wrong answer matches only specific IP address 172.16.5.0. One wrong answer uses a wildcard mask of 0.0.0.128, which has only one wildcard bit (in binary), and happens to only match addresses 172.16.5.0 and 172.16.5.128. The correct answer matches the range of addresses 172.16.4.0–172.16.5.255.

Chapter 3

1. E and F. Extended ACLs can look at the Layer 3 (IP) and Layer 4 (TCP, UDP) headers and a few others, but not any application layer information. Named extended ACLs can look for the same fields as numbered extended ACLs.
2. A and E. The correct range of ACL numbers for extended IP access lists is 100 to 199 and 2000 to 2699. The answers that list the **eq www** parameter after 10.1.1.1 match the source port number, and the packets are going toward the web server, not away from it.
3. E. Because the packet is going toward any web client, you need to check for the web server's port number as a source port. The client IP address range is not specified in the question, but the servers are, so the source address beginning with 172.16.5 is the correct answer.
4. A and C. Before IOS 12.3, numbered ACLs must be removed and then reconfigured to remove a line from the ACL. As of IOS 12.3, you can also use ACL configuration mode and sequence numbers to delete one ACL line at a time.
5. C and D. In the command output, line number 10 references a permit command that matches addresses in subnet 172.16.1.0/24. The question stem identifies the subnet, so it indirectly asks about line 10 of the ACL. Any specific Access Control Entry (ACE) in ACL can be deleted in ACL config mode. Two methods can be used: the short **no** *line-number*, where line-number is the ACE's line number, or by issuing a **no** version of the **permit** or **deny** command, as shown in one of the correct answers. The three incorrect answers show correct commands but incorrect modes in which to use the commands.
6. C and D. The **show ip access-lists** and **show access-lists** commands both display the configuration of IPv4 access lists, including ACL line numbers. Neither the **show running-config** nor **show startup-config** commands list the ACL line numbers; in this case, the startup-config file would not contain the ACL configuration at all.

Chapter 4

1. B. A vulnerability is a weakness that can be exploited. Attack is not correct because it is a threat that is taking place.
2. D. When a vulnerability can be exploited, a threat is possible.
3. A and B. Attackers usually spoof the source IP address in packets they send in order to disguise themselves and make the actual IP address owner into a victim of the attack. MAC addresses can also be spoofed in ARP replies to confuse other hosts and routers on the local network. Destination IP addresses are not normally spoofed because packets used in the attack would go to unknown or nonexistent hosts. Finally, ARP address is not correct because it is not a legitimate term.
4. D. A denial-of-service attack is likely occurring because the attacker is trying to exhaust the target's TCP connection table with embryonic or incomplete TCP connections.
5. C. In a reflection attack, the goal is to force one host (the reflector) to reflect the packets toward a victim. Therefore, the spoofed source address contains the address of the victim and not the reflector.
6. A and C. Once an attacker is in position in a man-in-the-middle attack, traffic between hosts can be passively inspected and actively modified. This type of attack does not lend itself to inducing buffer overflows or using sweeps and scans.
7. B. In a brute-force attack, an attacker's software tries every combination of letters, numbers, and special characters to eventually find a string that matches a user's password.
8. D. The Cisco ISE platform provides the AAA services needed for authentication, authorization, and accounting. DHCP does not perform AAA but leases IP addresses to hosts instead. DNS resolves hostnames to IP addresses. SNMP is used for network management functions.
9. C. Physical access control is a necessary element of a security program that keeps sensitive locations like data centers and network closets locked and inaccessible, except to authorized personnel.

Chapter 5

1. B. If both commands are configured, IOS accepts only the password as configured in the **enable secret** command
2. A. The **service password-encryption** command encrypts passwords on a router or switch that would otherwise be shown in clear text. While a great idea in concept, the algorithm can be easily broken using websites found in the Internet. Cisco long ago provided replacements for commands that store passwords as clear text, instead using hashes—commands like **enable secret** and **username secret**. These commands are preferred in part because they avoid the issues of clear-text passwords and easily decrypted passwords.
3. B. The **enable secret** command stores an MD5 hash of the password. It is unaffected by the **service password-encryption** command. The router does not unhash the value back to the clear-text password. Instead, when the user types her clear-text password,

the router also hashes that password and compares that hashed value with the hashed value as listed in the configuration.

4. A. The **ip access-class 1 in** command enables ACL 1 for processing inbound Telnet and SSH connections into that router, based on the source IP address of those incoming packets. It has no impact on Telnet or SSH attempts from the router to some other host. It has no impact on a user later reaching enable mode. It also has nothing to do with filtering packets that would otherwise be routed through the router. Note that the ACL matches all packets whose source IP address is in subnet 172.16.4.0/23, which includes the range of numbers from 172.16.4.0 to 172.16.5.255.

5. B. Traditional and next-generation firewalls can check TCP and UDP port numbers, but next-generation firewalls are generally characterized as being able to also check application data beyond the Transport layer header. An NGFW would look into the application data, identifying messages that contain data structures used by Telnet, instead of matching with port numbers. This matching can catch attacks that seek to use port numbers that the firewall allows while using those ports to send data from applications that do not normally use those ports.

   For the other answers, a traditional firewall would likely match based on destination port 23, which is the well-known port for Telnet. IP protocol number has nothing to do with Telnet.

6. A and D. Both traditional and next-generation IPSs (NGIPSs) use a signature database, with each signature listing details of what fields would be in a series of messages to identify those messages as part of some exploit. They both also generate events for review by the security team.

   NGIPS devices add features that go beyond using a signature database, including gathering contextual information from hosts, like the OS used, currently running apps, open ports, and so on, so that the NGIPS does not have to log events if the hosts could not possibly be affected. Additionally, an NGIPS can use a list of reputation scores about IP addresses, domain names, and URIs of known bad actors, filtering traffic for sources that have a configured poor reputation level.

Chapter 6

1. B. The setting for the maximum number of MAC addresses has a default of 1, so the **switchport port-security maximum** command does not have to be configured. With sticky learning, you do not need to predefine the specific MAC addresses either. However, you must enable port security, which requires the **switchport port-security** interface subcommand.

2. B and D. First, about the sticky parameter…this command causes the switch to learn the source MAC and to add it to a **switchport port-security mac-address** *address* interface subcommand. However, port security adds that command to the running-config file; the network engineer must also issue a **copy running-config startup-config** EXEC command to save that configuration.

   About the other correct answer, users can connect a switch to the end of the cable, with multiple devices connected to that switch. That happens in real networks when users decide they need more ports at their desk. However, the default setting of

switchport port-security maximum 1 means that a frame from the second unique source MAC address would cause a violation, and with the default violation action, to err-disable the port.

For the other incorrect answer, the configuration does not prevent unknown MAC addresses from accessing the port because the configuration does not predefine any MAC address.

3. B and C. IOS adds MAC addresses configured by the port security feature as static MAC addresses, so they do not show up in the output of the **show mac address-table dynamic** command. **show mac address-table port-security** is not a valid command.

4. B. The question states that the port security status is secure-shutdown. This state is used only by the shutdown port security mode, and when used, it means that the interface has been placed into an err-disabled state. Those facts explain why the correct answer is correct and two of the incorrect answers are incorrect.

   The incorrect answer that mentions the violation counter is incorrect because in shutdown mode, the counter no longer increments once the interface is placed into secure-shutdown mode, and it resets to 0 once the interface is reset with the **shutdown** and then **no shutdown** commands.

5. B and C. First, about the two incorrect answers: In restrict mode, the arrival of a frame that violates the port security policy does not cause the switch to put the interface into err-disabled state. It does cause the switch to discard any frames that violate the policy, but it leaves the interface up and does not discard frames that do not violate the security policy, like the second frame that arrives.

   Regarding the two correct answers, a port in port security restrict does cause the switch to issue log messages for a violating frame, send SNMP traps about that same event (if SNMP is configured), and increment the counter of violating frames.

Chapter 7

1. B and D. The client sends a Discover message, with the server returning an Offer message. The client then sends a Request, with the server sending back the IP address in the Acknowledgment message.

2. A and B. The two correct answers list the two primary facts that impact which IP addresses the server will lease to clients. For the incorrect answer about DNS servers, the DHCP server does supply the IP address of the DNS servers, but not the hostnames of the DNS servers. Also, the DHCP server supplies the IP address (but not the MAC address) of the default gateway in each subnet.

3. A and C. A router needs to act as a DHCP relay agent if DHCP clients exist on the connected subnet and there is no DHCP server in that subnet. If a DHCP exists in the subnet, the router does not need to forward DHCP messages to a remote DHCP server (which is the function of a DHCP relay agent). The answer that mentions the **ip address dhcp** command makes the router interface act as a DHCP client and has nothing to do with DHCP relay agent.

4. D. The **ip address dhcp** command tells the router to obtain its address using DHCP. The router learns all the same information that a normal DHCP client would learn. The router uses the address listed as the default gateway to build a default route, using the

default gateway IP address as the next-hop address. The router continues to work like a router always does, forwarding packets based on its IP routing table.

5. B and C. The output shows the MAC address, IP address, subnet mask (in hex format), and the subnet broadcast address. Of those, the DHCP server supplies the information in the two correct answers. The two incorrect answers mention the MAC address (not supplied by DHCP, but known to the device's NIC) and the subnet broadcast address (calculated by the host).
6. D. Windows supports both **ipconfig** and **ipconfig /all** commands, but the **ipconfig** command does not mention the DNS servers. Note that the **ifconfig** command works on Linux and macOS, and the **ifconfig /all** command is an invalid command.

Chapter 8

1. A and C. DHCP Snooping must be implemented on a device that performs Layer 2 switching. The DHCP Snooping function needs to examine DHCP messages that flow between devices within the same broadcast domain (VLAN). Layer 2 switches, as well as multilayer switches, perform that function. Because a router performs only Layer 3 forwarding (that is, routing) and does not forward messages between devices in the same VLAN, a router does not provide a good platform to implement DHCP Snooping (and is not even a feature of Cisco IOS on routers). End-user devices would be a poor choice as a platform for DHCP Snooping because they would not receive all the DHCP messages, nor would they be able to prevent frames from flowing should an attack occur.
2. B and C. Switch ports connected to IT-controlled devices from which DHCP server messages may be received should be trusted by the DHCP Snooping function. Those devices include IT-controlled DHCP servers and IT-controlled routers and switches. All devices that are expected to be DHCP client devices (like PCs) are then treated as untrusted, because DHCP Snooping cannot know beforehand from which ports a DHCP-based attack will be launched. In this case, the ports connected to all three PCs will be treated as untrusted by DHCP Snooping.
3. C and D. Because of a default setting of untrusted, the switch does not need any configuration commands to cause a port to be untrusted. Of the two (incorrect) answers that related to the trust state, **no ip dhcp snooping trust**, in interface config mode, would revert from a trust configuration state to an untrusted state. The other answer, **ip dhcp snooping untrusted**, is not a valid command.

   The two correct answers list a pair of configuration commands that both must be included to enable DHCP Snooping (**ip dhcp snooping**) and to specify the VLAN list on which DHCP Snooping should operate (**ip dhcp snooping vlan 5**).
4. A. All the answers list commands with correct syntax that are useful for DHCP Snooping; however, the correct answer, **no ip dhcp snooping information**, disables DHCP Snooping's feature of adding DHCP Option 82 fields to DHCP messages. This setting is useful if the switch does not act as a DHCP relay agent. The opposite setting (without the **no** to begin the command) works when the multilayer switch acts as a DHCP relay agent.

5. B. DAI always uses a core function that examines incoming ARP messages, specifically the ARP message origin hardware and origin IP address fields, versus tables of data in the switch about correct pairs of MAC and IP addresses. DAI on a switch can use DHCP Snooping's binding table as the table of data with valid MAC/IP address pairs or use the logic in configured ARP ACLs. The question stem states that DAI uses DHCP Snooping, so the correct answer notes that the switch will compare the ARP message's origin hardware address to the switch's DHCP Snooping binding table.

   One incorrect answer mentions a comparison of the message's ARP origin MAC (hardware) address with the message's Ethernet source MAC address. DAI can perform that check, but that feature can be configured to be enabled or disabled, so DAI would not always perform this comparison. The other incorrect answers list logic never performed by DAI.

6. B and D. Because of a default setting of untrusted, the switch must be configured so DAI trusts that one port. To add that configuration, the switch needs the **ip arp inspection trust** command in interface config mode. The similar (incorrect) answer of **no ip arp inspection untrust** is not a valid command.

   To enable DAI for operation on a VLAN, the configuration needs one command: the **ip arp inspection vlan 6** command. This command both enables DAI and does so specifically for VLAN 6 alone. The answer **ip arp inspection** shows a command that would be rejected by the switch as needing more parameters.

7. C and D. With DAI, you can set a limit on the number of received ARP messages with a default burst interval of 1 second, or you can configure the burst interval. Once configured, DAI allows the configured number of ARP messages over the burst interval number of seconds. With the two correct answers, one shows 16 ARP messages, with a 4-second interval, for an average of 4 per second. The other correct answer shows a limit of 4, with the default burst interval of 1 second, for an average of 4. The two incorrect answers result in averages of 2 per second and 5 per second.

Chapter 9

1. D. By default, all message levels are logged to the console on a Cisco device. To do so, IOS uses logging level 7 (debugging), which causes IOS to send severity level 7, and levels below 7, to the console. All the incorrect answers list levels below level 7.

2. C. The **logging trap 4** command limits those messages sent to a syslog server (configured with the **logging host** *ip-address* command) to levels 4 and below, thus 0 through 4.

3. A. NTP uses protocol messages between clients and servers so that the clients can adjust their time-of-day clock to match the server. NTP is totally unrelated to serial line clocking. It also does not count CPU cycles, instead relying on messages from the NTP server. Also, the client defines the IP address of the server and does not have to be in the same subnet.

4. C. The **ntp server 10.1.1.1** command tells the router to be both an NTP server and client. However, the router first acts as an NTP client to synchronize its time with NTP server 10.1.1.1. Once synchronized, R1 knows the time to supply and can act as an NTP server.

5. E and F. CDP discovers information about neighbors. **show cdp** gives you several options that display more or less information, depending on the parameters used.
6. E and F. The **show lldp neighbors** command lists one line of output per neighbor. However, it does list the platform information of the neighbor, which typically includes the hardware model number. The **show lldp entry Hannah** command lists a group of messages about the neighboring router, including more detail about the hardware model and the IOS version.

Chapter 10

1. D. CIDR's original intent was to allow the summarization of multiple Class A, B, and C networks to reduce the size of Internet routing tables. Of the answers, only 200.1.0.0 255.255.0.0 summarizes multiple networks.
2. B and E. RFC 1918 identifies private network numbers. It includes Class A network 10.0.0.0, Class B networks 172.16.0.0 through 172.31.0.0, and Class C networks 192.168.0.0 through 192.168.255.0.
3. C. With static NAT, the entries are statically configured. Because the question mentions translation for inside addresses, the **inside** keyword is needed in the command.
4. A. With dynamic NAT, the entries are created as a result of the first packet flow from the inside network.
5. A. The **list 1** parameter references an IP ACL, which matches packets, identifying the inside local addresses.
6. A and C. The configuration is missing the **overload** keyword in the **ip nat inside source** command and in the **ip nat outside** interface subcommand on the serial interface.
7. B. The last line mentions that the pool has seven addresses, with all seven allocated, with the misses counter close to 1000—meaning that close to 1000 new flows were rejected because of insufficient space in the NAT pool

Chapter 11

1. A, B, and E. QoS tools manage bandwidth, delay, jitter, and loss.
2. B and C. The Class of Service (CoS) field exists in the 802.1Q header, so it would be used only on trunks, and it would be stripped of the incoming data-link header by any router in the path. The MPLS EXP bits exist as the packet crosses the MPLS network only. The other two fields, IP Precedence (IPP) and Differentiated Services Code Point (DSCP), exist in the IP header and would flow from source host to destination host.
3. A, B, and C. In general, matching a packet with DiffServ relies on a comparison to something inside the message itself. The 802.1p CoS field exists in the data-link header on VLAN trunks; the IP DSCP field exists in the IP header; and extended ACLs check fields in message headers. The SNMP Location variable does not flow inside individual packets but is a value that can be requested from a device.
4. B and C. Low Latency Queuing (LLQ) applies priority queue scheduling, always taking the next packet from the LLQ if a packet is in that queue. To prevent queue starvation of the other queues, IOS also applies policing to the LLQ. However, applying shaping

to an LLQ slows the traffic, which makes no sense with the presence of a policing function already.

5. A and D. Policers monitor the bit rate and take action if the bit rate exceeds the policing rate. However, the action can be to discard some packets, or to re-mark some packets, or even to do nothing to the packets, simply measuring the rate for later reporting. For shaping, when a shaper is enabled because the traffic has exceeded the shaping rate, the shaper always queues packets and slows the traffic. There is no option to re-mark the packets or to bypass the shaping function.
6. C and D. Drop management relies on the behavior of TCP, in that TCP connections slow down sending packets due to the TCP congestion window calculation. Voice traffic uses UDP, and the question states that queue 1 uses UDP. So, queues 2 and 3 are reasonable candidates for using a congestion management tool.

Chapter 12

1. D. With this design but no FHRP, host A can send packets off-subnet as long as connectivity exists from host A to R1. Similarly, host B can send packets off-subnet as long as host B has connectivity to router R2. Both routers can attach to the same LAN subnet and basically ignore each other in relation to their roles as default router because they do not use an FHRP option. When either router fails, the hosts using that router as default router have no means by which to fail over.
2. C. The use of an FHRP in this design purposefully allows either router to fail and still support off-subnet traffic from all hosts in the subnet. Both routers can attach to the same LAN subnet per IPv4 addressing rules.
3. C. HSRP uses a virtual IP address. The virtual IP address comes from the same subnet as the routers' LAN interfaces but is a different IP address than the router addresses configured with the **ip address** interface subcommand. As a result, the hosts will not point to 10.1.19.1 or 10.1.19.2 in this design. The other wrong answer lists an idea of using the Domain Name System (DNS) to direct hosts to the right default router; although this idea exists in some other forms of network load balancing, it is not a part of any of the three FHRP protocols.
4. B. SNMPv1 and SNMPv2c use community strings to authenticate Get and Set messages from an NMS. The agent defines a read-only community and can define a read-write community as well. Get requests, which read information, will be accepted if the NMS sends either the read-only or the read-write community with those requests.
5. A and C. SNMP agents reside on a device being managed. When an event happens about which the device wants to inform the SNMP manager, the agent sends either an SNMP Trap or SNMP Inform to the SNMP manager. The SNMP manager normally sends an SNMP Get Request message to an agent to retrieve MIB variables or an SNMP Set Request to change an MIB variable on the agent.
6. A. FTP uses both a control connection and a data connection. The FTP client initiates the control connection. However, in active mode, the FTP server initiates the data connection. Also, note that FTP does not use TLS, while FTP Secure (FTPS) does use TLS.

7. B and D. TFTP supports fewer functions than FTP as a protocol. For instance, the client cannot change the current directory on the server, add directories, remove directories, or list the files in the directory. Both TFTP and FTP support the ability to transfer files in either direction.

Chapter 13

1. B and D. The access layer switches play the role of connecting to the endpoint devices, whether they are end-user devices or servers. Then, from the access to the distribution layer, each access layer connects to two distribution switches typically, but with no direct connections between access layer switches, creating a mesh (but a partial mesh). A two-tier design, also called a collapsed core, does not use core switches at all.
2. A and C. The access layer switches, not the distribution layer switches, play the role of connecting to the endpoint devices, whether they are end-user devices or servers. Then, from the access to the distribution layer, each access layer connects to two distribution switches typically, but with no direct connections between access layer switches, creating a mesh (but a partial mesh). A three-tier design, also called a core design, does use core switches, with a partial mesh of links between the distribution and core switches. Basically, each distribution switch connects to multiple core switches but often does not connect directly to other distribution switches.
3. D. The access layer uses access switches, which connect to endpoint devices. A single access switch with its endpoint devices looks like a star topology. The distribution layer creates a partial mesh of links between the distribution switches and access switches, so it is neither a full mesh nor a hybrid.
4. A and C. With a SOHO LAN, one integrated device typically supplies all the necessary functions, including routing, switching, wireless access point (AP), and firewall. The AP uses standalone mode, without a wireless LAN controller (WLC), and without a need to encapsulate frames in CAPWAP.
5. A. First, the switch does not supply power based on a configured value to avoid the unfortunate case of supplying power over the cable to a device that does not support the circuitry to receive the power, because doing so will likely harm the electronics on the connected device.

   If configured to use PoE, the switch begins with IEEE autonegotiation messages while sensing the load on the circuit, which indicates whether the device desires to receive power, and indicates the power class desired (which dictates the amount of power to initially deliver). Note that once the attached device (called the powered device, or PD) boots, the PD can request additional power using CDP and/or LLDP.
6. B and D. Universal Power over Ethernet (UPoE) and the enhanced UPoE Plus (UPoE+) supply power over all four pairs of the cable. Note that 1000BASE-T and faster UTP-based Ethernet standards often require four pair, whereas earlier/slower standards did not, and UPoE/UPoE+ take advantage of the existence of four pairs to supply power over all four pairs. Power over Ethernet (PoE) and PoE+ use two pairs for power and therefore work with Ethernet standards like 10BASE-T and 100BASE-T that use two pairs only.

Chapter 14

1. B and C. A Metro Ethernet E-Tree service uses a rooted point-to-multipoint Ethernet Virtual Connection (EVC), which means that one site connected to the service (the root) can communicate directly with each of the remote (leaf) sites. However, the leaf sites cannot send frames directly to each other; they can only send frames to the root site. Topology design like this that allows some but not all pairs of devices in the group to communicate is called a partial mesh, or hub and spoke, or in some cases a multipoint or point-to-multipoint topology.

   Of the incorrect answers, the *full mesh* term refers to topology designs in which each pair in the group can send data directly to each other, which is typical of a MetroE E-LAN service. The term *point-to-point* refers to topologies with only two nodes in the design, and they can send directly to each other, typical of a MetroE E-Line service.

2. A. Metro Ethernet uses Ethernet access links of various types. Time-division multiplexing (TDM) links such as serial links, even higher-speed links like T3 and E3, do not use Ethernet protocols, and are less likely to be used. MPLS is a WAN technology that creates a Layer 3 service.

   Two answers refer to Ethernet standards usable as the physical access link for a Metro Ethernet service. However, 100BASE-T supports cable lengths of only 100 meters, so it is less likely to be used as a Metro Ethernet access link in comparison to 100BASE-LX10, which supports lengths of 10 km.

3. A and D. An E-LAN service is one in which the Metro Ethernet service acts as if the WAN were a single Ethernet switch so that each device can communicate directly to every other device. As a result, the routers sit in the same subnet. With one headquarters router and 10 remote sites, each router will have 10 OSPF neighbors.

4. B and C. A Layer 3 MPLS VPN creates an IP service with a different subnet on each access link. With one headquarters router and 10 remote sites, 11 access links exist, so 11 subnets are used.

   As for the OSPF neighbor relationships, each enterprise router has a neighbor relationship with the MPLS provider edge (PE) router, but not with any of the other enterprise (customer edge) routers. So each remote site router would have only one OSPF neighbor relationship.

5. D. Architecturally, MPLS allows for a wide variety of access technologies. Those include TDM (that is, serial links), Frame Relay, ATM, Metro Ethernet, and traditional Internet access technologies such as DSL and cable.

6. A. The PE-CE link is the link between the customer edge (CE) router and the MPLS provider's provider edge (PE) router. When using OSPF, that link will be configured to be in some area. OSPF design allows for that link to be in the backbone area, or not, through the use of the OSPF super backbone, which exists between all the PE routers.

7. A. The term *remote access VPN*, or *client VPN*, typically refers to a VPN for which one endpoint is a user device, such as a phone, tablet, or PC. In those cases, TLS is the more likely protocol to use. TLS is included in browsers, and is commonly used to connect securely to websites. GRE along with IPsec is more likely to be used to create a site-to-site VPN connection. FTPS refers to FTP Secure, which uses TLS to secure FTP sessions.

Chapter 15

1. A, B, and E. The hypervisor will virtualize RAM, CPU, NICs, and storage for each VM. The hypervisor itself is not virtualized, but rather does the work to virtualize other resources. Also, as virtual machines, the VMs do not use power, so the power is not virtualized.
2. D. Hypervisors create a virtual equivalent of Ethernet switching and cabling between the VMs and the physical NICs. The VMs use a virtual NIC (vNIC). The hypervisor uses a virtual switch (vswitch), which includes the concept of a link between a vswitch port and each VM's vNIC. The vswitch also connects to both physical NICs. The switch can then be configured to create VLANs and trunks as needed.
3. B. Platform as a Service (PaaS) supplies one or more virtual machines (VMs) that have a working operating system (OS) as well as a predefined set of software development tools.

   As for the wrong answers, Software as a Service (SaaS) supplies a predefined software application, but typically with no ability to then later install your own applications. Infrastructure as a Service (IaaS) supplies one or more working VMs, optionally with an OS installed, so it could be used for software development, but the developer would have to install a variety of development tools, making IaaS less useful than a PaaS service. Finally, Server Load Balancing as a Service (SLBaaS) can be offered as a cloud service, but it is not a general service in which customers get access to VMs on which they can then install their own applications.
4. A. Infrastructure as a Service (IaaS) supplies one or more working virtual machines (VMs), optionally with an OS installed, as a place where you can then customize the systems by installing your own applications.

   Software as a Service (SaaS) supplies a predefined software application, but typically with no ability to then later install your own applications. Platform as a Service (PaaS) could be used to install your own application, because PaaS does supply one or more VMs, but it is most likely used as a software development environment, a service designed specifically to be used for development, with VMs that include various tools that are useful for software development. Finally, Server Load Balancing as a Service (SLBaaS) can be offered as a cloud service, but it is not a general service in which customers get access to VMs on which they can then install their own applications.
5. A. Both options that use the Internet allow for easier migration because public cloud providers typically provide easy access over the Internet. An intercloud exchange is a purpose-built WAN service that connects to enterprises as well as most public cloud providers, with the advantage of making the cloud migration process easier. The one correct answer—the worst option in terms of being prepared for migrating to a new cloud provider—is to use a private WAN connection to one cloud provider. While useful in other ways, migrating when using this strategy would require installing a new private WAN connection to the new cloud provider.
6. A and C. Private WAN options use technologies like Ethernet WAN and MPLS, both of which keep data private by their nature and which include QoS services. An intercloud exchange is a purpose-built WAN service that connects to enterprises as well as

most public cloud providers, using the same kinds of private WAN technology with those same benefits.

For the two incorrect answers, both use the Internet, so both cannot provide QoS services. The Internet VPN option does encrypt the data to keep it private.

Chapter 16

1. A. The *data plane* includes all networking device actions related to the receipt, processing, and forwarding of each message, as in the case described in the question. The term *table plane* is not used in networking. The *management plane* and *control plane* are not concerned with the per-message forwarding actions.
2. C. The *control plane* includes all networking device actions that create the information used by the data plane when processing messages. The control plane includes functions like IP routing protocols and Spanning Tree Protocol (STP).

   The term *table plane* is not used in networking. The *management plane* and *data plane* are not concerned with collecting the information that the data plane then uses.
3. C. Although many variations of SDN architectures exist, they typically use a centralized controller. That controller may centralize some or even all control plane functions in the controller. However, the data plane function of receiving messages, matching them based on header fields, taking actions (like making a forwarding decision), and forwarding the message still happens on the network elements (switches) and not on the controller.

   For the incorrect answers, the control plane functions may all happen on the controller, or some may happen on the controller, and some on the switches. The northbound and southbound interfaces are API interfaces on the controller, not on the switches.
4. A. The OpenDaylight Controller uses an Open SDN model with an OpenFlow southbound interface as defined by the Open Networking Foundation (ONF). The ONF SDN model centralizes most control plane functions. The APIC model for data centers partially centralizes control plane functions. The APIC-EM controller (as of time of publication) makes no changes to the control plane of routers and switches, leaving those to run with a completely distributed control plane.
5. C and D. ACI uses a spine-leaf topology. With a single-site topology, leaf switches must connect to all spine switches, and leaf switches must not connect to other leaf switches. Additionally, a leaf switch connects to some endpoints, with the endpoints being spread across the ports on all the leaf switches. (In some designs, two or more leaf switches connect to the same endpoints for redundancy and more capacity.)
6. A and D. Controller-based networks use a controller that communicates with each network device using a southbound interface (an API and protocol). By gathering network information into one central device, the controller can then allow for different operational models. The models often let the operator think in terms of enabling features in the network, rather than thinking about the particulars of each device and command on each device. The controller then configures the specific commands, resulting in more consistent device configuration.

For the incorrect answers, both the old and new models use forwarding tables on each device. Also, controllers do not add to or remove from the programmatic interfaces on each device, some of which existed before controllers, but rather supply useful and powerful northbound APIs.

Chapter 17

1. C. The SDA underlay consists of the network devices and connections, along with configuration that allows IP connectivity between the SDA nodes, for the purpose of supporting overlay VXLAN tunnels. The fabric includes both the underlay and overlay, while VXLAN refers to the protocol used to create the tunnels used by the overlay.
2. B. The overlay includes the control plane and data plane features to locate the endpoints, decide to which fabric node a VXLAN tunnel should connect, direct the frames into the tunnel, and perform VXLAN tunnel encapsulation and de-encapsulation. The SDA underlay exists as network devices, links, and a separate IP network to provide connectivity between nodes to support the VXLAN tunnels.

   The fabric includes both the underlay and overlay, while VXLAN refers to the protocol used to create the tunnels used by the overlay.
3. D. The SDA overlay creates VXLAN tunnels between SDA edge nodes. Edge nodes then create a data plane by forwarding frames sent by endpoints over the VXLAN tunnels. LISP plays a role in the overlay as the control plane, which learns the identifiers of each endpoint, matching the endpoint to the fabric node that can teach the endpoint, so that the overlay knows where to create VXLAN tunnels.

   For the other incorrect answers, note that while GRE is a tunneling protocol, SDA uses VXLAN for tunneling, and not GRE. Finally, OSPF acts as a control plane routing protocol, rather than a data plane protocol for SDA.
4. A and D. As with any SDA feature, the configuration model is to configure the feature using DNA Center, with DNA Center using southbound APIs to communicate the intent to the devices. The methods to configure the feature using DNA Center include using the GUI or using the northbound REST-based API.

   Of the incorrect answers, you would not normally configure any of the SDA devices directly. Also, while DNA Center can use NETCONF as a southbound protocol to communicate with the SDA fabric nodes, it does not use NETCONF as a northbound API for configuration of features.
5. B, C, and D. Cisco DNA Center manages traditional network devices with traditional protocols like Telnet, SSH, and SNMP. DNA Center can also use NETCONF and RESTCONF if supported by the device. Note that while useful tools, Ansible and Puppet are not used by DNA Center.
6. A and D. Traditional network management platforms can do a large number of functions related to managing traditional networks and network devices, including the items listed in the two correct answers. However, when using Cisco's Prime Infrastructure as a traditional network management platform for comparison, it does not support SDA configuration, nor does it find the end-to-end path between two endpoints and analyze the ACLs in the path. Note that the two incorrect answers reference features available in DNA Center.

Chapter 18

1. B and D. The six primary required features of REST-based APIs include three features mentioned in the answers: a client/server architecture, stateless operation, notation of whether each object is cacheable. Two items from these three REST attributes are the correct answers. Of the incorrect answers, classful operation is the opposite of the REST-based API feature of classless operation. For the other incorrect answer, although many REST-based APIs happen to use HTTP, REST APIs do not have to use HTTP.

2. B and D. In the CRUD software development acronym, the matching terms (create, read, update, delete) match one or more HTTP verbs. While the HTTP verbs can sometimes be used for multiple CRUD actions, the following are the general rules: create performed by HTTP POST; read by HTTP GET; update by HTTP PATCH, PUT (and sometimes POST); delete by HTTP DELETE.

3. C. The URI for a REST API call uses a format of protocol://hostname/resource?parameters. The API documentation details the resource part of the URI, as well as any optional parameters. For instance, in this case, the resource section is /dna/intent/api/v1/network-device. Additionally, the API documentation for this resource details optional parameters in the query field as listed after the ? in the URI.

4. A and D. Of the four answers, two happen to be most commonly used to format and serialize data returned from a REST API: JSON and XML. For the incorrect answers, JavaScript is a programming language that first defined JSON as a data serialization language. YAML is a data serialization/modeling language and can be found most often in configuration management tools like Ansible.

5. A and D. JSON defines variables as key:value pairs, with the key on the left of the colon (:) and always enclosed in double quotation marks, with the value on the right. The value can be a simple value or an object or array with additional complexity. The number of objects is defined by the number of matched curly brackets ({ and }), so this example shows a single JSON object.

   The one JSON object shown here includes one key and one :, so it has a single key:value pair (making one answer correct). The value in that key:value pair itself is a JSON array (a list in Python) that lists numbers 1, 2, and 3. The fact that the list is enclosed in square brackets defines it as a JSON array.

6. C and D. To interpret this JSON data, first look for the innermost pairing of either curly brackets { }, which denote one object, or square brackets [ ], which denote one array. In this case, the gray highlighted area is one JSON object, enclosed with { } and no other brackets of either type inside. That makes the gray area one object, which itself holds key:value pairs.

   Inside that one object, four key:value pairs exist, with the key before each colon and the value after each colon. That means "type" is a key, and "ACCESS" is one of the values.

   If you look at the other pair of curly brackets that begin and end the JSON data, that pair defines an object. That object has a key of "response" (making one answer incorrect). The "response" key then has a value equal to the entire inner object (the gray highlighted part), confirming one of the correct answers.

Chapter 19

1. C. Devices with the same role in an enterprise should have a very similar configuration. When engineers make unique changes on individual devices—different changes from those made in the majority of devices with that same role—those devices' configurations become different than the intended ideal configuration for every device with that role. This effect is known as configuration drift. Configuration management tools can monitor a device's configuration versus a file that shows the intended ideal configuration for devices in that role, noting when the device configuration drifts away from that ideal configuration.

2. A and B. The version control system, applied to the centralized text files that contain the device configurations, automatically tracks changes. That means the system can see which user edited the file, when, and exactly what change was made, with the ability to make comparisons between different versions of the files.

   The two incorrect answers list very useful features of a configuration management tool, but those answers list features typically found in the configuration management tool itself rather than in the version control tool.

3. D. Configuration monitoring (a generic description) refers to a process of checking the device's actual configuration versus the configuration management system's intended configuration for the device. If the actual configuration has moved away from the intended configuration—that is, if configuration drift has occurred—configuration monitoring can either reconfigure the device or notify the engineering staff.

   For the other answers, two refer to features of the associated version control software typically used along with the configuration management tool. Version control software will track the identity of each user who changes files and track the differences in files over time. The other incorrect answer is a useful feature of many configuration management tools, in which the tool verifies that the configuration will be accepted when attempted (or not). However, that useful feature is not part of what is called configuration monitoring.

4. A. Ansible uses a push model, in which the Ansible control node decides when to configure a device based on the instructions in a playbook. Puppet and Chef use pull models, in which an agent asks for information from a server, with the agent then making the decision of whether it needs to pull configuration data to itself and reconfigure itself.

5. B and C. Of the terms *manifest* and *recipe*, both refer to files that define the actions to take and/or the end state desired when taking action in one of the configuration management tools. These files go by the names Ansible playbook, Puppet manifest, and Chef recipe.

GLOSSARY

NUMERICS

3G/4G Internet An Internet access technology that uses wireless radio signals to communicate through mobile phone towers, most often used by mobile phones, tablets, and some other mobile devices.

802.1 Q The IEEE standardized protocol for VLAN trunking.

A

AAA Authentication, authorization, and accounting. Authentication confirms the identity of the user or device. Authorization determines what the user or device is allowed to do. Accounting records information about access attempts, including inappropriate requests.

AAA server *See* authentication, authorization, and accounting (AAA) server.

Access Control Entry (ACE) One line in an access control list (ACL).

access interface A LAN network design term that refers to a switch interface connected to end-user devices, configured so that it does not use VLAN trunking.

access layer In a campus LAN design, the switches that connect directly to endpoint devices (servers, user devices), and also connect into the distribution layer switches.

access link In Frame Relay, the physical serial link that connects a Frame Relay DTE device, usually a router, to a Frame Relay switch. The access link uses the same physical layer standards as do point-to-point leased lines.

access link (WAN) A physical link between a service provider and its customer that provides access to the SP's network from that customer site.

access rate The speed at which bits are sent over an access link.

accounting In security, the recording of access attempts. *See also* AAA.

ACI *See* Application Centric Infrastructure (ACI).

ACL Access control list. A list configured on a router to control packet flow through the router, such as to prevent packets with a certain IP address from leaving a particular interface on the router.

Active Directory A popular set of identity and directory services from Microsoft, used in part to authenticate users.

administrative distance In Cisco routers, a means for one router to choose between multiple routes to reach the same subnet when those routes are learned by different routing protocols. The lower the administrative distance, the more preferred the source of the routing information.

agent Generally, an additional software process or component running in a computing device for some specific purpose; a small and specific software service.

agent-based architecture With configuration management tools, an architecture that uses a software agent inside the device being managed as part of the functions to manage the configuration.

agentless architecture With configuration management tools, an architecture that does not need a software agent inside the device being managed as part of the functions to manage the configuration, instead using other mainstream methods like SSH and NETCONF.

amplification attack A reflection attack that leverages a service on the reflector to generate and reflect huge volumes of reply traffic to the victim.

analog modem *See* modem.

Ansible A popular configuration management application, which can be used with or without a server, using a push model to move configurations into devices, with strong capabilities to manage network device configurations.

Ansible inventory Device host names along with information about each device, like device roles, so Ansible can perform functions for subsets of the inventory.

Ansible playbook Files with actions and logic about what Ansible should do.

Ansible template A text file, written in Jinja2 language, that lists configuration but with variable names substituted for values, so that Ansible can create standard configurations for multiple devices from the same template.

anti-replay Preventing a man in the middle from copying and later replaying the packets sent by a legitimate user, for the purpose of appearing to be a legitimate user.

antivirus Software that monitors files transferred by any means, for example, web or email, to look for content that can be used to place a virus into a computer.

APIC *See* Application Policy Infrastructure Controller.

APIC-EM *See* Application Policy Infrastructure Controller—Enterprise Module.

Application Centric Infrastructure (ACI) Cisco's data center SDN solution, the concepts of defining policies that the APIC controller then pushes to the switches in the network using the OpFlex protocol, with the partially distributed control plane in each switch building the forwarding table entries to support the policies learned from the controller. It also supports a GUI, a CLI, and APIs.

Application Policy Infrastructure Controller—Enterprise Module (APIC-EM) The software that plays the role of controller in an enterprise network of Cisco devices, in its first version as of the publication of this book, which leaves the distributed routing and switching control plane as is, instead acting as a management and automation platform. It provides robust APIs for network automation and uses CLI (Telnet and SSH) plus SNMP southbound to control the existing routers and switches in an enterprise network.

Application Policy Infrastructure Controller (APIC) The software that plays the role of controller, controlling the flows that the switches create to define where frames are forwarded, in a Cisco data center that uses the Application Centric Infrastructure (ACI) approach, switches, and software.

application programming interface (API) A software mechanism that enables software components to communicate with each other.

application signature With Network Based Application Recognition (NBAR), the definition of a combination of matchable fields that Cisco has identified as being characteristic of a specific application, so that NBAR can be configured by the customer to match an application, while IOS then defines the particulars of that matching.

Application Visibility and Control (AVC) A firewall device with advanced features, including the ability to run many related security features in the same firewall device (IPS, malware detection, VPN termination), along with deep packet inspection with Application Visibility and Control (AVC) and the ability to perform URL filtering versus data collected about the reliability and risk associated with every domain name.

application-specific integrated circuit (ASIC) An integrated circuit (computer chip) designed for a specific purpose or application, often used to implement the functions of a networking device rather than running a software process as part of the device's OS that runs on a general-purpose processor.

AR *See* access rate.

ARP Address Resolution Protocol. An Internet protocol used to map an IP address to a MAC address. Defined in RFC 826.

ARP ACL A configuration feature on Cisco LAN switches that define MAC and IP address pairs that can be used directly for filtering, as well as to be referenced by the Dynamic ARP Inspection feature.

ARP reply An ARP message used to supply information about the sending (origin) host's hardware (Ethernet) and IP addresses as listed in the origin hardware and origin IP address fields. Typically sent in reaction to receipt of an ARP request message.

ARP request An ARP message used to request information from another host located on the same data link, typically listing a known target IP address but an all-zero target hardware address, to ask the host with that target IP address to identify its hardware address in an ARP reply message.

ARP table A list of IP addresses of neighbors on the same VLAN, along with their MAC addresses, as kept in memory by hosts and routers.

ASAv A Cisco ASA firewall software image that runs as a virtual machine rather than on Cisco hardware, intended to be used as a consumer-controlled firewall in a cloud service or in other virtualized environments.

ASIC *See* application-specific integrated circuit.

Assured Forwarding (AF) The name of a grid of 12 DSCP values and a matching grid of per-hop behavior as defined by DiffServ. AF defines four queuing classes and three packet drop priorities within each queuing class. The text names of the 12 DSCP values follow a format of AFXY, where X is the queuing class, and Y is the drop priority.

authentication In security, the verification of the identity of a person or a process. *See also* AAA.

authentication, authorization, and accounting (AAA) server A server that holds security information and provides services related to user login, particularly authentication (is the user who he says he is), authorization (once authenticated, what do we allow the user to do), and accounting (tracking the user).

Authoritative DNS server The DNS server with the record that lists the address that corresponds to a domain name (the A Record) for that domain.

authorization In security, the determination of the rights allowed for a particular user or device. *See also* AAA.

autonomous system (AS) An internetwork that is managed by one organization.

autonomous system number (ASN) A number used by BGP to identify a routing domain, often a single enterprise or organization. As used with EIGRP, a number that identifies the routing processes on routers that are willing to exchange EIGRP routing information with each other.

AutoQoS In Cisco switches and routers, an IOS feature that configures a variety of QoS features with useful settings as defined by the Cisco reference design guide documents.

B

bandwidth The speed at which bits can be sent and received over a link.

bandwidth profile In Metro Ethernet, a contractual definition of the amount of traffic that the customer can send into the service and receive out of the service. Includes a concept called the committed information rate (CIR), which defines the minimum amount of bandwidth (bits per second) the SP will deliver with the service.

Brownfield A term that refers to the choice to add new configuration to hardware and software that are already in use, rather that adding new hardware and software specifically for a new project.

brute-force attack An attack where a malicious user runs software that tries every possible combination of letters, numbers, and special characters to guess a user's password. Attacks of this scale are usually run offline, where more computing resources and time are available.

buffer overflow attack An attack meant to exploit a vulnerability in processing inbound traffic such that the target system's buffers overflow; the target system can end up crashing or inadvertently running malicious code injected by the attacker.

C

cable Internet An Internet access technology that uses a cable TV (CATV) cable, normally used for video, to send and receive data.

cacheable For resources that might be repeatedly requested over time, an attribute that means that the requesting host can keep in storage (cache) a copy of the resource for a specified amount of time.

candidate config With configuration management tools like Ansible, Puppet, and Chef, an updated configuration for a device as it exists in the management tool before the tool has moved the configuration into the device.

carrier Ethernet Per MEF documents, the term for what was formerly called Metro Ethernet, generally referring to any WAN service that uses Ethernet links as the access link between the customer and the service provider.

CDP Cisco Discovery Protocol. A media- and protocol-independent device-discovery protocol that runs on most Cisco-manufactured equipment, including routers, access servers, and switches. Using CDP, a device can advertise its existence to other devices and receive information about other devices on the same LAN or on the remote side of a WAN.

CDP neighbor A device on the other end of some communications cable that is advertising CDP updates.

central office (CO) A term used by telcos to refer to a building that holds switching equipment, into which the telco's cable plant runs so that the telco has cabling from each home and business into that building.

centralized control plane An approach to architecting network protocols and products that places the control plane functions into a centralized function rather than distributing the function across the networking devices.

Chef A popular configuration management application, which uses a server and a pull model with in-device agents.

Chef client Any device whose configuration is being managed by Chef.

Chef Cookbook A set of recipes about the same kinds of work, grouped together for easier management and sharing.

Chef Recipe The Chef logic applied to resources to determine when, how, and whether to act against the resources—analogous to a recipe in a cookbook.

Chef Runlist An ordered list of recipes that should be run against a given device.

Chef server The Chef software that collects all the configuration files and other files used by Chef from different Chef users and then communicates with Chef clients (devices) so that the Chef clients can synchronize their configurations.

CIDR Classless interdomain routing. An RFC-standard tool for global IP address range assignment. CIDR reduces the size of Internet routers' IP routing tables, helping deal with the rapid growth of the Internet. The term *classless* refers to the fact that the summarized groups of networks represent a group of addresses that do not conform to IPv4 classful (Class A, B, and C) grouping rules.

Cisco Access Control Server (ACS) A legacy Cisco product that acts as a AAA server.

Cisco AnyConnect Secure Mobility Client Cisco software product used as client software on user devices to create a client VPN. Commonly referred to as the Cisco VPN client.

Cisco Open SDN Controller (OSC) A former commercial SDN controller from Cisco that is based on the OpenDaylight controller.

Cisco Prime Graphical user interface (GUI) software that utilizes SNMP and can be used to manage your Cisco network devices. The term *Cisco Prime* is an umbrella term that encompasses many different individual software products.

Cisco Prime Infrastructure (PI) The name of Cisco's long-time enterprise network management application.

Cisco Talos Intelligence Group A part of the Cisco Systems company that works to perform security research on an ongoing basis, in part to supply up-to-date data, like virus signatures, that Cisco security products can frequently download.

Cisco VPN client *See* Cisco AnyConnect Secure Mobility Client.

Class of Service (CoS) The informal term for the 3-bit field in the 802.IQ header intended for marking and classifying Ethernet frames for the purposes of applying QoS actions. Another term for Priority Code Point (PCP).

Class Selector (CS) The name of eight DSCP values that all end with binary 000, for the purpose of having eight identifiable DSCP values whose first 3 bits match the eight values used for the older IP Precedence field. Originally used for backward compatibility with IP Precedence, but today the values are often used as just more values to use for packet marking.

classification The process of examining various fields in networking messages in an effort to identify which messages fit into certain predetermined groups (classes).

classless addressing A concept in IPv4 addressing that defines a subnetted IP address as having two parts: a prefix (or subnet) and a host.

client VPN A VPN for which one endpoint is a user device, like a phone, tablet, or PC. Also called a remote access VPN.

clock rate The speed at which a serial link encodes bits on the transmission medium.

clock source On serial links, the device to which the other devices on the link adjust their speed when using synchronous links. With NTP, the external device or NTP server on which a device bases its time.

clocking The process of supplying a signal over a cable, either on a separate pin on a serial cable or as part of the signal transitions in the transmitted signal, so that the receiving device can keep synchronization with the sending device.

Clos network A term for network topology that represents an ideal for a switch fabric and named after Charles Clos, who formalized the definition. Also called a spine-leaf network.

cloud services catalog A listing of the services available in a cloud computing service.

Cloud Services Router (CSR) A Cisco router software image that runs as a virtual machine rather than on Cisco hardware, intended to be used as a consumer-controlled router in a cloud service or in other virtualized environments.

code integrity A software security term that refers to how likely that the software (code) being used is the software supplied by the vendor, unchanged, with no viruses or other changes made to the software.

collapsed core design A campus LAN design in which the design does not use a separate set of core switches in addition to the distribution switches—in effect collapsing the core into the distribution switches.

confidentiality (privacy) Preventing anyone in the middle of the Internet (a.k.a. man in the middle) from being able to read the data.

configuration drift A phenomenon that begins with the idea that devices with similar roles can and should have a similar standard configuration, so when one device's configuration is changed to be different, its configuration is considered to have moved away (drifted) from the standard configuration for a device in that role.

configuration enforcement Another term for configuration monitoring.

configuration management A component of network management focused on creating, changing, removing, and monitoring device configuration.

configuration management tool A class of application that manages data about the configuration of servers, network devices, and other computing nodes, providing consistent means of describing the configurations, moving the configurations into the devices, noticing unintended changes to the configurations, and troubleshooting by easily identifying changes to the configuration files over time.

configuration monitoring With configuration management tools like Ansible, Puppet, and Chef, a process of comparing over time a device's on-device configuration (running-config) versus the text file showing the ideal device configuration listed in the tool's centralized configuration repository. If different, the process can either change the device's configuration or report the issue.

configuration provisioning With configuration management tools like Ansible, Puppet, and Chef, the process of configuring a device to match the configuration as held in the configuration management tool.

configuration template With configuration management tools like Ansible, Puppet, and Chef, a file with variables, for the purpose of having the tool substitute different variable values to create the configuration for a device.

congestion window With TCP, a calculation each TCP receiver does that limits the window it grants to the receiver by shrinking the window in response to the loss of TCP segments.

connection establishment The process by which a connection-oriented protocol creates a connection. With TCP, a connection is established by a three-way transmission of TCP segments.

control plane Functions in networking devices and controllers that directly control how devices perform data plane forwarding, but excluding the data plane processes that work to forward each message in the network.

controller-based networking A style of building computer networks that use a controller that centralizes some features and provides application programming interfaces (APIs) that allow for software interactions between applications and the controller (northbound APIs) and between the controller and the network devices (southbound APIs).

core In computer architecture, an individual processing unit that can execute instructions of a CPU; modern server processors typically have multiple cores, each capable of concurrent execution of instructions.

core design A campus LAN design that connects each access switch to distribution switches, and distribution switches into core switches, to provide a path between all LAN devices.

core layer In a campus LAN design, the switches that connect the distribution layer switches, and to each other, to provide connectivity between the various distribution layer switches.

CRUD In software development, an acronym that refers to the four most common actions taken by a program: Create, Read, Update, and Delete.

customer edge (CE) A term used by service providers, both generally and also specifically in MPLS VPN networks, to refer to the customer device that connects to the SP's network and therefore sits at the edge of the SP's network.

customer premises equipment (CPE) A telco term that refers to equipment on site at the telco customer site (the enterprise's site) that connects to the WAN service provided by the telco.

D

data integrity Verifying that the packet was not changed as the packet transited the Internet.

data model A set of variables and their structures, like lists and dictionaries.

data modeling language Another term for data serialization language.

data plane Functions in networking devices that are part of the process of receiving a message, processing the message, and forwarding the message.

data serialization language A language that includes syntax and rules that provides a means to describes the variables inside applications in a text format, for the purpose of sending that text between applications over a network or storing the data models in a file.

data structure Another term for data model.

declarative policy model A term that describes the approach in an intent-based network (IBN) in which the engineer chooses settings that describe the intended network behavior (the declared policy) but does not command the network with specific configuration commands for each protocol (as would be the case with an imperative policy model).

decrypt/decryption The ability to receive encrypted data and process it to derive the original unencrypted data.

default gateway/default router On an IP host, the IP address of some router to which the host sends packets when the packet's destination address is on a subnet.

delay In QoS, the amount of time it takes for a message to cross a network. Delay can refer to one-way delay (the time required for the message to be sent from the source host to the destination host) or two-way delay (the delay from the source to the destination host and then back again).

demilitarized zone (DMZ) In an Internet edge design at an enterprise, one or more subnets set aside as a place to locate servers that should allow users in the Internet to initiate connections to those servers. The devices in the DMZ typically sit behind a firewall.

denial-of-service (DoS) attack An attack that tries to deplete a system resource so that systems and services crash or become unavailable.

deny An action taken with an ACL that implies that the packet is discarded.

DevNet Cisco's community and resource site for software developers, open to all, with many great learning resources; https://developer.cisco.com.

DHCP Dynamic Host Configuration Protocol. A protocol used by hosts to dynamically discover and lease an IP address, and learn the correct subnet mask, default gateway, and DNS server IP addresses.

DHCP attack Any attack that takes advantage of DHCP protocol messages.

DHCP binding table A table built by the DHCP snooping feature on a switch when it sees messages about a new DHCP lease, with the table holding information about legitimate successful DHCP leases, including the device's IP address, MAC address, switch port, and VLAN.

DHCP chaddr Client hardware address. The original DHCP header field used to identify the DHCP clients; typically includes the client MAC address.

DHCP client Any device that uses DHCP protocols to ask to lease an IP address from a DHCP server or to learn any IP settings from that server.

DHCP client identifier A DHCP header field used to identify a DHCP client, used as a more flexible alternative to the DHCP chaddr field.

DHCP giaddr Gateway IP address. In DHCP, a header field used to identify a router on a subnet, typically an IP address on the DHCP relay agent, so that the DHCP server knows an address to which to send messages in reply to the client.

DHCP option 82 Optional DHCP header fields, as defined in RFC 3046, that provide useful features of use to a device that acts as a DHCP relay agent. The fields allow better relay agent operation and also help prevent various types of DHCP-based attacks.

DHCP relay agent The name of the router IOS feature that forwards DHCP messages from client to servers by changing the destination IP address from 255.255.255.255 to the IP address of the DHCP server.

DHCP server Software that waits for DHCP clients to request to lease IP addresses, with the server assigning a lease of an IP address as well as listing other important IP settings for the client.

DHCP Snooping A switch security feature in which the switch examines incoming DHCP messages and chooses to filter messages that are abnormal and therefore might be part of a DHCP attack.

DHCP Snooping binding table When using DHCP Snooping, a table that the switch dynamically builds by analyzing the DHCP messages that flow through the switch. DHCP Snooping can use the table for part of its filtering logic, with other features, such as Dynamic ARP Inspection and IP Source Guard also using the table.

dictionary attack An attack where a malicious user runs software that attempts to guess a user's password by trying words from a dictionary or word list.

dictionary variable In applications, a single variable whose value is a list of other variables with values, known as key:value pairs.

Differentiated Services (DiffServ) An approach to QoS, originally defined in RFC 2475, that uses a model of applying QoS per classification, with planning of which applications and other traffic types are assigned to each class, with each class given different QoS per-hop behaviors at each networking device in the path.

Differentiated Services Code Point (DSCP) A field existing as the first 6 bits of the ToS byte, as defined by RFC 2474, which redefined the original IP RFC's definition for the IP header ToS byte. The field is used to mark a value in the header for the purpose of performing later QoS actions on the packet.

Digital Subscriber Line (DSL) A public network technology that delivers high bandwidth over conventional telco local-loop copper wiring at limited distances. Typically used as an Internet access technology, connecting a user to an ISP.

distributed control plane An approach to architecting network protocols and products that places some control plane functions into each networking device rather than centralizing the control plane functions in one or a few devices. An example is the use of routing protocols on each router which then work together so that each router learns Layer 3 routes.

distributed denial-of-service (DDoS) attack A DoS attack that is distributed across many hosts under centralized control of an attacker, all targeting the same victim.

distribution layer In a campus LAN design, the switches that connect to access layer switches as the most efficient means to provide connectivity from the access layer into the other parts of the LAN.

DNA Digital Network Architecture—Cisco's software-oriented approach to networking and intent-based networking products and services.

DNA Center Cisco software, delivered by Cisco on a server appliance, that acts as a network management application as well as a being the control for Cisco's software-defined access (SDA) offering.

DNS Domain Name System. An application layer protocol used throughout the Internet for translating host names into their associated IP addresses.

DNS reply In the Domain Name System (DNS), a message sent by a DNS server to a DNS client in response to a DNS request, identifying the IP address assigned to a particular hostname or fully qualified domain name (FQDN).

DNS request In the Domain Name System (DNS), a message sent by a DNS client to a DNS server, listing a hostname or fully qualified domain name (FQDN), asking the server to discover and reply with the IP address associated with that host name or FQDN.

DNS server An application acting as a server for the purpose of providing name resolution services per the Domain Name System (DNS) protocol and worldwide system.

domain-specific language A generic term that refers to an attribute of different languages within computing, for languages created for a specific purpose (domain) rather than a general-purpose language like Python or JavaScript.

DSL Digital subscriber line. Public network technology that delivers high bandwidth over conventional telco local-loop copper wiring at limited distances. Usually used as an Internet access technology connecting a user to an ISP.

DSL modem A device that connects to a telephone line and uses DSL standards to transmit and receive data to/from a telco using DSL.

Dynamic ARP Inspection (DAI) A security feature in which a LAN switch filters a subset of incoming ARP messages on untrusted ports, based on a comparison of ARP, Ethernet, and IP header fields to data gathered in the IP DHCP Snooping binding table and found in any configured ARP ACLs.

E

egress tunnel router (ETR) With LISP, a node at the end of a tunnel that receives an encapsulated message and then de-encapsulates the message.

E-LAN A specific carrier/Metro Ethernet service defined by MEF (MEF.net) that provides a service much like a LAN, with two or more customer sites connected to one E-LAN service in a full mesh so that each device in the E-LAN can send Ethernet frames directly to every other device.

E-Line A specific carrier/metro Ethernet service defined by MEF (MEF.net) that provides a point-to-point topology between two customer devices, much as if the two devices were connected using an Ethernet crossover cable.

enable mode A part of the Cisco IOS CLI in which the user can use the most powerful and potentially disruptive commands on a router or switch, including the ability to then reach configuration mode and reconfigure the router.

enable password A reference to the password configured on the **enable password** *pass-value* command, which defines the password required to reach enable (privileged) mode if the **enable secret** *pass-value* command does not exist.

enable secret A reference to the password configured on the **enable secret** *pass-value* command, which defines the password required to reach enable (privileged) mode.

encrypt/encryption The ability to take data and send the data in a form that is not readable by someone who intercepts this data.

encryption key A secret value used as input to the math formulas used by an encryption process.

End of Row (EoR) switch In a traditional data center design with servers in multiple racks and the racks in multiple rows, a switch placed in a rack at the end of the row, intended to be cabled to all the Top of Rack (ToR) switches in the same row, to act as a distribution layer switch for the switches in that row.

endpoint group In ACI, a set (group) of VMs, containers, physical servers, or other endpoints in an ACI data center that should receive the same policy treatment.

Endpoint ID (EID) With LISP, a number that identifies the endpoint.

err-disable recovery Cisco switches can place ports in a nonworking state called "err-disabled" in reaction to a variety of events, and by default, to leave the port in the nonworking err-disabled state until the engineer takes action to recover from the issue. The err-disable recovery configuration feature includes settings to direct the switch to automatically revert away from the err-disabled state, back to a working state, after a period of time.

error detection The process of discovering whether a data-link level frame was changed during transmission. This process typically uses a Frame Check Sequence (FCS) field in the data-link trailer.

error disabled (err-disable) An interface state on LAN switches that can be the result of one of many security violations.

error recovery The process of noticing when some transmitted data was not successfully received and resending the data until it is successfully received.

Ethernet access link A WAN access link (a physical link between a service provider and its customer) that happens to use Ethernet.

Ethernet LAN Service Another term for E-LAN; *see also* E-LAN.

Ethernet Line Service Another term for E-Line; *see also* E-Line.

Ethernet Tree Service Another term for E-Tree; *see also* E-Tree.

Ethernet Virtual Connection (EVC) A concept in carrier/Metro Ethernet that defines which customer devices can send frames to each other over the Ethernet WAN service; includes E-Line, E-LAN, and E-Tree EVCs.

Ethernet WAN A general and informal term for any WAN service that uses Ethernet links as the access link between the customer and the service provider.

E-Tree A specific carrier/metro Ethernet service defined by MEF (MEF.net) that provides a rooted multipoint service, in which the root site can send frames directly to all leaves, but the leaf sites can send only to the root site.

Expedited Forwarding (EF) The name of a particular DSCP value, as well as the term for one per-hop behavior as defined by DiffServ. The value, decimal 46, is marked for packets to which the networking devices should apply certain per-hop behaviors, like priority queuing.

exploit A means of taking advantage of a vulnerability to compromise something.

extended access list A list of IOS **access-list** global configuration commands that can match multiple parts of an IP packet, including the source and destination IP address and TCP/UDP ports, for the purpose of deciding which packets to discard and which to allow through the router.

F

fabric In SDA, the combination of overlay and underlay that together provide all features to deliver data across the network with the desired features and attributes.

fabric border node In SDA, a switch that connects to devices outside SDA's control—for example, switches that connect to the WAN routers or to an ACI data center.

fabric control node In SDA, a switch that performs special functions for the underlay (LISP), requiring more CPU and memory.

fabric edge node In SDA, a switch that connects to endpoint devices.

fiber Internet A general term for any Internet access technology that happens to use fiber-optic cabling. It often uses Ethernet protocols on the fiber link.

filter Generally, a process or a device that screens network traffic for certain characteristics, such as source address, destination address, or protocol. This process determines whether to forward or discard that traffic based on the established criteria.

firewall A device that forwards packets between the less secure and more secure parts of the network, applying rules that determine which packets are allowed to pass and which are not.

First Hop Redundancy Protocol (FHRP) A class of protocols that includes HSRP, VRRP, and GLBP, which allows multiple redundant routers on the same subnet to act as a single default router (first-hop router).

flash memory A type of read/write permanent memory that retains its contents even with no power applied to the memory, and uses no moving parts, making the memory less likely to fail over time.

flow control The process of regulating the amount of data sent by a sending computer toward a receiving computer. Several flow control mechanisms exist, including TCP flow control, which uses windowing.

forward acknowledgment A process used by protocols that do error recovery, in which the number that acknowledges data lists the next data that should be sent, not the last data that was successfully received.

forwarding plane A synonym for data plane. *See also* data plane.

FTP File Transfer Protocol. An application protocol, part of the TCP/IP protocol stack, used to transfer files between network nodes. FTP is defined in RFC 959.

FTP active mode One of two modes of operation for FTP connections (the other being passive mode) that dictates how the FTP data mode connection is established. In active mode, the FTP client listens on a port, it identifies that port to the server, and the server initiates the TCP connection.

FTP client An application that can connect to an FTP server for the purpose of transferring copies of files to and from the server.

FTP control connection A TCP connection initiated by an FTP client to an FTP server for the purpose of sending FTP commands that direct the activities of the connection.

FTP data connection A TCP connection created by an FTP client and server for the purpose of transferring data.

FTP over TLS An FTP standard defined by RFC 4217, also known as FTP Secure (FTPS), which adds a variety of security features to the somewhat insecure original FTP standard (RFC 957), including the addition of the encryption of all data as well as username/password information using Transport Layer Security (TLS).

FTP passive mode One of two modes of operation for FTP connections (the other being active mode) that dictates how the FTP data mode connection is established. In passive mode, the FTP client declares the use of passive mode, causing the server to choose and identify a new listening port, with the client establishing a TCP connection to that port.

FTP server An application that runs and waits for FTP clients to connect to it over TCP port 21 to support the client's commands to transfer copies of files to and from the server.

FTPS FTP Secure. Common term for FTP over TLS.

full mesh From a topology perspective, any topology that has two or more devices, with each device being able to send frames to every other device.

G

Gateway Load Balancing Protocol (GLBP) A Cisco-proprietary protocol that allows two (or more) routers to share the duties of being the default router on a subnet, with an active/active model, with all routers actively forwarding off-subnet traffic for some hosts in the subnet.

Generic Routing Encapsulation (GRE) A protocol, defined in RFC 2784, that defines the headers used when creating a site-to-site VPN tunnel. The protocol defines the use of a normal IP header, called the Delivery Header, and a GRE header that the endpoints use to create and manage traffic over the GRE tunnel.

Git An open-source version control application, widely popular for version control in software development and for other uses, like managing network device configurations.

GitHub A software-as-a-service application that implements Git.

gratuitous ARP An ARP Reply not sent as a reaction to an ARP request message, but rather as a general announcement informing other hosts of the values of the sending (origin) host's addresses.

GRE tunnel A site-to-site VPN idea, in which the endpoints act as if a point-to-point link (the tunnel) exists between the sites, while actually encapsulating packets using GRE standards.

greenfield A term that refers to the installation of new equipment for a project rather than adding configuration to existing in-use hardware and software.

H

host (context: DC) In a virtualized server environment, the term used to refer to one physical server that is running a hypervisor to create multiple virtual machines.

Hot Standby Router Protocol (HSRP) A Cisco-proprietary protocol that allows two (or more) routers to share the duties of being the default router on a subnet, with an active/standby model, with one router acting as the default router and the other sitting by waiting to take over that role if the first router fails.

HSRP active A Hot Standby Router Protocol (HSRP) state in which the router actively supports the forwarding of off-subnet packets for hosts in that subnet.

HSRP standby A Hot Standby Router Protocol (HSRP) state in which the router does not currently support the forwarding of off-subnet packets for hosts in that subnet, instead waiting for the currently active router to fail before taking over that role.

HTML Hypertext Markup Language. A simple document-formatting language that uses tags to indicate how a given part of a document should be interpreted by a viewing application, such as a web browser.

HTTP Hypertext Transfer Protocol. The protocol used by web browsers and web servers to transfer files, such as text and graphic files.

HTTP verb The action defined in an HTTP request message.

hub and spoke From a topology perspective, any topology that has a device that can send messages to all other devices (the hub), with one or more spoke devices that can send messages only to the hub. Also called point-to-multipoint.

hyperthreading The name of Intel's multithreading technology.

hypervisor Software that runs on server hardware to create the foundations of a virtualized server environment primarily by allocating server hardware components like CPU core/threads, RAM, disk, and network to the VMs running on the server.

I

IANA The Internet Assigned Numbers Authority. An organization that owns the rights to assign many operating numbers and facts about how the global Internet works, including public IPv4 and IPv6 addresses. *See also* ICANN.

ICANN The Internet Corporation for Assigned Names and Numbers. An organization appointed by IANA to oversee the distributed process of assigning public IPv4 and IPv6 addresses across the globe.

imperative policy model A term that describes the approach in traditional networks in which the engineer chooses configuration settings for each control and data plane protocol (the imperative commands) that dictate specifically how the devices act. This model acts in contrast to the newer declarative policy model and intent-based networking (IBN).

Infrastructure as a Service (IaaS) A cloud service in which the service consists of a virtual machine that has defined computing resources (CPUs, RAM, disk, and network) and may or may not be provided with an installed OS.

ingress tunnel router (ITR) With LISP, the node that receives an unencapsulated message and encapsulates the message.

inside global For packets sent to and from a host that resides inside the trusted part of a network that uses NAT, a term referring to the IP address used in the headers of those packets when those packets traverse the global (public) Internet.

inside local For packets sent to and from a host that resides inside the trusted part of a network that uses NAT, a term referring to the IP address used in the headers of those packets when those packets traverse the enterprise (private) part of the network.

integrity In data transfers, means that the network administrator can determine that the information has not been tampered with in transit.

intent-based networking (IBN) An approach to networking in which the system gives the operator the means to express business intent, with the networking system then determining what should be done by the network, activating the appropriate configuration, and monitoring (assuring) the results.

intercloud exchange A WAN service that provides connectivity between public cloud providers and their customers so that customers can install and keep the WAN connections, even when migrating from one cloud provider to another.

Internet access technology Any technology that an ISP offers that allows its customers to send and receive data to/from the ISP, including serial links, Frame Relay, MPLS, Metro Ethernet, DSL, cable, and fiber Internet.

Internet edge The part of the topology of the Internet that sits between an ISP and the ISP's customer.

Internet service provider A company or organization that provides Internet services to customers; the company may have a heritage as a telco, WAN service provider, or cable company.

internetwork operating system (IOS) *See* IOS.

intrusion detection system (IDS) A security function that examines more complex traffic patterns against a list of both known attack signatures and general characteristics of how attacks can be carried out, rating each perceived threat and reporting the threats.

intrusion prevention system (IPS) A security function that examines more complex traffic patterns against a list of both known attack signatures and general characteristics of how attacks can be carried out, rating each perceived threat, and reacting to prevent the more significant threats. *See also* IPS.

IOS Cisco operating system software that provides the majority of a router's or switch's features, with the hardware providing the remaining features.

IOS feature set A set of related features that can be enabled on a router to enable certain functionality. For example, the Security feature set would enable the ability to have the router act as a firewall in the network.

IOS File System (IFS) A file system created by a Cisco device that uses IOS.

IOS image A file that contains the IOS.

IP Precedence (IPP) In the original definition of the IP header's Type of Service (ToS) byte, the first 3 bits of the ToS byte, used for marking IP packets for the purpose of applying QoS actions.

IPS *See* intrusion prevention system.

IPsec The term referring to the IP Security protocols, which is an architecture for providing encryption and authentication services, usually when creating VPN services through an IP network.

ISDN Integrated Services Digital Network. A communication protocol offered by telephone companies that permits telephone networks to carry data, voice, and video.

Iterative DNS server A DNS server that will answer DNS requests directly but will not take on the extra work to recursively send other DNS messages to find the answer.

J

JavaScript A programming language popular for building dynamic web pages, commonly used to run scripts on a web client.

Jinja2 A text-based language used to define templates, with text plus variables; used by Ansible for templates.

jitter The variation in delay experienced by successive packets in a single application flow.

JSON (JavaScript Object Notation) A popular data serialization language, originally used with the JavaScript programming language, and popular for use with REST APIs.

JSON array A part of a set of JSON text that begins and ends with a matched set of square brackets that contain a list of values.

JSON object A part of a set of JSON text that begins and ends with a matched set of curly brackets that contain a set of key:value pairs.

K–L

key:value pair In software, one variable name (key) and its value, separated by a colon in some languages and data serialization languages.

keyboard, video, mouse (KVM) Three components of a typical desktop computer that are typically not included in a modern server because the server is installed and managed remotely.

KVM (Red Hat) Kernel-Based Virtual Machine (KVM), a server virtualization/hypervisor product from the Red Hat company.

leaf In an ACI network design, a switch that connects to spine switches and to endpoints, but not to other leaf switches, so that the leaf can forward frames from an endpoint to a spine, which then delivers the frame to some other leaf switch.

library In software, a collection of programs packaged so that it can be posted as available in a software repository, found by others, and installed as one entity, as a means to make it easier to share code.

LISP Locator/ID Separation Protocol. A protocol, defined in RFC 6830, that separates the concepts and numbers used to identify an endpoint (the endpoint identifier) versus identifying the location of the endpoint (routing locator).

LISP mapping database With LISP, the table that contains mapped pairs of endpoint identifiers and routing locators.

LISP Routing Locator (RLOC) With LISP, a value that identifies the location of an endpoint, typically the address of the egress device.

list variable In applications, a single variable whose value is a list of values, rather than a simple value.

LLDP Link Layer Discovery Protocol. An IEEE standard protocol (IEEE 802.1AB) that defines messages, encapsulated directly in Ethernet frames so they do not rely on a working IPv4 or IPv6 network, for the purpose of giving devices a means of announcing basic device information to other devices on the LAN. It is a standardized protocol similar to Cisco Discovery Protocol (CDP).

local loop A line from the premises of a telephone subscriber to the telephone company CO.

local username A username (with matching password), configured on a router or switch. It is considered local because it exists on the router or switch, and not on a remote server.

log message A message generated by any computer, but including Cisco routers and switches, for which the device OS wants to notify the owner or administrator of the device about some event.

loss A reference to packets in a network that are sent but do not reach the destination host.

low latency queue In Cisco queuing systems, a queue from which the queue scheduling algorithm always takes packets next if the queue holds any packets. This scheduling choice means that packets in this queue spend little time in the queue, achieving low delay (latency) as well as low jitter.

Low Latency Queuing (LLQ) The name of a queuing system that can be enabled on Cisco routers and switches by which messages sensitive to latency and jitter are placed in a queue that is always serviced first, resulting in low latency and jitter for those messages.

LTE Literally, Long Term Evolution, but this term is used as a word itself to represent the type of wireless 4G technology that allows faster speeds than the original 4G specifications.

M

malware Malicious software.

Management Information Base (MIB) The data structures defined by SNMP to define a hierarchy (tree) structure with variables at the leaves of the tree, so that SNMP messages can reference the variables.

management plane Functions in networking devices and controllers that control the devices themselves but that do not impact the forwarding behavior of the devices like control plane protocols do.

man-in-the-middle attack An attack where an attacker manages to position a machine on the network such that it is able to intercept traffic passing between target hosts.

marking The process of changing one of a small set of fields in various network protocol headers, including the IP header's DSCP field, for the purpose of later classifying a message based on that marked value.

markup language A language that provides conventions to tag text to identify the type of text, which allows application of different treatments to different types of text.

match/action logic The basic logic done by a networking element: to receive incoming messages, to match fields in the message, to then use logic based on those matches to take action against the message, and to then forward the message.

MD5 hash A specific mathematical algorithm intended for use in various security protocols. In the context of Cisco routers and switches, the devices store the MD5 hash of certain passwords, rather than the passwords themselves, in an effort to make the device more secure.

Metro Ethernet The original term used for WAN service that used Ethernet links as the access link between the customer and the service provider.

MIB *See* Management Information Base.

MIB view A concept in SNMPv3 that identifies a subset of an SNMP agent's MIB for the purpose of limiting access to some parts of the MIB to certain SNMP managers.

mitigation technique A method to counteract or prevent threats and malicious activity.

modem Modulator-demodulator. A device that converts between digital and analog signals so that a computer may send data to another computer using analog telephone lines. At the source, a modem converts digital signals to a form suitable for transmission over analog communication facilities. At the destination, the analog signals are returned to their digital form.

MPLS *See* Multiprotocol Label Switching.

MPLS experimental bits A 3-bit field in the MPLS label used for QoS marking.

MPLS VPN A WAN service that uses MPLS technology, with many customers connecting to the same MPLS network, but with the VPN features keeping each customer's traffic separate from others.

MTU Maximum transmission unit. The maximum packet size, in bytes, that a particular interface can handle.

multifactor authentication A technique that uses more than one type of credential to authenticate users.

multipoint A topology with more than two devices in it (in contrast to a point-to-point topology, which has exactly two devices). Without any further context, the term *multipoint* does not define whether all devices in the topology can send messages directly to each other (full mesh) or not (partial mesh).

Multiprotocol BGP (MPBGP) A particular set of BGP extensions that allows BGP to support multiple address families, which when used to create an MPLS VPN service gives the SP the method to advertise the IPv4 routes of many customers while keeping those route advertisements logically separated.

Multiprotocol Label Switching (MPLS) A WAN technology used to create an IP-based service for customers, with the service provider's internal network performing forwarding based on an MPLS label rather than the destination IP address.

multithreading In computer architecture, a process of maximizing the use of a processor core by sharing an individual core among multiple programs, taking advantage of the typical idle times for the core while it waits on various other tasks like memory reads and writes.

N

name resolution The process by which an IP host discovers the IP address associated with a host name, often involving sending a DNS request to a DNS server, with the server supplying the IP address used by a host with the listed host name.

name server A server connected to a network that resolves network names into network addresses.

named access list An ACL that identifies the various statements in the ACL based on a name rather than a number.

NAT Network Address Translation. A mechanism for reducing the need for globally unique IP addresses. NAT allows an organization with addresses that are not globally unique to connect to the Internet, by translating those addresses into public addresses in the globally routable address space.

NAT overload Another term for Port Address Translation (PAT). One of several methods of configuring NAT, in this case translating TCP and UDP flows based on port numbers in addition to using one or only a few inside global addresses.

National Institute of Standards and Technology (NIST) A U.S. federal agency that develops national standards, including standards for cloud computing.

NBI *See* northbound API.

Nest In JSON, the concept that values can contain objects and arrays so that each object can contain other objects and arrays in a myriad of combinations.

Network Based Application Recognition (NBAR) A Cisco router feature that looks at message details beyond the Layer 2, 3, and 4 headers to identify over 1000 different classifications of packets from different applications.

Network Management System (NMS) Software that manages the network, often using SNMP and other protocols.

Network Time Protocol (NTP) A protocol used to synchronize time-of-day clocks so that multiple devices use the same time of day, which allows log messages to be more easily matched based on their timestamps.

Next-generation firewall (NGFW) A firewall device with advanced features, including the ability to run many related security features in the same firewall device (IPS, malware detection, VPN termination), along with deep packet inspection with Application Visibility and Control (AVC) and the ability to perform URL filtering versus data collected about the reliability and risk associated with every domain name.

Next-generation IPS (NGIPS) An IPS device with advanced features, including the capability to go beyond a comparison to known attack signatures to also look at contextual data, including the vulnerabilities in the current network, the capability to monitor for new zero-day threats, with frequent updates of signatures from the Cisco Talos security research group.

Nexus 1000v A Cisco Nexus data center switch that runs as a software-only virtual switch inside one host (one hardware server), to provide switching features to the virtual machines running on that host.

NMS Network Management Station. The device that runs network management software to manage network devices. SNMP is often the network management protocol used between the NMS and the managed device.

northbound API In the area of SDN, a reference to the APIs that a controller supports that gives outside programs access to the services of the controller; for instance, to supply information about the network or to program flows into the network. Also called a northbound interface.

northbound interface Another term for northbound API. *See also* northbound API.

notification community An SNMP community (a value that acts as a password), defined on an SNMP manager, which then must be supplied by any SNMP agent that that sends the manager any unsolicited SNMP notifications (like SNMP Trap and Notify requests).

NTP client Any device that attempts to use the Network Time Protocol (NTP) to synchronize its time by adjusting the local device's time based on NTP messages received from a server.

NTP client/server mode A mode of operation with the Network Time Protocol (NTP) in which the device acts as both an NTP client, synchronizing its time with some servers, and as an NTP server, supplying time information to clients.

NTP primary server A term defined in NTP RFCs 1305 and 5905 to refer to devices that act as NTP servers alone, with a stratum 1 external clock source.

NTP secondary server A term defined in NTP RFCs 1305 and 5905 to refer to devices that act as NTP clients and servers, synchronizing as a client to some NTP server, and then acting as an NTP server for other NTP clients.

NTP server Any device that uses Network Time Protocol (NTP) to help synchronize time-of-day clocks for other devices by telling other devices its current time.

NTP synchronization The process with the Network Time Protocol (NTP) by which different devices send messages, exchanging the devices' current time-of-day clock information and other data, so that some devices adjust their clocks to the point that the time-of-day clocks list the same time (often accurate to at least the same second).

NVRAM Nonvolatile RAM. A type of random-access memory (RAM) that retains its contents when a unit is powered off.

O

ODL *See* OpenDaylight.

OID Object identifier. Used to uniquely describe an MIB variable in the SNMP database. This is a numeric string that identifies the variable uniquely and also describes where the variable exists in the MIB tree structure.

on-demand self-service One of the five key attributes of a cloud computing service as defined by NIST, referring to the fact that the consumer of the server can request the service, with the service being created without any significant delay and without waiting on human intervention.

one-way delay The elapsed time from sending the first bit of data at the sending device until the last bit of that data is received on the destination device.

ONF *See* Open Networking Foundation.

on-premises An alternate term for private cloud. *See also* private cloud.

Open Networking Foundation A consortium of SDN users and vendors who work together to foster the adoption of open SDN in the marketplace.

OpenDaylight An open-source SDN controller, created by an open-source effort of the OpenDaylight project under the Linux foundation, built with the intent to have a common SDN controller code base from which vendors could then take the code and add further features and support to create SDN controller products.

OpenFlow The open standard for Software-Defined Networking (SDN) as defined by the Open Networking Foundation (ONF), which defines the OpenFlow protocol as well as the concept of an abstracted OpenFlow virtual switch.

operational management A component of network management focused on extracting data about the network from the network devices, analyzing that data, and providing the data to operations staff.

OpFlex The southbound protocol used by the Cisco ACI controller and the switches it controls.

ordered data transfer A networking function, included in TCP, in which the protocol defines how the sending host should number the data transmitted, defines how the receiving device should attempt to reorder the data if it arrives out of order, and specifies to discard the data if it cannot be delivered in order.

origin hardware address In both an ARP request and reply message, the field intended to be used to list the sender (origin) device's hardware address, typically an Ethernet LAN address.

origin IP address In both an ARP request and reply message, the field intended to be used to list the sender (origin) device's IP address.

outside global With source NAT, the one address used by the host that resides outside the enterprise, which NAT does not change, so there is no need for a contrasting term.

overlay In SDA, the combination of VXLAN tunnels between fabric edge nodes as a data plane for forwarding frames, plus LISP for the control plane for the discovery and registration of endpoint identifiers.

P

partial mesh A network topology in which more than two devices could physically communicate, but by choice, only a subset of the pairs of devices connected to the network is allowed to communicate directly.

password guessing An attack where a malicious user simply makes repeated attempts to guess a user's password.

per-hop behavior (PHB) The general term used to describe the set of QoS actions a device can apply to a message from the time it enters a networking device until the device forwards the message. PHBs include classification, marking, queuing, shaping, policing, and congestion avoidance.

permit An action taken with an ACL that implies that the packet is allowed to proceed through the router and be forwarded.

pharming An attack that compromises name services to silently redirect users toward a malicious site.

phishing An attack technique that sends specially crafted emails to victims in the hope that the users will follow links to malicious websites.

Platform as a Service (PaaS) A cloud service intended for software developers as a development platform, with a variety of tools useful to developers already installed so that developers can focus on developing software rather than on creating a good development environment.

PoE Power over Ethernet. Both a generalized term for any of the standards that supply power over an Ethernet link, as well as a specific PoE standard as defined in the IEEE 802.3af amendment to the 802.3 standard.

point of presence (PoP) A term used for a service provider's (SP) perspective to refer to a service provider's installation that is purposefully located relatively near to customers, with several spread around major cities, so that the distance from each customer site to one of the SP's PoPs is short.

point-to-multipoint *See* hub and spoke.

point-to-point From a topology perspective, any topology that has two and only two devices that can send messages directly to each other.

policing A QoS tool that monitors the bit rate of the messages passing some point in the processing of a networking device, so that if the bit rate exceeds the policing rate for a period of time, the policer can discard excess packets to lower the rate.

policing rate The bit rate at which a policer compares the bit rate of packets passing through a policing function, for the purpose of taking a different action against packets that conform (are under) to the rate versus those that exceed (go over) the rate.

policy model In both ACI and other intent-based networks (IBNs), the operational conventions (model) that combine policies of what the network will provide to grouped sets of network endpoints (endpoint groups) to create a contract for what the network will provide.

port (Multiple definitions) (1) In TCP and UDP, a number that is used to uniquely identify the application process that either sent (source port) or should receive (destination port) data. (2) In LAN switching, another term for switch interface.

Port Address Translation (PAT) A NAT feature in which one inside global IP address supports over 65,000 concurrent TCP and UDP connections.

port number A field in a TCP or UDP header that identifies the application that either sent (source port) or should receive (destination port) the data inside the data segment.

port security A Cisco switch feature in which the switch watches Ethernet frames that come in an interface (a port), tracks the source MAC addresses of all such frames, and takes a security action if the number of different such MAC addresses is exceeded.

port-scanner Jargon that refers to a security vulnerability during the time between the day in which the vulnerability was discovered, until the vendor or open-source group responsible for that software can develop a fix and make it public.

power budget With PoE, data and calculations about the amount of power expected to be used by the various powered devices (PDs), the numbers of devices expected to connect to each switch, versus the amount of power available to PoE based on the capacity of the power supplies in the switches.

power class In various PoE standards, a designation that can be sensed/identified via different discovery processes, with the class defining the maximum amount of power the powered device (PD) would like to receive over the Ethernet link.

Power over Ethernet (PoE) Both a generalized term for any of the standards that supply power over an Ethernet link and a specific PoE standard as defined in the IEEE 802.3af amendment to the 802.3 standard.

Power over Ethernet Plus (PoE+) A specific PoE standard as defined in the IEEE 802.3at amendment to the 802.3 standard, which uses two wire pairs to supply power with a maximum of 30 watts as supplied by the PSE.

power sourcing equipment (PSE) With any Power over Ethernet standard, a term that refers to the device supplying the power over the cable, which is then used by the powered device (PD) on the other end of the cable.

powered device (PD) With any Power over Ethernet standard, a term that refers to the device that receives or draws its power over the Ethernet cable, with the power being supplied by the power sourcing equipment (PSE) on the other end of the cable.

Priority Code Point (PCP) The formal term for the 3-bit field in the 802.IQ header intended for marking and classifying Ethernet frames for the purposes of applying QoS actions. Another term for Class of Service (CoS).

priority queue In Cisco queuing systems, another term for a low latency queue (LLQ).

private cloud A cloud computing service in which a company provides its own IT services to internal customers inside the same company but by following the practices defined as cloud computing.

private IP network Any of the IPv4 Class A, B, or C networks as defined by RFC 1918, intended for use inside a company but not used as public IP networks.

private key A secret value used in public/private key encryption systems. Either encrypts a value that can then be decrypted using the matching public key, or decrypts a value that was previously encrypted with the matching public key.

programmable network A computer network which provides programmatic interfaces that allow automation applications to change and interrogate the configuration of network devices.

provider edge (PE) A term used by service providers, both generally and also specifically in MPLS VPN networks, to refer to the SP device in a point of presence (PoP) that connects to the customer's network and therefore sits at the edge of the SP's network.

public cloud A cloud computing service in which the cloud provider is a different company than the cloud consumer.

public key A publicly available value used in public/private key encryption systems. Either encrypts a value that can then be decrypted using the matching private key, or decrypts a value that was previously encrypted with the matching private key.

pull model With configuration management tools, a practice by which an agent representing the device requests configuration data from the centralized configuration management tool, in effect pulling the configuration to the device.

Puppet A popular configuration management application, which can be used with or without a server, using a pull model in which agents request details and pull configuration into devices, with the capability to manage network device configurations.

Puppet manifest A human-readable text file on the Puppet master, using a language defined by Puppet, used to define the desired configuration state of a device.

Puppet master Another term for Puppet server. *See also* Puppet server.

Puppet server The Puppet software that collects all the configuration files and other files used by Puppet from different Chef users and then communicates with Puppet agents (devices) so that the agents can synchronize their configurations.

Push model With configuration management tools, a practice by which the centralized configuration management tool software initiates the movement of configuration from that node to the device that will be configured, in effect pushing the configuration to the device.

Python A programming language popular as a first language to learn and also popular for network automation tasks.

Python dictionary A Python variable like a JSON dictionary, containing a set of key:value pairs.

Python list A Python variable like a JSON array, containing a list of values.

Q–R

Quality of Experience (QoE) The users' perception of the quality of their experience in using applications in the network.

Quality of Service (QoS) The performance of a message, or the messages sent by an application, in regard to the bandwidth, delay, jitter, or loss characteristics experienced by the message(s).

queuing The process by which networking devices hold packets in memory while waiting on some constrained resource; for example, when waiting for the outgoing interface to become available when too many packets arrive in a short period of time.

RADIUS A security protocol often used for user authentication, including being used as part of the IEEE 802.lx messages between an 802.lx authenticator (typically a LAN switch) and a AAA server.

RAM Random-access memory. A type of volatile memory that can be read and written by a microprocessor.

rapid elasticity One of the five key attributes of a cloud computing service as defined by NIST, referring to the fact that the cloud service reacts to requests for new services quickly, and it expands (is elastic) to the point of appearing to be a limitless resource.

read-only community An SNMP community (a value that acts as a password), defined on an SNMP agent, which then must be supplied by any SNMP manager that sends the agent any messages asking to learn the value of a variable (like SNMP Get and GetNext requests).

read-write community An SNMP community (a value that acts as a password), defined on an SNMP agent, which then must be supplied by any SNMP manager that sends the agent any messages asking to set the value of a variable (like SNMP Set requests).

reconnaissance attack An attack crafted to discover as much information about a target organization as possible; the attack can involve domain discovery, ping sweeps, port scans, and so on.

recursive DNS server A DNS server that, when asked for information it does not have, performs a repetitive (recursive) process to ask other DNS servers in sequence, hoping to find the DNS server that knows the information.

reflection attack An attack that uses spoofed source addresses so that a destination machine will reflect return traffic to the attack's target; the destination machine is known as the reflector.

remote access VPN A VPN for which one endpoint is a user device, such as a phone, tablet, or PC, typically created dynamically, and often using TLS. Also called a client VPN.

Representational State Transfer (REST) A type of API that allows two programs that reside on separate computers to communicate, with a set of six primary API attributes as defined early in this century by its creator, Roy Fielding. The attributes include client/server architecture, stateless operation, cachability, uniform interfaces, layered, and code-on-demand.

resource pooling One of the five key attributes of a cloud computing service as defined by NIST, referring to the fact that the cloud provider treats its resources as a large group (pool) of resources that its cloud management systems then allocate dynamically based on self-service requests by its customers.

REST *See* Representational State Transfer.

REST API Any API that uses the rules of Representational State Transfer (REST).

RESTful API A turn of phrase that means that the API uses REST rules.

RFC Request For Comments. A document used as the primary means for communicating information about the TCP/IP protocols. Some RFCs are designated by the Internet Architecture Board (IAB) as Internet standards, and others are informational. RFCs are available online from numerous sources, including www.rfc-editor.org.

root DNS server A small number of DNS servers worldwide that provide name resolution for the root zone of DNS, providing information about servers that know details about top-level domains (TLDs) such as .com, .org, .edu, and so on.

round robin A queue scheduling algorithm in which the scheduling algorithm services one queue, then the next, then the next, and so on, working through the queues in sequence.

Round Trip Time (RTT) The time it takes a message to go from the original sender to the receiver, plus the time for the response to that message to be sent back.

round-trip delay The elapsed time from sending the first bit of data at the sending device until the last bit of that data is received on the destination device, plus the time waiting for the destination device to form a reply, plus the elapsed time for that reply message to arrive back to the original sender.

route redistribution A method by which two routing protocol processes running in the same device can exchange routing information, thereby causing a route learned by one routing protocol to then be advertised by another.

routed access layer A design choice in which all the switches, including the access layer switches that connect directly to endpoint devices, all use Layer 3 switching so that they route packets.

Router on a Stick (ROAS) Jargon to refer to the Cisco router feature of using VLAN trunking on an Ethernet interface, which then allows the router to route packets that happen to enter the router on that trunk and then exit the router on that same trunk, just on a different VLAN.

S

SBI *See* Southbound API.

scalable group In SDA, the concept of a set of related users that should have the equivalent security access.

scalable group tag (SGT) In SDA, a value assigned to the users in the same security group.

Secure Shell (SSH) A TCP/IP application layer protocol that supports terminal emulation between a client and server, using dynamic key exchange and encryption to keep the communications private.

Secure Sockets Layer (SSL) A deprecated security protocol that was formerly used to secure networks and was commonly integrated into web browsers to provide encryption and authentication services between the browser and a website.

segment (Multiple definitions) (1) In TCP, a term used to describe a TCP header and its encapsulated data (also called an L4PDU). (2) Also in TCP, the set of bytes formed when TCP breaks a large chunk of data given to it by the application layer into smaller pieces that fit into TCP segments. (3) In Ethernet, either a single Ethernet cable or a single collision domain (no matter how many cables are used).

service provider (SP) A company that provides a service to multiple customers. Used most often to refer to providers of private WAN services and Internet services. *See also* Internet service provider.

session key With encryption, a secret value that is known to both parties in a communication, used for a period of time, which the endpoints use when encrypting and decrypting data.

SFTP SSH File Transfer Protocol. A file transfer protocol that assumes a secure channel, such as an encrypted SSH connection, which then provides the means to transfer files over the secure channel.

shaping A QoS tool that monitors the bit rate of the messages exiting networking devices, so that if the bit rate exceeds the shaping rate for a period of time, the shaper can queue the packets, effectively slowing down the sending rate to match the shaping rate.

shaping rate The bit rate at which a shaper compares the bit rate of packets passing through the shaping function, so that when the rate is exceeded, the shaper enables the queuing of packets, resulting in slowing the bit rate of the collective packets that pass through the shaper, so the rate of bits getting through the shaper does not exceed the shaping rate.

shared key A reference to a security key whose value is known (shared) by both the sender and receiver.

shared port With 802.lw RSTP, a port type that is determined by the fact that the port uses half duplex, which could then imply a shared LAN as created by a LAN hub.

Simple Network Management Protocol (SNMP) An Internet standard protocol for managing devices on IP networks. It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention.

simple variable In applications, a variable that has a single value of a simple type, such as text and integer or floating-point numbers.

single point of failure In a network, a single device or link that, if it fails, causes an outage for a given population of users.

site-to-site VPN The mechanism that allows all devices at two different sites to communicate securely over some unsecure network like the Internet, by having one device at each site perform encryption/decryption and forwarding for all the packets sent between the sites.

sliding windows For protocols such as TCP that allow the receiving device to dictate the amount of data the sender can send before receiving an acknowledgment—a concept called a *window*—a reference to the fact that the mechanism to grant future windows is typically just a number that grows upward slowly after each acknowledgment, sliding upward.

SNMP *See* Simple Network Management Protocol.

SNMP agent Software that resides on the managed device and processes the SNMP messages sent by the Network Management Station (NMS).

SNMP community A simple password mechanism in SNMP in which either the SNMP agent or manager defines a community string (password), and the other device must send that same password value in SNMP messages, or the messages are ignored. *See also* read-only community, read-write community, and notification community.

SNMP Get Message used by SNMP to read from variables in the MIB.

SNMP Inform An unsolicited SNMP message like a Trap message, except that the protocol requires that the Inform message needs to be acknowledged by the SNMP manager.

SNMP manager Typically a Network Management System (NMS), with this term specifically referring to the use of SNMP and the typical role of the manager, which retrieves status information with SNMP Get requests, sets variables with the SNMP Set requests, and receives unsolicited notifications from SNMP agents by listening for SNMP Trap and Notify messages.

SNMP Set SNMP message to set the value in variables of the MIB. These messages are the key to an administrator configuring the managed device using SNMP.

SNMP Trap An unsolicited SNMP message generated by the managed device, and sent to the SNMP manager, to give information to the manager about some event or because a measurement threshold has been passed.

SNMPv2c A variation of the second version of SNMP. SNMP Version 2 did not originally support communities; the term *SNMPv2c* refers to SNMP version 2 with support added for SNMP communities (which were part of SNMPvl).

SNMPv3 The third version of SNMP, with the notable addition of several security features as compared to SNMPv2c, specifically message integrity, authentication, and encryption.

social engineering Attacks that leverage human trust and social behaviors to divulge sensitive information.

Software as a Service (SaaS) A cloud service in which the service consists of access to working software, without the need to be concerned about the details of installing and maintaining the software or the servers on which it runs.

Software-Defined Access Cisco's intent-based networking (IBN) offering for enterprise networks.

software-defined architecture In computer networking, any architecture that provides mechanisms for automated software control of the network components, typically using a controller. Any architecture that leads to a Software-Defined Network (SDN).

Software-Defined Networking (SDN) A branch of networking that emerged in the marketplace in the 2010s characterized by the use of a centralized software controller that takes over varying amounts of the control plane processing formerly done inside networking devices, with the controller directing the networking elements as to what forwarding table entries to put into their forwarding tables.

SOHO A classification of a business site with a relatively small number of devices, sometimes in an employee office in their home.

Source NAT The type of Network Address Translation (NAT) used most commonly in networks (as compared to destination NAT), in which the source IP address of packets entering an inside interface is translated.

southbound API In the area of SDN, a reference to the APIs used between a controller and the network elements for the purpose of learning information from the elements and for programming (controlling) the forwarding behavior of the elements. Also called a southbound interface.

southbound interface Another term for southbound API. *See also* southbound API.

spear phishing Phishing that targets a group of users who share a common interest or connection.

spine In an ACI network design for a single site, a switch that connects to leaf switches only, for the purpose of receiving frames from one leaf switch and then forwarding the frame to some other leaf switch.

spine-leaf network A single-site network topology in which endpoints connect to leaf switches, leaf switches connect to all spine switches (but not to other leaf switches), and spine switches connect to all leaf switches (but not to other spine switches). The resulting topology results in predictable switching paths with three switches between any two endpoints that connect to different leaf switches.

spoofing attack A type of attack in which parameters such as IP and MAC addresses are spoofed with fake values to disguise the sender.

spurious DHCP server A DHCP server that is used by an attacker for attacks that take advantage of DHCP protocol messages.

SSL *See* Secure Sockets Layer.

standard access list A list of IOS global configuration commands that can match only a packet's source IP address for the purpose of deciding which packets to discard and which to allow through the router.

star topology A network topology in which endpoints on a network are connected to a common central device by point-to-point links.

stateful A protocol or process that requires information stored from previous transactions to perform the current transaction.

stateless A protocol or process that does not use information stored from previous transactions to perform the current transaction.

subinterface One of the virtual interfaces on a single physical interface.

switch abstraction The fundamental idea of what a switch does, in generalized form, so that standards protocols and APIs can be defined that then program a standard switch abstraction; a key part of the OpenFlow standard.

syslog A server that takes system messages from network devices and stores them in a database. The syslog server also provides reporting capabilities on these system messages. Some syslog servers can even respond to select system messages with certain actions such as emailing and paging.

syslog server A server application that collects syslog messages from many devices over the network and provides a user interface so that IT administrators can view the log messages to troubleshoot problems.

T

T1 A line from the telco that allows transmission of data at 1.544 Mbps, with the capability to treat the line as 24 different 64-Kbps DSO channels (plus 8 Kbps of overhead).

T3 A line from the telco that allows transmission of data at 44.736 Mbps, with the capability to treat the line as 28 different 1.544-Mbps DS1 (Tl) channels, plus overhead.

TACACS+ A security protocol often used for user authentication as well as authorization and accounting, often used to authenticate users who log in to Cisco routers and switches.

tail drop Packet drops that occur when a queue fills, another message arrives that needs to be placed into the queue, and the networking device tries to add the new message to the tail of the queue but finds no room in the queue, resulting in a dropped packet.

target hardware address In both an ARP request and reply message, the field intended to be used to list the destination (target) device's hardware address, typically an Ethernet LAN address. This field is left as all binary 0s for typical ARP request messages.

target IP address In both an ARP request and reply message, the field intended to be used to list the destination (target) device's IP address.

TCAM *See* ternary content-addressable memory.

TCP Transmission Control Protocol. A connection-oriented transport layer TCP/IP protocol that provides reliable data transmission.

TCP window The mechanism in a TCP connection used by each host to manage how much data the receiver allows the sender to send to the receiver.

TCP/IP Transmission Control Protocol/Internet Protocol. A common name for the suite of protocols developed by the U.S. Department of Defense in the 1970s to support the construction of worldwide internetworks. TCP and IP are the two best-known protocols in the suite.

telco A common abbreviation for telephone company.

ternary content-addressable memory (TCAM) A type of physical memory, either in a separate integrated circuit or built into an ASIC, that can store tables and then be searched against a key, such that the search time happens quickly and does not increase as the size of the table increases. TCAMs are used extensively in higher-performance networking devices as the means to store and search forwarding tables in Ethernet switches and higher-performance routers.

TFTP Trivial File Transfer Protocol. An application protocol that allows files to be transferred from one computer to another over a network, but with only a few features, making the software require little storage space.

TFTP client An application that can connect to a TFTP server for the purpose of transferring copies of files to and from the server.

TFTP server An application that runs and waits for TFTP clients to connect to it over UDP port 69 to support the client's commands to transfer copies of files to and from the server.

threat An actual potential to use an exploit to take advantage of a vulnerability.

three-tier design *See* core design.

time interval (shaper) Part of the internal logic used by a traffic shaping function, which defines a short time period in which the shaper sends packets until a number of bytes are sent, and then the shaper stops sending for the rest of the time interval, with a goal of averaging a defined bit rate of sending data.

TLD DNS server A DNS server with the role of identifying the IP address of the authoritative DNS server for a domain that resides within its top-level domain.

Top of Rack (ToR) switch In a traditional data center design with servers in multiple racks and the racks in multiple rows, a switch placed in the top of the rack for the purpose of providing physical connectivity to the servers (hosts) in that rack.

top-level domain (TLD) With DNS name services, the top-level domain is the most significant (rightmost) of the period-separated values in a DNS host name—for example, the .com within host name www.example.com.

Transport Layer Security (TLS) A security standard that replaced the older Secure Sockets Layer (SSL) protocol, providing functions such as authentication, confidentiality, and message integrity over reliable in-order data streams like TCP.

trojan horse Malware that is hidden and packaged inside other legitimate software.

trust boundary When thinking about a message as it flows from the source device to the destination device, the trust boundary is the first device the message reaches for which the QoS markings in the message's various headers can be trusted as having an accurate value, allowing the device to apply the correct QoS actions to the message based on the marking.

trusted port With both the DHCP snooping and Dynamic ARP Inspection (DAI) switch features, the concept and configuration setting that tells the switch to allow all incoming messages of that respective type, rather than to consider the incoming messages (DHCP and ARP, respectively) for filtering.

tunnel interface A virtual interface in a Cisco router used to configure a variety of features, including Generic Routing Encapsulation (GRE), which encapsulates IP packets into other IP packets for the purpose of creating VPNs.

two-tier design *See* collapsed core design.

Type of Service (ToS) In the original definition of the IP header, a byte reserved for the purpose of QoS functions, including holding the IP Precedence field. The ToS byte was later repurposed to hold the DSCP field.

U

UDP User Datagram Protocol. Connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery.

uncacheable For resources that might be repeatedly requested over time, an attribute that means that the requesting host should not use its local copy of the resource, but instead ask for a new copy every time the resource is required.

underlay In SDA, the network devices and links that create basic IP connectivity to support the creation of VXLAN tunnels for the overlay.

Unified Computing System (UCS) The Cisco brand name for its server hardware products.

Universal Power over Ethernet (UPoE) A specific PoE standard as defined in the IEEE 802.3bt amendment to the 802.3 standard, which uses four wire pairs to supply power with a maximum of 60 watts as supplied by the PSE.

Universal Power over Ethernet Plus (UPoE+) A specific PoE standard as defined in the IEEE 802.3bt amendment to the 802.3 standard, which uses four wire pairs to supply power with a maximum of 100 watts as supplied by the PSE.

untrusted port With both the DHCP snooping and Dynamic ARP Inspection (DAI) switch features, the concept and configuration setting that tells the switch to analyze each incoming message of that respective type (DHCP and ARP) and apply some rules to decide whether to discard the message.

UPoE Universal Power over Ethernet. A specific PoE standard as defined in IEEE 802.3bt amendment to the 802.3 standard, which uses four wire pairs to supply power with a maximum of 60 watts as supplied by the PSE.

URI Uniform Resource Identifier. The formal and correct term for the formatted text used to refer to objects in an IP network. This text is commonly called a URL or a web address. For example, http://www.certskills.com/blog is a URI that identifies the protocol (HTTP), host name (www.certskills.com), and web page (blog).

URI parameters *See* URI query (parameters).

URI path (resource) In a URI, the part that follows the first /, up to the query field (which begins with a ?), which identifies the resource in the context of a server.

URI query (parameters) In a URI, the part that follows the first ?, which provides a place to list variable names and values as parameters.

URI resource *See* URI path (resource).

URL Uniform Resource Locator. The widely popular terms for the formatted text used to refer to objects in an IP network. For example, http://www.certskills.com/blog is a URL that identifies the protocol (HTTP), host name (www.certskills.com), and web page (blog).

user network interface (UNI) A term used in a variety of WAN standards, including carrier/Metro Ethernet, that defines the standards for how a customer device communicates with a service provider's device over an access link.

username secret A reference to the password configured on the **username** *name* **secret** *pass-value* command, which defines a username and an encoded password, used to build a local username/password list on the router or switch.

V

variable In applications, a method to assign a name to a value so that the application can refer to the value, change it, compare it to other values, apply logic, and perform other actions typical of software applications.

version control software Applications that monitor files for changes, tracking each specific change, the user, the date/time, with tools so that users can compare versions of each file through its history to see the differences.

violation mode In port security, a configuration setting that defines the specific set of actions to take on a port when a port security violation occurs. The modes are shutdown, restrict, and protect.

virtual CPU (vCPU) In a virtualized server environment, a CPU (processor) core or thread allocated to a virtual machine (VM) by the hypervisor.

virtual IP address For any FHRP protocol, an IP address that the FHRP shares between multiple routers so that they appear as a single default router to hosts on that subnet.

virtual MAC address (vMAC) For any FHRP protocol, a MAC address that the FHRP uses to receive frames from hosts.

virtual machine An instance of an operating system, running on server hardware that uses a hypervisor to allocate a subset of the server hardware (CPU, RAM, disk, and network) to that VM.

virtual network function (VNF) Any function done within a network (for example, router, switch, firewall) that is implemented not as a physical device but as an OS running in a virtualized system (for instance, a VM).

virtual network identifier (VNID) In SDA and VXLAN, the identifier for a separate routing and switching instance. All devices in the same VNID are considered to be allowed to send data to each other unless prevented from doing so by other security mechanisms.

virtual NIC (vNIC) In a virtualized server environment, a network interface card (NIC) used by a virtual machine, which then connects to some virtual switch (vSwitch) running on that same host, which in turn connects to a physical NIC on the host.

virtual private network (VPN) A set of security protocols that, when implemented by two devices on either side of an unsecure network such as the Internet, can allow the devices to send data securely. VPNs provide privacy, device authentication, anti-replay services, and data integrity services.

Virtual Router Redundancy Protocol (VRRP) A TCP/IP RFC protocol that allows two (or more) routers to share the duties of being the default router on a subnet, with an active/standby model, with one router acting as the default router and the other sitting by waiting to take over that role if the first router fails.

virtual switch (vSwitch) A software-only virtual switch inside one host (one hardware server), to provide switching features to the virtual machines running on that host.

virus Malware that injects itself into other applications and then propagates through user intervention.

VPN *See* virtual private network.

VPN client Software that resides on a PC, often a laptop, so that the host can implement the protocols required to be an endpoint of a VPN.

vulnerability A weakness that can be used to compromise security.

VXLAN Virtual Extensible LAN. A flexible encapsulation protocol used for creating tunnels (overlays).

W

WAN edge The device (typically a router) at enterprise sites that connects to private WAN links, therefore sitting at the edge of the WAN.

WAN link Another term for leased line.

WAN service provider A company that provides private WAN services to customers; the company may have a heritage as a telco or cable company.

watering hole attack An attack where a site frequently visited by a group of users is compromised; when the target users visit the site, they will be infected with malware, but other users will not.

web server Software, running on a computer, that stores web pages and sends those web pages to web clients (web browsers) that request the web pages.

well-known port A TCP or UDP port number reserved for use by a particular application. The use of well-known ports allows a client to send a TCP or UDP segment to a server, to the correct destination port for that application.

whaling A phishing technique that targets high-profile individuals to follow links to malicious sites.

wildcard mask The mask used in Cisco IOS ACL commands and OSPF and EIGRP **network** commands.

window Represents the number of bytes that can be sent without receiving an acknowledgment.

worm Malware that propagates from one system to another, infecting as it goes, all autonomously.

write community *See* read-write community.

X–Y–Z

XML (eXtensible Markup Language) A markup language that helps enable dynamic web pages; also useful as a data serialization language.

YAML (YAML Ain't Markup Language) A data serialization language that can be easily read by humans; used by Ansible.

zero-day vulnerability Jargon that refers to a security vulnerability during the time between the day in which the vulnerability was discovered, until the vendor or open-source group responsible for that software can develop a fix and make it public.

Index

Numbers

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y - Z

This page intentionally left blank

REGISTER YOUR PRODUCT at CiscoPress.com/register

Access Additional Benefits and SAVE 35% on Your Next Purchase

- Download available product updates.
- Access bonus material when applicable.
- Receive exclusive offers on new editions and related products. (Just check the box to hear from us when setting up your account.)
- Get a coupon for 35% for your next purchase, valid for 30 days. Your code will be available in your Cisco Press cart. (You will also find it in the Manage Codes section of your account page.)

Registration benefits vary by product. Benefits will be listed on your account page under Registered Products.

CiscoPress.com – Learning Solutions for Self-Paced Study, Enterprise, and the Classroom

Cisco Press is the Cisco Systems authorized book publisher of Cisco networking technology, Cisco certification self-study, and Cisco Networking Academy Program materials.

At CiscoPress.com you can

- Shop our books, eBooks, software, and video training.
- Take advantage of our special offers and promotions (ciscopress.com/promotions).
- Sign up for special offers and content newsletters (ciscopress.com/newsletters).
- Read free articles, exam profiles, and blogs by information technology experts.
- Access thousands of free chapters and video lessons.

Connect with Cisco Press – Visit CiscoPress.com/community

Learn about Cisco Press community events and programs.

Cisco Press

APPENDIX D

Topics from Previous Editions

Cisco changes the exams, renaming the exams on occasion, and changing the exam numbers every time it changes the exam with a new blueprint, even with a few name changes over the years. As a result, the current CCNA 200-301 exam serves as the eighth separate version of CCNA in its 20-plus year history. At every change to the exams, we create new editions of the books to match the new exam.

We base the books' contents on Cisco's exam topics; that is, the book attempts to cover the topics Cisco lists as exam topics. However, the book authoring process does create some challenges, particularly with the balance of what to include in the books and what to leave out.

For instance, when comparing a new exam to the old, I found Cisco had removed some topics—and I might want to keep the content in the book. There are a few reasons why. Sometimes I just expect that some readers will still want to read about that technology. Also, more than a few schools use these books as textbooks, and keeping some of the older-but-still-relevant topics can be a help. And keeping the old material available on each book's companion website takes only a little extra work, so we do just that.

Some of the older topics that I choose to keep on the companion website are small, so I collect them into this appendix. Other topics happen to have been an entire chapter in a previous edition of the books, so we include those topics each as a separate appendix. Regardless, the material exists here in this appendix, and in the appendixes that follow, for your use if you have a need. But do not feel like you have to read this appendix for the current exam.

The topics in this appendix are as follows:

- Cisco Device Hardening
- Implementing DHCP
- Troubleshooting with IPv4 ACLs
- Implementing HSRP
- Global Load Balancing Protocol (GLBP)
- Implementing Simple Network Management Protocol
- Analyzing LAN Physical Standard Choices
- Metro Ethernet Virtual Circuits
- MPLS VPNs and OSPF

NOTE The content under the heading "Cisco Device Hardening" was most recently published for the 100-105 Exam in 2016, in Chapter 34 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Cisco Device Hardening

The term *device hardening* refers to making it more difficult for attackers to gain access to the device or to cause problems for the device. This section does not attempt to mention all such details, but it does touch on a few items. (Note that the CCNA Security certification gets into much more detail about router and switch device security.)

In particular, this second major section of the chapter begins by showing how to set some login banner message text for users. The next two topics look at how to secure items unused in the device—unused switch ports on switches and unused software services in both routers and switches.

Configuring Login Banners

Cisco switches and routers can display a variety of banners to a new user when logging in to the switch or router. A banner is simply some text that appears on the screen for the user. You can configure a router or switch to display multiple banners, some before login and some after.

IOS supports three banners based on the first keyword in the **banner** command. Table D-1 lists the three most popular banners and their typical use.

Table D-1 Banners and Their Typical Use

Banner	Typical Use
Message of the Day (MOTD)	Used for temporary messages that can change from time to time, such as "Router1 down for maintenance at midnight."
Login	Because it is always shown before the user logs in, this message is often used to show warning messages, like "Unauthorized Access Prohibited."
Exec	Because this banner always appears after login, it typically lists device information that outsiders should not see but that internal staff might want to know, for example, the exact location of the device.

In what may seem like trivia, the banners actually appear in different places based on a couple of conditions. Figure D-1 summarizes when the user sees each of these banners, reading from top to bottom. Console and Telnet users see the banners in the order shown on the left, and SSH users see the banners in the order on the right.

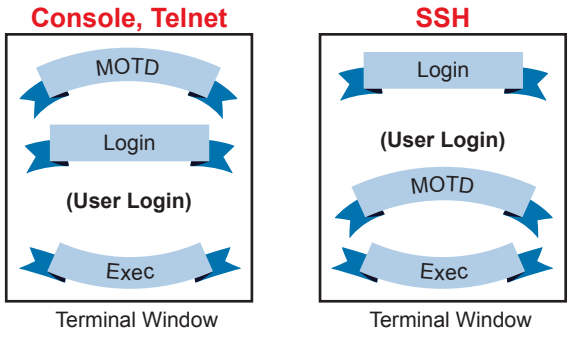


Figure D-1 *Banner Sequence Compared: Console/Telnet Versus SSH (Blue Ribbon Set © petrnutil/123RF)*

NOTE If using SSH, and the switch or router uses only SSHv1, the login banner is not shown to the SSH user.

The **banner** global configuration command can be used to configure all three types of these banners. In each case, the type of banner is listed as the first parameter, with **motd** being the default option. The first nonblank character after the banner type is called a beginning delimiter character. When a delimiter character is used, the banner text can span several lines, with the CLI user pressing Enter at the end of each line. The CLI knows that the banner has been configured as soon as the user enters the same delimiter character again.

Example D-1 shows the configuration process for all three types of banners from Table D-1, followed by a sample user login session from the console that shows the banners in use. The first configured banner in the example, the MOTD banner, omits the banner type in the **banner** command as a reminder that **motd** is the default banner type. The first two **banner** commands use a **#** as the delimiter character. The third **banner** command uses a Z as the delimiter, just to show that any character can be used. Also, the last **banner** command shows multiple lines of banner text.

Example D-1 *Banner Configuration*

```
! Below, the three banners are created in configuration mode. Note that any
! delimiter can be used, as long as the character is not part of the message
! text.

SW1(config)# banner #
Enter TEXT message.  End with the character '#'.
(MOTD) Switch down for maintenance at 11PM Today #
SW1(config)# banner login #
Enter TEXT message.  End with the character '#'.
(Login) Unauthorized Access Prohibited!!!!
#
SW1(config)# banner exec Z
Enter TEXT message.  End with the character 'Z'.
(Exec) Company picnic at the park on Saturday.
```

```
 Don't tell outsiders!
Z
SW1(config)# end

! Below, the user of this router quits the console connection, and logs
! back in, seeing the motd and login banners, then the password prompt,
! and then the exec banner.

SW1# quit

SW1 con0 is now available

Press RETURN to get started.

(MOTD) Switch down for maintenance at 11PM Today
(Login) Unauthorized Access Prohibited!!!!

User Access Verification

Username: fred
Password:
(Exec) Company picnic at the park on Saturday.
 Don't tell outsiders!
SW1>
```

Securing Unused Switch Interfaces

The default settings on Cisco switches work great if you want to buy a switch, unbox it, plug it in, and have it immediately work without any other effort. Those same defaults have an unfortunate side effect for security, however. With all default configuration, an attacker might use unused interfaces to gain access to the LAN. So, Cisco makes some general recommendations to override the default interface settings to make the unused ports more secure, as follows:

- Administratively disable the interface using the **shutdown** interface subcommand.
- Prevent VLAN trunking by making the port a nontrunking interface using the **switchport mode access** interface subcommand.
- Assign the port to an unused VLAN using the **switchport access vlan** *number* interface subcommand.
- Set the native VLAN so that it is not VLAN 1 but instead is an unused VLAN, using the **switchport trunk native vlan** *vlan-id* interface subcommand.

Frankly, if you just shut down the interface, the security exposure goes away, but the other tasks prevent any immediate problems if someone else comes around and enables the interface by configuring a **no shutdown** command.

NOTE The contents under the headings "DHCP Server Configuration on Routers," "IOS DHCP Server Verification," and "Troubleshooting DHCP Services" were most recently published for the 100-105 Exam in 2016, in Chapter 20 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Implementing DHCP

This section includes DHCP implementation topics from an earlier edition of the book.

DHCP Server Configuration on Routers

A quick Google search on "DHCP server products" reveals that many companies offer DHCP server software. Cisco routers (and some Cisco switches) can also act as a DHCP server with just a little added configuration.

Configuring a Cisco router to act as a DHCP server uses a new configuration concept, one per subnet, called a *DHCP pool*. All the per-subnet settings go into a per-subnet DHCP pool. The only DHCP command that sits outside the pool is the command that defines the list of addresses excluded from being leased by DHCP. The Cisco IOS DHCP server configuration steps are as follows:

Step 1. Use the **ip dhcp excluded-address** *first last* command in global configuration mode to list addresses that should be excluded (that is, not leased by DHCP).

Step 2. Use the **ip dhcp pool** *name* command in global configuration mode to both create a DHCP pool for a subnet and to navigate into DHCP pool configuration mode. Then also:

- **A.** Use the **network** *subnet-ID mask* or **network** *subnet-ID prefix-length* command in DHCP pool configuration mode to define the subnet for this pool.
- **B.** Use the **default-router** *address1 address2…* command in DHCP pool configuration mode to define default router IP address(es) in that subnet.
- **C.** Use the **dns-server** *address1 address2…* command in DHCP pool configuration mode to define the list of DNS server IP addresses used by hosts in this subnet.
- **D.** Use the **lease** *days hours minutes* command in DHCP pool configuration mode to define the length of the lease, in days, hours, and minutes
- **E.** Use the **domain-name** *name* command in DHCP pool configuration mode to define the DNS domain name.
- **F.** Use the **next-server** *ip-address* command in DHCP pool configuration mode to define the TFTP server IP address used by any hosts (like phones) that need a TFTP server.

Of course, an example can help, particularly with so many configuration commands required. Figure D-2 shows the organization of the configuration, while sticking to pseudocode rather than the specific configuration commands. (Upcoming Example D-2 shows a matching configuration.) Note that for each of the two LAN subnets, there is a global command to exclude addresses, and then a group of settings for each of two different DHCP pools.

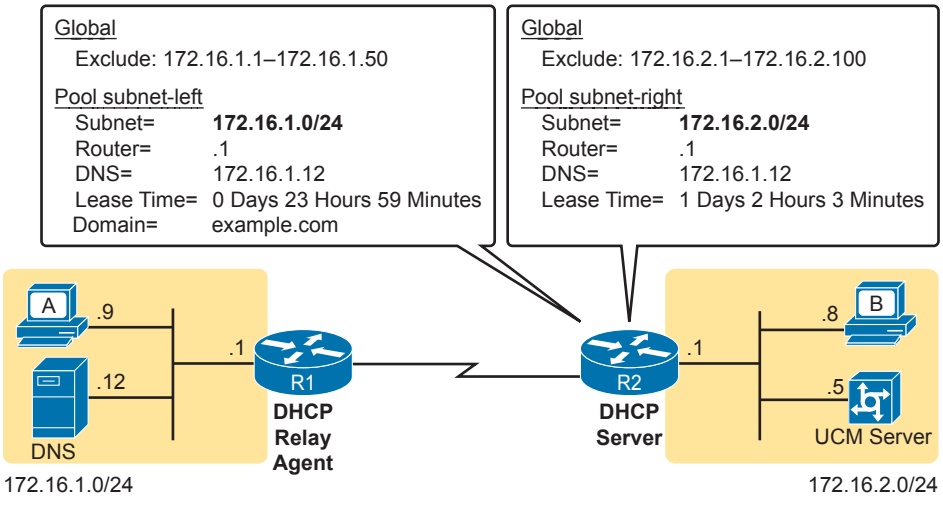


Figure D-2 *DHCP Server Configuration Pseudocode*

Example D-2 *R2 as a DHCP Server Per the Concepts in Figure D-2*

```
ip dhcp excluded-address 172.16.1.1 172.16.1.50
ip dhcp excluded-address 172.16.2.1 172.16.2.100
!
ip dhcp pool subnet-left
 network 172.16.1.0 255.255.255.0
 dns-server 172.16.1.12
 default-router 172.16.1.1
 lease 0 23 59
 domain-name example.com
 next-server 172.16.2.5
!
ip dhcp pool subnet-right
 network 172.16.2.0 /24
 dns-server 172.16.1.12
 default-router 172.16.2.1
 lease 1 2 3
 next-server 172.16.2.5
```

Focus on subnet 172.16.1.0/24 for a moment: the subnet configured as pool subnet-left. The subnet ID and mask match the subnet ID chosen for that subnet. Then, the global **ip dhcp excluded-address** command, just above, reserves 172.16.1.1 through 172.16.1.50, so that this DHCP server will not lease these addresses. The server will automatically exclude the subnet ID (172.16.1.0) as well, so this DHCP server will begin leasing IP addresses starting with the .51 address.

Now look at the details for subnet-right. It uses a DHCP pool **network** command with a prefix style mask. It defines the same DNS server, as does the pool for the other subnet, but a different default router setting, because, of course, the default router in each subnet

is different. This pool includes a lease time of 1:02:03 (1 day, 2 hours, and 3 minutes) just as an example.

Also note that both subnets list a TFTP server IP address of the Unified Communications Manager (UCM) server with the **next-server** command. In most cases, you would find this setting in the pools for subnets in which phones reside.

Finally, note that configuring a router as a DHCP server does not remove the need for the **ip helper-address** command. If DHCP clients still exist on LANs that do not have a DHCP server, then the routers connected to those LANs still need the **ip helper-address** command. For example, in Figure D-2, R1 would still need the **ip helper-address** command on its LAN interface. R2 would not need the command on its LAN interface, because R2 could service those requests, rather than needing to forward the DHCP messages to some other server.

IOS DHCP Server Verification

The IOS DHCP server function has several different **show** commands. These three commands list most of the details:

show ip dhcp binding: Lists state information about each IP address currently leased to a client

show ip dhcp pool [*poolname*]**:** Lists the configured range of IP addresses, plus statistics for the number of currently leased addresses and the high-water mark for leases from each pool

show ip dhcp server statistics: Lists DHCP server statistics

Example D-3 shows sample output from two of these commands, based on the configuration from Figure D-2 and Example D-2. In this case, the DHCP server leased one IP address from each of the pools, one for host A, and one for host B, as shown in the highlighted portions of the output.

Example D-3 *Verifying Current Operation of a Router-Based DHCP Server*

```
R2# show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
172.16.1.51         0063.6973.636f.2d30.    Oct 12 2012 02:56 AM    Automatic
                    3230.302e.3131.3131.
                    2e31.3131.312d.4661.
                    302f.30
172.16.2.101        0063.6973.636f.2d30.    Oct 12 2012 04:59 AM    Automatic
                    3230.302e.3232.3232.
                    2e32.3232.322d.4769.
                    302f.30
R2# show ip dhcp pool subnet-right
Pool subnet-right :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
```

```
Leased addresses                : 1
Pending event                   : none
1 subnet is currently in the pool :
Current index        IP address range                    Leased addresses
172.16.2.102         172.16.2.1       - 172.16.2.254     1
```

Note that the output in Example D-3 does not happen to list the excluded addresses, but it does show the effects. The addresses assigned to the clients end with .51 (host A, subnet 172.16.1.0) and .101 (host B, subnet 172.16.2.0), proving that the server did exclude the addresses as shown in the configuration in Example D-2. The server avoided the .1 through .50 addresses in subnet 172.16.1.0, and the .1 through .100 addresses in subnet 172.16.2.0.

NOTE The DHCP server keeps status (state) information about each DHCP client that leases an address. Specifically, it remembers the DHCP client ID, and the IP address leased to the client. As a result, an IPv4 DHCP server can be considered to be a stateful DHCP server.

Troubleshooting DHCP Services

To be prepared for the CCNA simlet questions, you have to be ready to predict what symptoms would occur when the network was misconfigured in particular ways. This next section takes a similar approach, pointing out the most typical issues that could be introduced through incorrect or missing configuration, and then discussing what symptoms should happen and how to recognize those problems.

This section begins with a typical look at configuration mistakes and the symptoms that occur with those mistakes. In particular, this section looks at problems with the relay agent's helper address as well as the IOS DHCP server configuration. This section then looks at non-DHCP problems related to that data plane, breaking the problem into issues between the client and relay agent, and between the relay agent and DHCP server. The final section takes a short look at how a DHCP server prevents duplicate IP addresses between hosts that use static IP addresses and those that use DHCP.

DHCP Relay Agent Configuration Mistakes and Symptoms

One configuration mistake that prevents DHCP client from leasing an IP address is the misconfiguration or the omission of the **ip helper-address** interface subcommand on the router acting as the DHCP relay agent. The relay agent takes the incoming DHCP message, changes the destination address of the packet to be the address on the **ip helper-address** *address* command, and forwards the packet to that address. If the command is missing, the router does not attempt to forward the DHCP messages at all; if it is incorrect, the relay agent forwards the DHCP packets, but they never arrive at the actual DHCP server.

The main problem symptom in this case is the failure of a DHCP client to lease an address. If you can identify a client that has a problem, and you know what VLAN or subnet in which that host resides, you can then work to identify any routers connected to that subnet, to find and correct the **ip helper-address** subcommands.

Beyond that step, this list summarizes a few other related points.

- The DHCP relay agent feature is needed on interfaces only if the DHCP server is on a different subnet; it is not needed if the DHCP server is on the same subnet as the client.

- On routers with VLAN trunks (with a router-on-a-stick [ROAS] subinterface configuration), the subinterfaces also need an **ip helper-address** command (assuming they meet the first criteria in this list).
- If an exam question does not allow you to look at the configuration, use the **show ip interface** [*type number*] command to view the **ip helper-address** setting on an interface.

About that last point, Example D-4 shows an example of the **show ip interface g0/0** command. In this case, the interface has been configured with the **ip helper-address 172.16.2.11** command; the **show** command output basically restates that fact. Note that if there were no **ip helper-address** configured on the interface, the text would instead read "Helper address is not set."

Example D-4 *Listing the Current Helper Address Setting with* **show ip interface**

```
R1# show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet address is 182.16.1.1/24
  Broadcast address is 255.255.255.255
  Address determined by non-volatile memory
  MTU is 1500 bytes
  Helper address is 172.16.2.11
! Lines omitted for brevity (about 20 lines)
```

IOS DHCP Server Configuration Mistakes and Symptoms

When using an IOS DHCP server, from a troubleshooting perspective, break issues into two broad categories: those that prevent DHCP clients from leasing an address, and those that allow the lease but provide incorrect settings to the client.

First, the primary configuration mistake that causes a failure in the DHCP lease process is the misconfiguration of the **network** command. The problem revolves around these key facts:

- The packet from the relay agent to the DHCP server uses the relay agent's interface IP address as the source IP address in the forwarded DHCP message.
- The DHCP server compares that source IP address in the received DHCP packet to the **network** commands in its DHCP pools to find the right pool.
- Each **network** *subnet mask* command implies a range of addresses, just like any other IP network or subnet shown with a subnet mask.
- If the source IP address of the packet is not in the range of addresses implied by any **network** command in all the pools, the DHCP server has no pool to use for that request. The DHCP server does not know how to respond, so it does not reply at all.

As an example of that failure, consider the configuration shown in Figure D-3. The left side shows the configuration on R1, a DHCP relay agent that has two interfaces configured with the **ip helper-address 172.16.2.11** command. The DHCP server configuration on the right lists two pools, intended as one pool for each subnet off Router R1. However, the **network 172.16.3.0 /25** command implies an address range of 172.16.3.0 to 172.16.3.127, and the relay agent's interface address of 172.16.3.254 is not within that range of numbers. The solution would be to correct the DHCP server's **network** command to use a /24 mask.

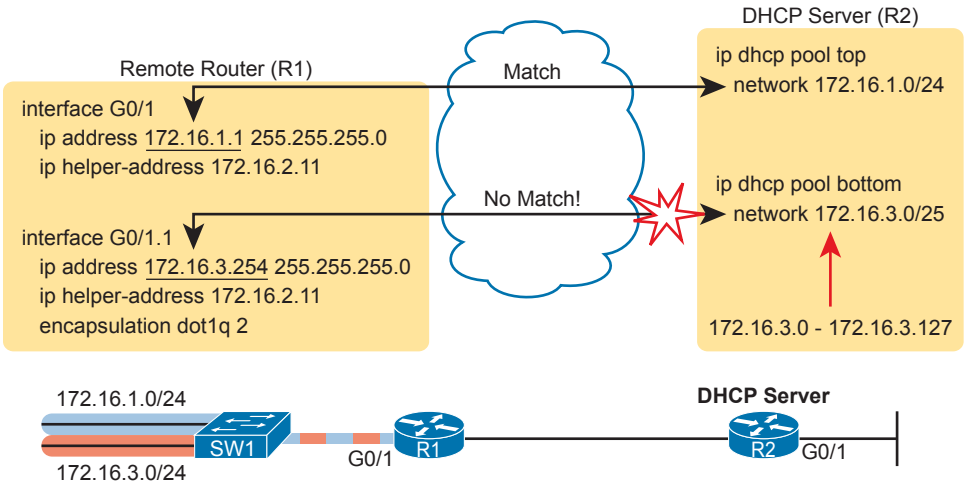


Figure D-3 *An Example Misconfiguration of a DHCP Pool* **network** *Command*

NOTE The **ip helper-address** configuration on the left is correct. The figure uses a ROAS configuration here just to reinforce the comment in the earlier section that ROAS subinterfaces also need an **ip helper-address** subcommand.

While you ultimately need to find this kind of problem and fix the configuration, on the exam you need to be ready to discover the root cause based on symptoms and **show** commands as well. So, when troubleshooting DHCP issues, and the client fails to lease an address, look at the IOS DHCP server's **network** commands. Calculate the range of IP addresses as if that command were defining a subnet. Then compare that range of addresses by the **network** command in each pool to the interface addresses on the DHCP relay agent routers. Every relay agent interface (that is, every interface with an **ip helper-address** command configured) should be included in a pool defined at the IOS DHCP server.

The DHCP server can also be misconfigured in a way that allows the lease of an address, but then causes other problems. If the lease process works, but the rest of the parameters given to the client are incorrect or missing, the client could operate, but operate poorly. This list summarizes the kinds of mistakes and the resulting symptoms:

- With the DNS server IP addresses incorrectly configured on the server (or omitted), hosts would fail to resolve hostnames into their associated IP addresses.
- With the default gateway IP address incorrectly configured on the server (or omitted), hosts could not communicate outside the local subnet.
- With the TFTP server IP address incorrectly configured (or omitted), an IP phone would fail to correctly load its configuration.

IP Connectivity from DHCP Relay Agent to DHCP Server

For the DHCP process to work with a centralized server, IP broadcast packets must flow between the client and relay agent, and IP unicast packets must flow between the relay agent and the DHCP server. Any problem that prevents the flow of these packets also prevents DHCP from working.

For perspective, consider the topology in Figure D-4, which again shows the relay agent on the left and the DHCP server on the right. The server uses IP address 172.16.2.11, and the relay agent uses interface address 172.16.1.1. Any failure that prevents the flow of IP packets between those two IP addresses would prevent host A from leasing an IP address.

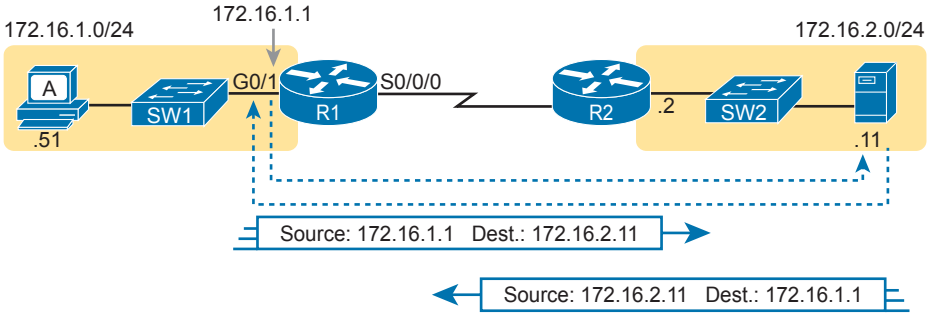


Figure D-4 *Addresses Used Between Relay Agent and Server*

Remember that the IP addresses used on the packets between the relay agent and server, and know that you may need to troubleshoot IP routing to ensure those packets can be delivered.

LAN Connectivity Between the DHCP Client and Relay Agent

You might encounter a network environment where DHCP messages on the same LAN as the DHCP client all show a destination IP address of 255.255.255.255. What does that really mean? When a packet uses this 255.255.255.255 address:

- The address is called the *local broadcast address*.
- Packets sent to this address are not forwarded as-is by routers.
- On a LAN, the sender of an IP local broadcast packet encapsulates these IP packets in an Ethernet frame with an Ethernet broadcast destination address (FFFF.FFFF.FFFF), so the LAN broadcasts the frame.

As a result of the logic in these steps, the broadcast DHCP messages can easily flow between the client and router, as long as the LAN works.

Summary of DHCP Troubleshooting

In summary, as a study tool, the following list summarizes the key troubleshooting ideas from this section on troubleshooting DHCP:

Step 1. If using a centralized DHCP server, at least one router on each remote subnet that has DHCP clients must act as DHCP relay agent, and have a correctly configured **ip helper-address** *address* subcommand on the interface connected to that subnet.

Step 2. If using a centralized IOS DHCP server, make sure the DHCP pools' **network** commands match the entire network's list of router interfaces that have an **ip helper-address** command pointing to this DHCP server.

Step 3. Troubleshoot for any IP connectivity issues between the DHCP relay agent and the DHCP server, using the relay agent interface IP address and the server IP address as the source and destination of the packets.

Step 4. Troubleshoot for any LAN issues between the DHCP client and the DHCP relay agent.

Also, as one final note about DHCP in the real world, DHCP might seem dangerous at this point, with all the focus on potential problems in this section, combined with the importance of DHCP and its use by most end user devices. However, DHCP has some great availability features. First, most DHCP servers set their lease times for at least a few days, often a week, or maybe longer. Combined with that, the DHCP protocol has several processes through which the client reconfirms the existing lease with the server, and re-leases the same IP address in advance of the expiration of the lease. Clients do not simply wait until the moment the lease would expire to then contact the DHCP server, hoping it is available. So the network can have outages, and DHCP clients that have already leased an address can continue to work without any problem.

Detecting Conflicts with Offered Versus Used Addresses

Beyond troubleshooting the types of problems that would prevent DHCP from working, the IOS DHCP server tries to prevent another type of problem: assigning IP addresses with DHCP when another host tries to statically configure that same IP address. Although the DHCP server configuration clearly lists the addresses in the pool, plus those to be excluded from the pool, hosts can still statically configure addresses from the range inside the DHCP pool. In other words, no protocols prevent a host from statically configuring and using an IP address from within the range of addresses used by the DHCP server.

Knowing that some host might have statically configured an address from within the range of addresses in the DHCP pool, both DHCP servers and clients try to detect such problems, called *conflicts*, before the client uses a newly leased address.

DHCP servers detect conflicts by using pings. Before offering a new IP address to a client, the DHCP server first pings the address. If the server receives a response to the ping, some other host must already be using the address, which lets the server know a conflict exists. The server notes that particular address as being in conflict, and the server does not offer the address, moving on to the next address in the pool.

The DHCP client can also detect conflicts, but instead of using ping, it uses ARP. In the client case, when the DHCP client receives from the DHCP server an offer to use a particular IP address, the client sends an Address Resolution Protocol (ARP) request for that address. If another host replies, the DHCP client has found a conflict.

Example D-5 lists output from the router-based DHCP server on R2, after host B detected a conflict using ARP. Behind the scenes, host B used DHCP to request a lease, with the process working normally until host B used ARP and found some other device already used 172.16.2.102. At that point, host B then sent a DHCP message back to the server, rejecting the use of address 172.16.2.102. The example shows the router's log message related to host B's discovery of the conflict, and a **show** command that lists all conflicted addresses.

Example D-5 *Displaying Information About DHCP Conflicts in IOS*

```
*Oct 16 19:28:59.220: %DHCPD-4-DECLINE_CONFLICT: DHCP address conflict:
  client 0063.6973.636f.2d30.3230.302e.3034.3034.2e30.3430.342d.4769.302f.30
    declined 172.16.2.102.
R2# show ip dhcp conflict
IP address        Detection method   Detection time          VRF
172.16.2.102      Gratuitous ARP     Oct 16 2012 07:28 PM
```

D

The **show ip dhcp conflict** command lists the method through which the server added each address to the conflict list: either gratuitous ARP, as detected by the client, or ping, as detected by the server. The server avoids offering these conflicted addresses to any future clients, until the engineer uses the **clear ip dhcp conflict** command to clear the list.

NOTE The content under the heading "Troubleshooting with IPv4 ACLs" was most recently published for the 200-105 Exam in 2016, in Chapter 17 of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

Troubleshooting with IPv4 ACLs

The use of IPv4 ACLs makes troubleshooting IPv4 routing more difficult. Any data plane troubleshooting process can include a catchall phrase to include checking for ACLs. A network can have all hosts working, DHCP settings correct, all LANs working, all router interfaces working, and all routers having learned all routes to all subnets—and ACLs can still filter packets. Although ACLs provide that important service of filtering some packets, ACLs can make the troubleshooting process that much more difficult.

This third of the three major sections of this chapter focuses on troubleshooting in the presence of IPv4 ACLs. It breaks the discussion into two parts. The first part gives advice about common problems you might see on the exam, and how to find those with **show** commands and some analysis. The second part then looks at how ACLs impact the **ping** command.

Analyzing ACL Behavior in a Network

ACLs cause some of the biggest challenges when troubleshooting problems in real networking jobs. The packets created by commands like **ping** and **traceroute** do not exactly match the fields in packets created by end users. The ACLs sometimes filter the **ping** and **traceroute** traffic, making the network engineer think some other kind of problems exists when no problems exist at all. Or, the problem with the end-user traffic really is caused by the ACL, but the ping and traceroute traffic works fine, because the ACL matches the end-user traffic with a **deny** action but matches the ping and traceroute traffic with a **permit** action.

As a result, much of ACL troubleshooting requires thinking about ACL configuration versus the packets that flow in a network, rather than using a couple of IOS commands that identify the root cause of the problem. The **show** commands that help are those that give you the configuration of the ACL, and on what interfaces the ACL is enabled. You can also see statistics about which ACL statements have been matched. And using pings and traceroutes can help—as long as you remember that ACLs may apply different actions to those packets versus the end-user traffic.

The following phrases the ACL troubleshooting steps into a list for easier study. The list also expands on the idea of analyzing each ACL in step 3. None of the ideas in the list are new compared to this chapter and the previous chapter, but it acts more as a summary of the common issues:

Step 1. Determine on which interfaces ACLs are enabled, and in which direction (**show running-config, show ip interfaces**).

Step 2. Find the configuration of each ACL (**show access-lists, show ip access-lists, show running-config**).

Step 3. Analyze the ACLs to predict which packets should match the ACL, focusing on the following points:

- **A.** **Misordered ACLs:** Look for misordered ACL statements. IOS uses first-match logic when searching an ACL.
- **B.** **Reversed source/destination addresses:** Analyze the router interface, the direction in which the ACL is enabled, compared to the location of the IP address ranges matched by the ACL statements. Make sure the source IP address field could match packets with that source IP address, rather than the destination, and vice versa for the destination IP address field.
- **C.** **Reversed source/destination ports:** For extended ACLs that reference UDP or TCP port numbers, continue to analyze the location and direction of the ACL versus the hosts, focusing on which host acts as the server using a well-known port. Ensure that the ACL statement matches the correct source or destination port depending on whether the server sent or will receive the packet.
- **D.** **Syntax:** Remember that extended ACL commands must use the **tcp** and **udp** keywords if the command needs to check the port numbers.
- **E.** **Syntax:** Note that ICMP packets do not use UDP or TCP; ICMP is considered to be another protocol matchable with the **icmp** keyword (instead of **tcp** or **udp**).
- **F.** **Explicit deny any:** Instead of using the implicit **deny any** at the end of each ACL, use an explicit configuration command to deny all traffic at the end of the ACL so that the **show** command counters increment when that action is taken.
- **G.** **Dangerous inbound ACLs:** Watch for inbound ACLs, especially those with deny all logic at the end of the ACL. These ACLs may discard incoming overhead protocols, like routing protocol messages.
- **H.** **Standard ACL location:** Standard ACLs enabled close to the source of matched addresses can discard the packets as intended, but also discard packets that should be allowed through. Always pay close attention to the requirements of the ACL in these cases.

The first two steps are important for simlet questions in case you are not allowed to look at the configuration; you can use other **show** commands to determine all the relevant ACL configuration. The next few pages show some of the related commands and how they can uncover some of the issues described in the just-completed ACL troubleshooting checklist.

ACL Troubleshooting Commands

If you suspect ACLs are causing a problem, the first problem-isolation step is to find the location and direction of the ACLs. The fastest way to do this is to look at the output of the **show running-config** command and to look for **ip access-group** commands under each interface. However, in some cases, enable mode access may not be allowed, and **show** commands are required. Instead, use the **show ip interfaces** command to find which ACLs are enabled on which interfaces, as shown in Example D-6.

Example D-6 *Sample* **show ip interface** *Command*

```
R1> show ip interface s0/0/1
Serial0/0/1 is up, line protocol is up
  Internet address is 10.1.2.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Multicast reserved groups joined: 224.0.0.9
  Outgoing access list is not set
  Inbound access list is 102
! roughly 26 more lines omitted for brevity
```

Note that the command output lists whether an ACL is enabled, in both directions, and which ACL it is. The example shows an abbreviated version of the **show ip interface S0/0/1** command, which lists messages for just this one interface. The **show ip interface** command would list the same messages for every interface in the router.

Step 2 of the ACL troubleshooting checklist then says that the contents of the ACL must be found. Again, the quickest way to look at the ACL is to use the **show running-config** command. If it's not available, the **show access-lists** and **show ip access-lists** commands list the same details shown in the configuration. These commands also list a useful counter that lists the number of packets that have matched each line in the ACL. Example D-7 shows an example.

Example D-7 **show ip access-lists** *Command Example*

```
R1# show ip access-lists
Extended IP access list 102
    10 permit ip 10.1.2.0 0.0.0.255 10.1.4.0 0.0.1.255 (15 matches)
```

The counter can be very useful for troubleshooting. If you can generate traffic that you think should match a particular line in an ACL, then you should see the matches increment on that counter. If you keep generating traffic that should match, but that line's counter never goes up, then those packets do not match that line in that ACL. Those packets could be matching an earlier line in the same ACL, or might not even be reaching that router (for any reason).

After the locations, directions, and configuration details of the various ACLs have been discovered in steps 1 and 2, the hard part begins—analyzing what the ACL really does. For example, one of the most common tasks you will do is to look at the address fields and decide the range of addresses matched by that field. Remember, for an ACL that sits in a router configuration, you can easily find the address range. The low end of the range is the address (the first number), and the high end of the range is the sum of the address and wildcard mask. For instance, with ACL 102 in Example D-7, which is obviously configured in some router, the ranges are as follows:

Source 10.1.2.0, wildcard 0.0.0.255: Matches from 10.1.2.0 through 10.1.2.255

Destination 10.1.4.0, wildcard 0.0.1.255: Matches from 10.1.4.0 through 10.1.5.255

The next few pages work through some analysis of a few of the items from step 3 in the troubleshooting checklist.

Example Issue: Reversed Source/Destination IP Addresses

IOS cannot recognize a case in which you attempt to match the wrong addresses in the source or destination address field. So, be ready to analyze the enabled ACLs and their direction versus the location of different subnets in the network. Then ask yourself about the packets that drive that ACL: what could the source and destination addresses of those packets be? And does the ACL match the correct address ranges, or not?

For example, consider Figure D-5, a figure that will be used in several troubleshooting examples in this chapter. The requirements for the next ACL follow the figure.

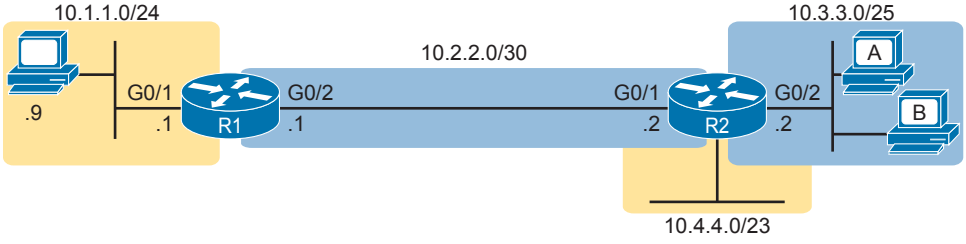


Figure D-5 *Example Network Used in IPv4 ACL Troubleshooting Examples*

For this next ACL, the requirements ask that you allow and prevent various flows, as follows:

- Allow hosts in subnet 10.3.3.0/25 and subnet 10.1.1.0/24 to communicate
- Prevent hosts in subnet 10.4.4.0/23 and subnet 10.1.1.0/24 from communicating
- Allow all other communications between hosts in network 10.0.0.0
- Prevent all other communications

Example D-8 shows the ACL used in this case on R2. At first glance, it meets all those requirements straight down the list.

Example D-8 *Troubleshooting Example 2 per Step 3B: Source and Destination Mismatch*

```
R2# show ip access-lists
Standard IP access list Step3B
   10 permit 10.3.3.0 0.0.0.127
   20 deny 10.4.4.0 0.0.1.255
   30 permit 10.0.0.0 0.255.255.255 (12 matches)
R2#
R2# show ip interface G0/1 | include Inbound
   Inbound access list is Step3B
```

The problem in this case is that the ACL has been enabled on R2's G0/1 interface, inbound. Per the figure, packets coming from a source address in subnets 10.3.3.0/25 and 10.4.4.0/23 should be forwarded out R2's G0/1 interface, rather than coming in that interface. So, do not let the matching logic in the ACL that perfectly mirrors the requirements fool you; make sure and check the location of the ACL, direction, and the location of the IP addresses.

Note that step 3C suggests a similar issue regarding matching well-known ports with TCP and UDP. The earlier section in this chapter titled "Matching TCP and UDP Port Numbers" has already discussed those ideas in plenty of detail. Just make sure to check where the server sits versus the location and direction of the ACL.

Steps 3D and 3E: Common Syntax Mistakes

Steps 3D and 3E describe a couple of common syntax mistakes. First, to match a TCP port in an ACL statement, you must use a **tcp** protocol keyword instead of **ip** or any other value. Otherwise, IOS rejects the command as having incorrect syntax. Same issue with trying to match UDP ports: a **udp** protocol keyword is required.

To match ICMP, IOS includes an **icmp** protocol keyword to use instead of **tcp** or **udp**. In fact, the main conceptual mistake is to think of ICMP as an application protocol that uses either UDP or TCP; it uses neither. To match all ICMP messages, for instance, use the **permit icmp any any** command in an extended named ACL.

Example Issue: Inbound ACL Filters Routing Protocol Packets

A router bypasses outbound ACL logic for packets the router itself generates. That might sound like common sense, but it is important to stop and think about that fact in context. A router can have an outgoing ACL, and that ACL can and will discard packets that the router receives in one interface and then tries to forward out some other interface. But if the router creates the packet, for instance, for a routing protocol message, the router bypasses the outbound ACL logic for that packet.

However, a router does not bypass inbound ACL logic. If an ACL has an inbound ACL enabled, and a packet arrives in that interface, the router checks the ACL. Any and all IPv4 packets are considered by the ACL—including important overhead packets like routing protocol updates.

For example, consider a seemingly good ACL on a router, like the step 3G ACL in Example D-9. That ACL lists a couple of **permit** commands, and has an implicit deny any at the end of the list. At first, it looks like any other reasonable ACL.

Example D-9 *Troubleshooting Example 2 per Step 3G: Filtering RIP by Accident*

```
R1# show ip access-lists
Standard IP access list Step3G
   10 permit host 10.4.4.1
   20 permit 10.3.3.0 0.0.0.127 (12 matches)
! using the implicit deny to match everything else
R1#
! On router R1:
R1# show ip interface G0/2 | include Inbound
   Inbound access list is Step3G
```

Now look at the location and direction (inbound on R1, on R1's G0/2) and consider that location versus the topology Figure D-5 for a moment. None of those **permit** statements match the RIP updates sent by R2, sent out R2's G0/1 interface toward R1. RIP messages use UDP (well-known port 520), and R2's G0/1 interface is 10.2.2.2 per the figure. R1 would match incoming RIP messages with the implicit deny all at the end of the list. The symptoms in this case, assuming only that one ACL exists, would be that R1 would not learn routes from R2, but R2 could still learn RIP routes from R1.

Of the three routing protocols discussed in the ICND1 and ICND2 books, RIPv2 uses UDP as a transport, while OSPF and EIGRP do not even use a transport protocol. As a result, to match RIPv2 packets with an ACL, you need the **udp** keyword and you need to match well-known port 520. OSPF and EIGRP can be matched with special keywords as noted in Table D-2. The table also list the addresses used by each protocol.

Table D-2 Key Fields for Matching Routing Protocol Messages

Protocol	Source IP Address	Destination IP Addresses	ACL Protocol Keyword
RIPv2	Source interface	224.0.0.9	**udp** (port 520)
OSPF	Source interface	224.0.0.5, 224.0.0.6	**ospf**
EIGRP	Source interface	224.0.0.10	**eigrp**

Example D-10 shows a sample ACL with three lines, one to match each routing protocol, just to show the syntax. Note that in this case, the ACL matches the address fields with the **any** keyword. You could include lines like these in any inbound ACL to ensure that routing protocol packets would be permitted.

Example D-10 *Example ACL that Matches all RIPv2, OSPF, and EIGRP with a Permit*

```
R1# show ip access-lists
ip access-list extended RoutingProtocolExample
  10 permit udp any any eq 520
  20 permit ospf any any
  30 permit eigrp any any
  remark a complete ACL would also need more statements here
R1#
```

ACL Interactions with Router-Generated Packets

Routers bypass outbound ACL logic for packets generated by that same router. This logic helps avoid cases in which a router discards its own overhead traffic. This logic applies to packets that a router creates for overhead processes like routing protocols, as well as for commands, like **ping** and **traceroute**. This section adds a few perspectives about how ACLs impact troubleshooting, and how this exception to outbound ACL logic applies, particularly commands used from the router CLI.

Local ACLs and a Ping from a Router

For the first scenario, think about a **ping** command issued by a router. The command generates packets, and the router sends those packets (holding the ICMP echo request messages) out one of the router interfaces, and typically some ICMP echo reply messages are received back. As it turns out, not all ACLs will attempt to filter those packets.

As a backdrop to discuss what happens, Figure D-6 illustrates a simple network topology with two routers connected to a serial link. Note that in this figure four IP ACLs exist, named A, B, C, and D, as noted by the thick arrows in the drawing. That is, ACL A is an outbound ACL on R1's S0/0/0, ACL B is an inbound ACL on R2's S0/0/1, and so on.

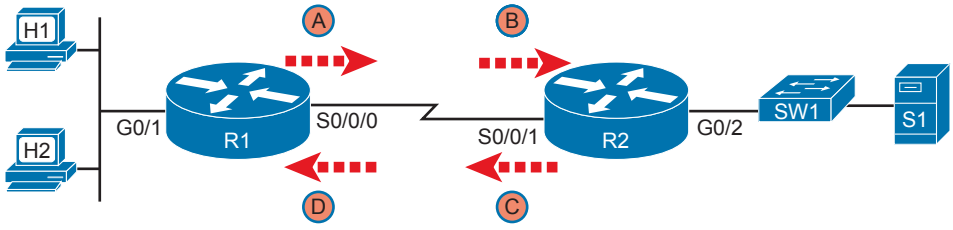


Figure D-6 *Sample Network with IP ACLs in Four Locations*

As an example, consider a **ping** command issued from R1's CLI (after a user connects to R1's CLI using SSH). The **ping** command pings server S1's IP address. The IPv4 packets with the ICMP messages flow from R1 to S1 and back again. Which of those four ACLs could possibly filter the ICMP Echo Request toward S1, and the ICMP Echo Reply back toward R1?

Routers bypass their own outbound ACLs for packets generated by the router, as shown in Figure D-7. Even though ACL A exists as an outgoing ACL on Router R1, R1 bypasses its own outgoing ACL logic of ACL A for the ICMP Echo Requests generated by R1.

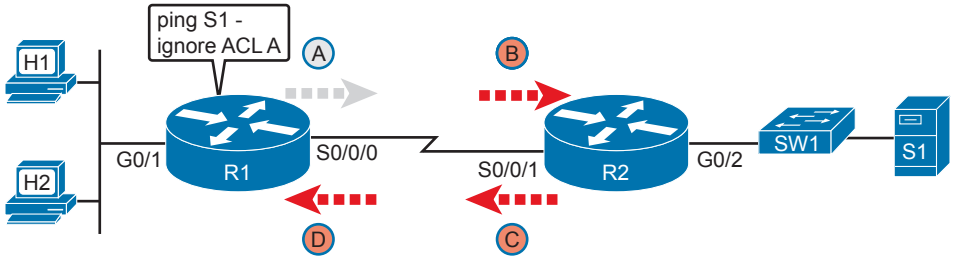


Figure D-7 *R1 Ignores Outgoing ACL for Packets Created by Its Own* **ping** *Command*

Router Self-Ping of a Serial Interface IPv4 Address

The previous example uses a router's **ping** command when pinging a host. However, network engineers often need to ping router IP addresses, including using a self-ping. The term *self-ping* refers to a ping of a device's own IPv4 address. And for point-to-point serial links, a self-ping actually sends packets over the serial link, which causes some interesting effects with ACLs.

When a user issues a self-ping for that local router's serial IP address, the router actually sends the ICMP echo request out the link to the other router. The neighboring router then receives the packet and routes the packet with the ICMP echo request back to the original router. Figure D-8 shows an example of a self-ping (**ping 172.16.4.1**) of Router R1's own IP address on a point-to-point serial link, with the ICMP echo request out the link to Router R2. At step 2, R2 treats it like any other packet not destined for one of R2's own IPv4 addresses: R2 routes the packet. Where? Right back to Router R1, as shown in the figure.

Now think about those four ACLs in the earlier figures compared to Figure D-8. R1 generates the ICMP echo request, so R1 bypasses outbound ACL A. ACLs B, C, and D could filter the packet. Note that the packet sent by R2 back to R1 is not generated by R2 in this case; R2 is just routing R1's original packet back to R1.

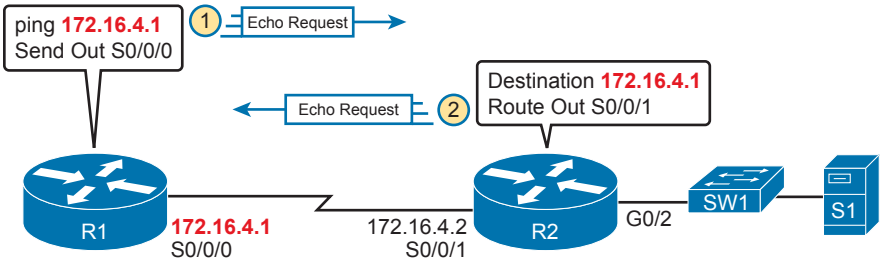


Figure D-8 *The First Steps in a Self-Ping on R1, for R1's S0/0/0 IP Address*

A self-ping of a serial interface actually tests many parts of a point-to-point serial link, as follows:

- The link must work at Layers 1, 2, and 3. Specifically, both routers must have a working (up/up) serial interface, with correct IPv4 addresses configured.
- ACLs B, C, and D must permit the ICMP echo request and reply packets.

So, when troubleshooting, if you choose to use self-pings and they fail, but the serial interfaces are in an up/up state, do not forget to check to see whether the ACLs have filtered the Internet Control Management Protocol (ICMP) traffic.

Router Self-Ping of an Ethernet Interface IPv4 Address

A self-ping of a router's own Ethernet interface IP address works a little like a self-ping of a router's serial IP address, but with a couple of twists:

- Like with serial interface, the local router interface must be working (in an up/up state); otherwise, the ping fails.
- Unlike serial interfaces, the router does not forward the ICMP messages physically out the interface, so security features on neighboring switches (like port security) or routers (like ACLs) cannot possibly filter the messages used by the **ping** command.
- Like serial interfaces, an incoming IP ACL on the local router does process the router self-ping of an Ethernet-based IP address.

Figure D-9 walks through an example. In this case, R2 issues a **ping 172.16.2.2** command to ping its own G0/2 IP address. Just like with a self-ping on serial links, R2 creates the ICMP echo request. However, R2 basically processes the ping down its own TCP/IP stack and back up again, with the ICMP echo never leaving the router's Ethernet interface. R2 does check the Ethernet interface status, showing a failure if the interface is not up/up. R2 does not apply outbound ACL logic to the packet, because R2 created the packet, but R2 will apply inbound ACL logic to the packet, as if the packet had been physically received on the interface.

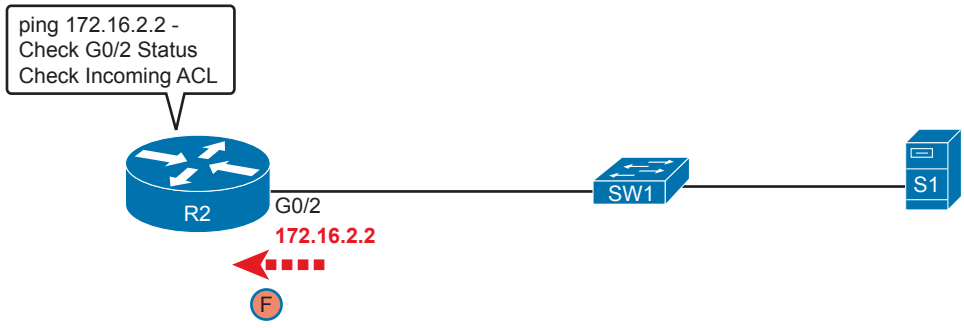


Figure D-9 *Self-Ping of a Router's Ethernet Address*

NOTE The content under the heading "Implementing HSRP" was most recently published for the 200-105 Exam in 2016, in Chapter 20 of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

Implementing HSRP

The goal of this section is to show enough of the operation of each tool to reinforce your understanding of configuring the basic functions of HSRP.

Configuring and Verifying Basic HSRP

HSRP configuration requires only one command on the two (or more) routers that want to share default router responsibilities with HSRP: the **standby** *group* **ip** *virtual-ip* interface subcommand. The first value defines the HSRP group number, which must match on both routers. The group number lets one router support multiple HSRP groups at a time on the same interface, and it allows the routers to identify each other based on the group. The command also configures the virtual IP address shared by the routers in the same group; the virtual IP address is the address the hosts in the VLAN use as their default gateway.

Example D-11 shows a configuration example where both routers use group 1, with virtual IP address 10.1.1.1, with the **standby 1 ip 10.1.1.1** interface subcommand.

Example D-11 *HSRP Configuration on R1 and R2, Sharing IP Address 10.1.1.1*

```
R1# show running-config
! Lines omitted for brevity
interface GigabitEthernet0/0
 ip address 10.1.1.9 255.255.255.0
 standby version 2
 standby 1 ip 10.1.1.1
 standby 1 priority 110
 standby 1 name HSRP-group-for-book
```

```
! The following configuration, on R2, is identical except for the HSRP priority and
! the interface IP address
R2# show running-config
! Lines omitted for brevity
interface GigabitEthernet0/0
 ip address 10.1.1.129 255.255.255.0
```

```
 standby version 2
 standby 1 ip 10.1.1.1
 standby 1 name HSRP-group-for-book
```

The configuration shows other optional parameters, as well. For instance, R1 has a priority of 110 in this group, and R2 defaults to 100. With HSRP, if the two routers are brought up at the same time, the router with the higher priority wins the election to become the active router. The configuration also shows a name that can be assigned to the group (when using **show** commands) and a choice to use HSRP Version 2. (This chapter provides more details on these settings in the coming pages.)

Once configured, the two routers negotiate the HSRP settings and choose which router will currently be active and which will be standby. With the configuration as shown, R1 will win the election and become active because of its higher (better) priority. Both routers reach the same conclusion, as confirmed with the output of the **show standby brief** command on both R1 and R2 in Example D-12.

Example D-12 *HSRP Status on R1 and R2 with* **show standby brief**

```
! First, the group status as seen from R1
R1# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Gi0/0       1    110   Active  local           10.1.1.129      10.1.1.1
! The output here on R2 shows that R2 agrees with R1.
R2# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Gi0/0       1    100   Standby 10.1.1.9        local           10.1.1.1
```

The **show standby brief** command packs a lot of detail in the output, so take your time and work through the highlighted fields. First, look at the Grp column for each command. This lists the HSRP group number, so when looking at output from multiple routers, you need to look at the lines with the same group number to make sure the data relates to that one HSRP group. In this case, both routers have only one group number (1), so it is easy to find the information.

Each line of output lists the local router's view of the HSRP status for that group. In particular, based on the headings, the **show standby brief** command identifies the following:

Interface: The local router's interface on which the HSRP group is configured

Grp: The HSRP group number

Pri: The local router's HSRP priority

State: The local router's current HSRP state

Active: The interface IP address of the currently active HSRP router (or "local" if the local router is HSRP active)

Standby: The interface IP address of the currently standby HSRP router (or "local" if the local router is HSRP standby)

Virtual IP: The virtual IP address defined by this router for this group

For instance, following the highlighted text in Example D-12, R2 believes that its own current state is standby, that the router with interface address 10.1.1.9 is active (which happens to be Router R1), with a confirmation that the "local" router (R2, on which this command was issued) is the standby router.

In comparison, the **show standby** command (without the **brief** keyword) lists a more detailed description of the current state, while repeating many of the facts from the **show standby brief** command. Example D-13 shows an example of the new information with the **show standby** command, listing several counters and timers about the HSRP protocol itself, plus the virtual MAC address 0000.0c9f.f001.

Example D-13 *HSRP Status on R1 and R2 with* **show standby**

```
R1# show standby
GigabitEthernet0/0 - Group 1 (version 2)
  State is Active
    6 state changes, last state change 00:12:53
  Virtual IP address is 10.1.1.1
  Active virtual MAC address is 0000.0c9f.f001
    Local virtual MAC address is 0000.0c9f.f001 (v2 default)
 Hello time 3 sec, hold time 10 sec
   Next hello sent in 1.696 secs
 Preemption disabled
 Active router is local
 Standby router is 10.1.1.129, priority 100 (expires in 8.096 sec)
 Priority 110 (configured 110)
 Group name is "HSRP-group-for-book" (cfgd)
```

```
! The output here on R2 shows that R2 agrees with R1.
R2# show standby
GigabitEthernet0/0 - Group 1 (version 2)
  State is Standby
    4 state changes, last state change 00:12:05
 Virtual IP address is 10.1.1.1
 Active virtual MAC address is 0000.0c9f.f001
   Local virtual MAC address is 0000.0c9f.f001 (v2 default)
 Hello time 3 sec, hold time 10 sec
   Next hello sent in 0.352 secs
 Preemption disabled
 Active router is 10.1.1.9, priority 110 (expires in 9.136 sec)
   MAC address is 0200.0101.0101
 Standby router is local
 Priority 100 (default 100)
 Group name is "HSRP-group-for-book" (cfgd)
```

HSRP Active Role with Priority and Preemption

HSRP defines some rules to determine which router acts as the active HSRP router and which acts as standby. Those rules also define details about when a standby router should

take over as active. The following list summarizes the rules; following the list, this section takes a closer look at those rules and the related configuration settings.

First, the HSRP rules. When a router (call it the local router) has an HSRP-enabled interface, and that interface comes up, the router sends HSRP messages to negotiate whether it should be active or standby. When it sends those messages, if it…

Step 1. …discovers no other HSRP routers in the subnet, the local router becomes the active router.

Step 2. …discovers an existing HSRP router, and both are currently negotiating to decide which should become the HSRP active router, the routers negotiate, with the router with the highest HSRP priority becoming the HSRP active router.

Step 3. …discovers an existing HSRP router in the subnet, and that router is already acting as the active router:

- **A.** If configured with no preemption (the default; **no standby preempt**), the local router becomes a standby router, even if it has a better (higher) priority.
- **B.** If configured with preemption (**standby preempt**), the local router checks its priority versus the active router; if the local router priority is better (higher), the local router takes over (preempts) the existing active router to become the new active HSRP router.

Steps 1 and 2 in the list are pretty obvious, but steps 3A and 3B could use a little closer look. For instance, the examples so far in this chapter show R1's G0/0 with a priority of 110 versus R2's G0/0 with priority 100. The **show** commands in Example D-13 show that R1 is currently the HSRP active router. That same example also lists a line for both R1 and R2 that confirms "preemption disabled," which is the default.

To show a test of step 3A logic, Example D-14 shows a process by which R1's G0/0 interface is disabled and then enabled again, but after giving Router R2 long enough to take over and become active. That is, R1 comes up but R2 is already HSRP active for group 1. The bottom of the example lists output from the **show standby brief** command from R2, confirming that R2 becomes HSRP active and R1 becomes standby (10.1.1.9), proving that R1 does not preempt R2 in this case.

Example D-14 *Showing How No Preemption Keeps R1 as Standby After R1 Recovers*

```
! First, R1's G0/0 is disabled and enabled; the ending log message shows a standby
! state.
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# shutdown
*Mar 8 18:10:29.242: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Active ->
Init
*Mar 8 18:10:31.205: %LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to
administratively down
*Mar 8 18:10:32.205: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther
net0/0, changed state to down
```

```
R1(config-if)#
R1(config-if)# no shutdown
R1(config-if)# ^Z
R1#
*Mar 8 18:11:08.355: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Speak ->
Standby
```

```
! Now from R2, note R2 is active, and 10.1.1.9 (R1) is standby
R2# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P  State  Active          Standby         Virtual IP
Gi0/1       1    100    Active local           10.1.1.9        10.1.1.1
```

If R1 had been configured with preemption for that previous scenario, R1 would have taken over from R2 when R1's interface came back up. Example D-15 shows exactly that. Before the output in Example D-15 was gathered, the network had been put back to the same beginning state as at the beginning of Example D-14, with R1 active and R2 as standby. Within Example D-15, R1's interface is shut down, then configured with preemption using the **standby 1 preempt** command, enabling preemption. Then, after enabling the interface again, R1 takes over as HSRP active, as shown at the bottom of the example's **show standby brief** command from R2. That output now shows the local router's state as Standby, and the active as 10.1.1.9 (R1).

Example D-15 *Showing How Preemption Causes R1 to Take Over As Active upon Recovery*

```
! First, R1's G0/0 is disabled and enabled; the ending log message shows a standby !
state.
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# shutdown
*Mar 8 18:10:29.242: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Active ->
Init
*Mar 8 18:10:31.205: %LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to
administratively down
*Mar 8 18:10:32.205: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther
net0/0, changed state to down
R1(config-if)# standby 1 preempt
R1(config-if)# no shutdown
R1(config-if)# ^Z
R1#
*Mar 8 18:19:14.355: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Listen ->
Active
```

```
! Now from R2, note it is active, and 10.1.1.9 (R1) is standby


*Mar 8 18:18:55.948: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Standby ->
Active
*Mar 8 18:19:14.528: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Active ->
Speak
```

```
*Mar 8 18:19:26.298: %HSRP-5-STATECHANGE: GigabitEthernet0/0 Grp 1 state Speak ->
Standby

R2# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Gi0/0       1    100   Standby 10.1.1.9        local           10.1.1.1
```

Note that it is the preemption setting on the router that is taking over (preempting) that determines if preemption happens. For instance, in this case, R1 came up when R2 was active; R1 was set to preempt; so R1 preempted R2.

HSRP Versions

Cisco IOS on routers and Layer 3 switches supports two versions of HSRP: versions 1 and 2. The versions have enough differences, like multicast IP addresses used and message formats, so that routers in the same HSRP group must use the same version. If two routers configured to be in the same HSRP group mistakenly configure to use different versions, they will not understand each other and ignore each other for the purposes of HSRP.

To configure the version, each interface/subinterface uses the **standby version** {**1** | **2**} interface subcommand. Note that the HSRP group number is not included in the command, because it sets the version for all HSRP messages sent out that interface/subinterface.

There are some good reasons to want to use the more recent HSRP version 2 (HSRPv2). For example, HSRPv1 existed before IPv6 became popular. Cisco enhanced HSRP to version 2 in part to make IPv6 support possible. Today, to use HSRP with IPv6 requires HSRPv2.

As another example of a benefit of HSRPv2, HSRP uses a Hello message, similar in concept to routing protocols, so that HSRP group members can realize when the active router is no longer reachable. HSRPv2 allows for shorter Hello timer configuration (as low as a small number of milliseconds), while HSRPv1 typically had a minimum of 1 second. So, HSRPv2 can be configured to react more quickly to failures with a lower Hello timer.

Beyond IPv6 support and shorter Hello timer options, other differences for version 2 versus version 1 include a different virtual MAC address base value and a different multicast IP address used as the destination for all messages. Table D-3 lists the differences between HSRPv1 and HSRPv2.

Table D-3 HSRPv1 Versus HSRPv2

Feature	Version 1	Version 2
IPv6 support	No	Yes
Smallest unit for Hello timer	Second	Millisecond
Range of group numbers	0..255	0..4095
MAC address used (xx or xxx is the hex group number)	0000.0C07.ACxx	0000.0C9F.Fxxx
IPv4 multicast address used	224.0.0.2	224.0.0.102
Does protocol use a unique identifier for each router?	No	Yes

Of the details in the table, make sure to look at the MAC addresses for both versions 1 and 2. Cisco reserves the prefixes of 0000.0C07.AC for HSRPv1 and 0000.0C9F.F for HSRPv2. HSRPv1, with 256 possible HSRP groups per interface, then uses the last two hex digits to identify the HSRP group. For example, an HSRP group 1 using version 1 would use a virtual MAC address that ends in hex 01. Similarly, because HSRPv2 supports 4096 groups per interface, the MAC address reserves three hex digits to identify the group. An HSRP group 1 using version 2 would use a virtual MAC address that ends in hex 001.

NOTE The content under the heading "Gateway Load Balancing Protocol (GLBP)" was most recently published for the 200-105 Exam in 2016, in Appendix K of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

Gateway Load Balancing Protocol (GLBP)

This section first discusses GLBP concepts, followed by GLBP configuration.

GLBP Concepts

Hot Standby Router Protocol (HSRP) and Virtual Router Redundancy Protocol (VRRP), which were introduced before Gateway Load Balancing Protocol (GLBP), balanced the packet load per subnet. However, because traffic loads vary unpredictably from subnet to subnet, Cisco wanted a First Hop Redundancy Protocol (FHRP) option with better load-balancing options than just the per-subnet load balancing of HSRP and VRRP. To meet that need, Cisco introduced GLBP.

GLBP balances the packet load per host by using an active/active model in each subnet. Each GLBP router in a subnet receives off-subnet packets from some of the hosts in the subnet. Each host still remains unaware of the FHRP, allowing the hosts to configure the same default gateway/router setting and for the hosts to make no changes when a router fails.

GLBP creates a world that at first glance looks like HSRP, but with a few twists that let GLBP balance the traffic. Like HSRP, all the routers configure a virtual IP address, which is the IP address used by hosts as their default router. Like with HSRP, hosts use a default router setting that points to the virtual IP address, and that setting does not need to change. GLBP differs from HSRP with regard to the MAC addresses it uses and the Address Resolution Protocol (ARP) process, because GLBP actually uses ARP Reply messages to balance traffic from different hosts through different routers.

With GLBP, one router acts in a special role called the *active virtual gateway* (AVG). The AVG replies to all ARP requests for the virtual IP address. Each router has a unique virtual MAC address, so that the AVG can reply to some ARP Requests with one virtual MAC, and some with the other. As a result, some hosts in the subnet send frames to the Ethernet MAC address of one of the routers, with other hosts sending their frames to the MAC address of the second router.

As an example, Figure D-10 shows the process by which a GLBP balances traffic for host A based on the ARP Reply sent by the AVG (R1). The two routers support virtual IP address 10.1.1.1, with the hosts using that address as their default router setting.

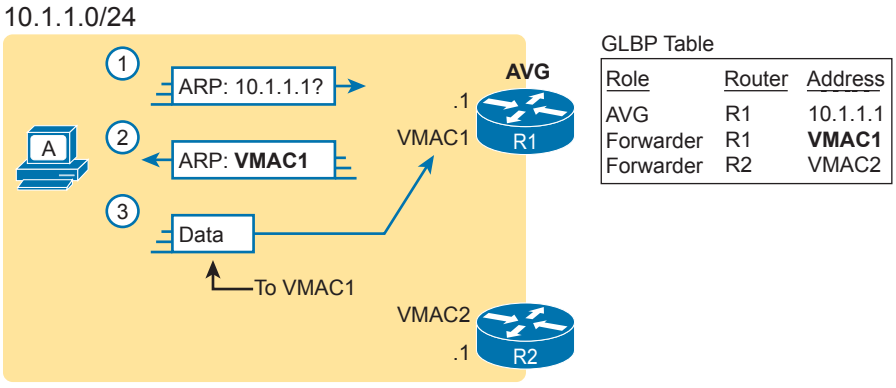


Figure D-10 *GLBP Directs Host A by Sending Back ARP Reply with R1's VMAC1*

The figure shows three messages, top to bottom, with the following action:

1. Host A has no ARP table entry for its default router, 10.1.1.1, so host A sends an ARP Request to learn 10.1.1.1's MAC address.
2. The GLBP AVG, R1 in this case, sends back an ARP Reply. The AVG chooses to include its own virtual MAC address in the ARP Reply, VMAC1.
3. Future IP packets sent by host A are encapsulated in Ethernet frames, destined to VMAC1, so that they arrive at R1.

From now on, host A sends off-subnet packets to R1 due to host A's ARP table entry for its default gateway (10.1.1.1). Host A's ARP table entry for 10.1.1.1 now refers to a MAC address on R1 (VMAC1), so packets host A sends off-subnet flow through R1.

To balance the load, the AVG answers each new ARP Request with the MAC addresses of alternating routers. Figure D-11 continues the load-balancing effect with the ARP Request for 10.1.1.1 coming from host B. The router acting as AVG (R1) still sends the ARP Reply, but this time with R2's virtual MAC (VMAC2).

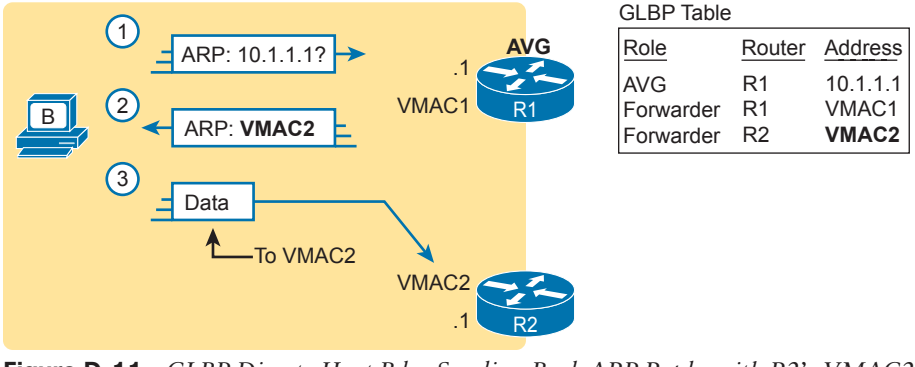


Figure D-11 *GLBP Directs Host B by Sending Back ARP Reply with R2's VMAC2*

Here are the steps in the figure:

1. Host B sends an ARP Request to learn 10.1.1.1's MAC address.
2. The GLBP AVG (R1) sends back an ARP Reply, listing VMAC2, R2's virtual MAC address.
3. For future packets sent off-subnet, host B encapsulates the packets in Ethernet frames, destined to VMAC2, so that they arrive at R2.

The process shown in Figures D-10 and D-11 balances the traffic, per host, but the routers must also be ready to take over for the other router if it fails. GLBP refers to each router as a *forwarder*. When all is well, each router acts as forwarder for its own virtual MAC address, but it listens to GLBP messages to make sure the other forwarders are still working. If another forwarder fails, the still-working forwarder takes over the failed forwarder's virtual MAC address role and continues to forward traffic.

Configuring and Verifying GLBP

GLBP configuration mimics HSRP configuration to a great degree.

Example D-16 shows a GLBP configuration with both routers using GLBP group 1, with virtual IP address 10.1.1.1, with the **glbp 1 ip 10.1.1.1** interface subcommand.

Example D-16 *GLBP Configuration on R1 and R2, Sharing IP Address 10.1.1.1*

```
! First, the configuration on R1
R1# show running-config
! Lines omitted for brevity
interface GigabitEthernet0/0
 ip address 10.1.1.9 255.255.255.0
 glbp 1 ip 10.1.1.1
 glbp 1 priority 110
 glbp 1 name GLBP-group-for-book
```

```
! The following configuration, on R2, is identical except for
! the interface IP address, and the GLBP priority
R2# show running-config
! Lines omitted for brevity
interface GigabitEthernet0/0
 ip address 10.1.1.129 255.255.255.0
 glbp 1 ip 10.1.1.1
 glbp 1 name GLBP-group-for-book
```

Once configured, the two routers negotiate as to which will be the AVG. As with HSRP, if both come up at the same time, R1 will win, with a priority set to 110 with the **glbp 1 priority 110** command versus R2's default priority of 100. However, if either router comes up before the other, that router goes ahead and takes on the AVG role.

Sifting through the GLBP **show** command output takes a little more work than with HSRP, in particular because of the added detail in how GLBP works. First, consider the **show glbp brief** command on Router R1, as shown in Example D-17. (Note that many **show glbp** commands have the same options as equivalent HSRP **show standby** commands.)

Example D-17 *GLBP Status on R1 with* **show glbp brief**

```
R1# show glbp brief
Interface   Grp  Fwd  Pri State   Address         Active router   Standby router
Gi0/0       1    -    110 Active  10.1.1.1        local           10.1.1.129
Gi0/0       1    1    -   Listen  0007.b400.0101  10.1.1.129      -
Gi0/0       1    2    -   Active  0007.b400.0102  local
```

Before looking at the right side of the output, first consider the context for a moment. This example lists a heading line and three rows of data. These data rows are identified by the Grp and Fwd headings, short for Group and Forwarder. With only one GLBP group configured, R1 lists lines only for group 1. More important, each row defines details about a different part of what GLBP does, as follows:

Fwd is -: This line refers to none of the forwarders, and instead describes the AVG.

Fwd is 1: This line describes GLBP forwarder (router) 1.

Fwd is 2: This line describes GLBP forwarder (router) 2.

The output usually lists the line about the AVG first, as noted with a dash in the Forwarder column. Now look at the highlighted portions on the right of Example D-17. This line will list the virtual IP address and identify the active AVG and the standby AVG. This particular command, from Router R1, lists R1 itself ("local") as the active router. So, R1 is the current AVG.

Each of the next two lines lists status information about one of the forwarder roles; that is, a router that uses a virtual MAC address, receives frames sent to that address, and routes the packets encapsulated in those frames. To that end, the Address column lists MAC addresses, specifically the virtual MAC addresses used by GLBP, and not the interface MAC addresses.

Each forwarder row also identifies the router that currently uses the listed virtual MAC in the Active Router column. In Example D-17, 0007.b400.0101 is used by the router with interface IP address 10.1.1.129 (which happens to be R2). 0007.b400.0102 is supported by the local router (the router on which the **show** command was issued), which is R1.

The brief output of the **show glbp brief** command lists many details, but it takes some effort to learn how to sift through it all. For more perspective on the output, Example D-18 lists this same **show glbp brief** command, this time on R2. Note that the Fwd column again identifies the first line of output as being about the AVG, with the next two lines about the two forwarders.

Example D-18 *GLBP Status on R2 with* **show glbp brief**

```
R2# show glbp brief
Interface   Grp  Fwd  Pri State    Address         Active router     Standby router
Gi0/0       1    -    100 Standby  10.1.1.1        10.1.1.9          local
Gi0/0       1    1    -   Active   0007.b400.0101  local             -
Gi0/0       1    2    -   Listen   0007.b400.0102  10.1.1.9          -
```

The State column in the output in Examples D-17 and D-18 can pull the GLBP concepts together. First, to define the meaning of the state values, the following short list defines

D

the states expected for the first line of output, about the AVG, and then about each GLBP forwarder:

AVG: One router should be the active AVG, with the other acting as standby, ready to take over the AVG role if the AVG fails.

Each forwarder: One router should be active, while the other should be listening, ready to take over that virtual MAC address if that forwarder fails.

Table D-4 collects the values of the State column from Examples D-17 and D-18 for easier reference side by side. Note that, indeed, each line has either an active/standby pair (for the AVG) or an active/listen pair (for the forwarder function).

Table D-4 Comparing Local State in **show glbp brief** Commands

Row Is About...	Fwd Column Value	R1 State	R2 State
AVG	-	Active	Standby
Forwarder 1	1	Listen	Active
Forwarder 2	2	Active	Listen

Finally, the **show glbp** command lists a more detailed view of the current GLBP status. Example D-19 shows a sample from Router R1. Note that the first half of the output has similar information compared to HSRP's **show standby** command, plus it lists the IP and MAC addresses of the routers in the GLBP group. Then, the end of the output lists a group of messages per GLBP forwarder.

Example D-19 *GLBP Status on R1 with* **show glbp**

```
R1# show glbp
GigabitEthernet0/0 - Group 1
  State is Active
    2 state changes, last state change 00:20:59
  Virtual IP address is 10.1.1.1
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.112 secs
  Redirect time 600 sec, forwarder timeout 14400 sec
  Preemption disabled
  Active is local
  Standby is 10.1.1.129, priority 100 (expires in 8.256 sec)
  Priority 110 (configured)
  Weighting 100 (default 100), thresholds: lower 1, upper 100
  Load balancing: round-robin
  IP redundancy name is "GLBP-group-for-book"
  Group members:
    0200.0101.0101 (10.1.1.9) local
    0200.0202.0202 (10.1.1.129)
  There are 2 forwarders (1 active)
  Forwarder 1
  State is Listen
    2 state changes, last state change 00:20:34
```

```
MAC address is 0007.b400.0101 (learnt)
Owner ID is 0200.0202.0202
Redirection enabled, 598.272 sec remaining (maximum 600 sec)
Time to live: 14398.272 sec (maximum 14400 sec)
Preemption enabled, min delay 30 sec
Active is 10.1.1.129 (primary), weighting 100 (expires in 8.352 sec)
Client selection count: 1
Forwarder 2
  State is Active
    1 state change, last state change 00:24:25
MAC address is 0007.b400.0102 (default)
Owner ID is 0200.0101.0101
Redirection enabled
Preemption enabled, min delay 30 sec
Active is local, weighting 100
Client selection count: 1
```

NOTE The content under the heading "Implementing Simple Network Management Protocol" was most recently published for the 200-105 Exam in 2016, in Chapter 26 of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

Implementing Simple Network Management Protocol

This section includes details of how to implement SNMPv2c and SNMPv3.

Implementing SNMP Version 2c

The exam topics mention SNMPv2c and SNMPv3 by name. As it turns out, SNMPv1 and SNMPv2c configuration is very similar, because both use communities. SNMPv3 varies quite a bit, mainly to implement the better SNMPv3 security features. This next section shows how to configure and verify SNMPv2c.

Configuring SNMPv2c Support for Get and Set

SNMP configuration in Cisco IOS routers and switches works a little differently than many other IOS features. First, the SNMP configuration exists in a series of global commands; there is no SNMP agent configuration mode in which to collect subcommands. Secondly, no single command enables the SNMP agent. Instead, IOS typically defaults for the SNMP agent to be disabled. Then, the first time an **snmp-server** global command is configured, IOS enables the SNMP agent.

NOTE To disable the SNMP agent, you must remove all the **snmp-server** commands. You can do this with a single **no snmp-server** command (with no parameters).

With that backdrop, a typical SNMPv2c configuration requires only one or two settings. To be useful, the agent needs at least a read-only (RO) community string. The agent will not reply to SNMPv2c Get messages without at least the RO community string configured. The network engineer may also want the agent to have a read-write (RW) community string, to support Set messages.

NOTE When configuring an RW community, use some caution: configuring an RW community means that you have defined a clear-text password that can be used to configure many settings on the router or switch.

The following checklist details the commands used to configure SNMPv2c on a Cisco router or switch. This list shows the method to configure the RO and RW communities, plus a few optional but common settings (location and contact information).

Step 1. Use the **snmp-server community** *communitystring* **RO** [**ipv6** *acl-name*] [*acl-name*] command in global configuration mode to enable the SNMP agent (if not already started), set the read-only community string, and restrict incoming SNMP messages based on the optional referenced IPv4 or IPv6 ACL.

Step 2. (Optional) Use the **snmp-server community** *communitystring* **RW** [**ipv6** *acl-name*] [*acl-name*] command in global configuration mode to enable the SNMP agent (if not already started), set the read-write community string, and restrict incoming SNMP messages based on the optional referenced IPv4 or IPv6 ACL.

Step 3. (Optional) If referenced by an **snmp-server community** command, configure an IPv4 or IPv6 ACL, with the same name or number referenced by the **snmp-server community** command, with the ACL permitting by matching the source IPv4 or IPv6 address of the allowed SNMP management hosts.

Step 4. (Optional) Use the **snmp-server location** *text-describing-location* command in global configuration mode to document the location of the device.

Step 5. (Optional) Use the **snmp-server contact** *contact-name* command in global configuration mode to document the person to contact if problems occur.

NOTE In the SNMP model, the SNMP agent acts as a server, with the NMS (SNMP Manager) acting as an SNMP client by requesting information with Get messages. The IOS **snmp-server** command happens to emphasize the idea that the SNMP agent on a router or switch acts as the SNMP server.

Example D-20 shows a sample configuration based on Figure D-12. The examples in this section come from Router R1, although the exact same SNMP configuration syntax could be used in the LAN switches or in R2. (The configuration of the location information would likely differ for each device, however.) Note that the configuration creates an IPv4 ACL that permits traffic with source IP address 10.1.3.3, which is the address of the NMS shown in the figure. It then defines read-only and read-write communities, along with the location and contact name for the router.

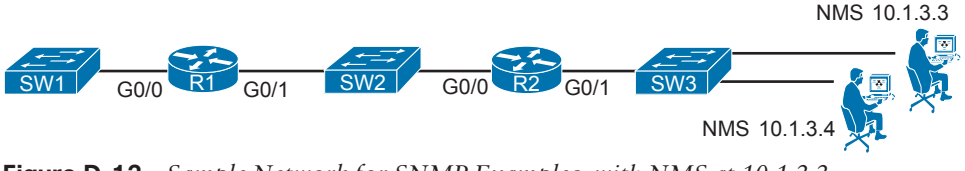


Figure D-12 *Sample Network for SNMP Examples, with NMS at 10.1.3.3*

Example D-20 *Configuring SNMP Version 2c on Router R1 to Support Get and Set*

```
ip access-list standard ACL_PROTECTSNMP
 permit host 10.1.3.3
!
snmp-server community secretROpw RO ACL_PROTECTSNMP
snmp-server community secretRWpw RW ACL_PROTECTSNMP
snmp-server location Atlanta
snmp-server contact Tyler B
```

To begin managing Router R1 (or any of the other devices that use the same community strings), the SNMP manager at address 10.1.3.3 now needs to configure the community strings listed in Example D-20.

Configuring SNMPv2c Support for Trap and Inform

For an SNMPv2c agent in a router or switch to be able to send unsolicited notifications to an SNMP manager (that is, to send Trap and Inform messages), the device needs to be configured with the **snmp-server host** command. This command references the NMS to which the Traps or Informs should be sent, along with the SNMP version.

Beyond telling the SNMP agent the hostname or address of the NMS, the agent typically needs to know the *notification community* string used by the NMS. Think of the RO and RW community strings as protecting the SNMP agent from the messages originated by an NMS (Get or Set Requests), so the agent requires the NMS to supply the correct RO or RW community string. For Traps and Informs, the NMS can protect itself from the Trap and Inform messages originated by SNMP agents by requiring those agents to include the notification community with those messages. The agent can configure this value on the **snmp-server host** command as well.

The following list details the command to enable the sending of SNMPv2c Trap or Inform messages to an NMS:

Step 1. Use the **snmp-server host** {*hostname* | *ip-address*} [**informs**] **version 2c** *notification-community* command in global configuration mode to configure the SNMP agent to send either SNMPv2c Traps (default) or Informs to the listed host. Use this command once for each host to which this device should send Traps.

Step 2. Use the **snmp-server enable traps** command in global configuration mode to enable the sending of all supported types of Trap and Inform messages.

Example D-21 shows a sample configuration. In most cases, you would send either Traps or Informs to a particular NMS, but not both. So, for this example, the configuration shows how to configure to send Traps to one host (10.1.3.3), and Informs to another host (10.1.3.4). Note that this configuration is added to Router R1 from Figure D-12, but it could have been added to Router R2 or to any of the LAN switches as well.

Example D-21 *Configuring SNMP Version 2c on Router R1 to Support Sending Traps*

```
snmp-server host 10.1.3.3 version 2c secretTRAPpw
snmp-server host 10.1.3.4 informs version 2c secretTRAPpw
snmp-server enable traps
```

Verifying SNMPv2c Operation

Example D-22 displays some of the status information based on the configuration seen in the previous two examples. The variations on the **show snmp** command highlight several configuration settings. For example, the **show snmp community** command repeats the community string values, with reference to any attached IPv4 or IPv6 ACLs. The **show snmp host** command lists the IP address or hostname of the NMS referenced by each **snmp-server host** configuration command.

Example D-22 *Confirming SNMPv2c Configuration Settings on Router R1*

```
R1# show snmp community

Community name: secretROpw
Community Index: secretROpw
Community SecurityName: secretROpw
storage-type: nonvolatile    active access-list: ACL_PROTECTSNMP

Community name: secretRWpw
Community Index: secretRWpw
Community SecurityName: secretRWpw
storage-type: nonvolatile    active access-list: ACL_PROTECTSNMP

Community name: secretTRAPpw
Community Index: secretTRAPpw
Community SecurityName: secretTRAPpw
storage-type: nonvolatile active

R1# show snmp location
Atlanta

R1# show snmp contact
Tyler B

R1# show snmp host
Notification host: 10.1.3.4     udp-port: 162    type: inform
user: secretTRAPpw      security model: v2c

Notification host: 10.1.3.3     udp-port: 162    type: trap
user: secretTRAPpw      security model: v2c
```

The **show snmp** command takes the opposite approach from the commands in Example D-22, focusing almost completely on status and counter information, rather than repeating configuration settings. This command lists dozens of lines of detailed information, so the sample in Example D-23 shows just enough of the output to give you a sense of the kinds of information found there, with comments following the example.

Example D-23 *Finding SNMPv2c Message Load on Router R1*

```
R1# show snmp
Chassis: FTX162883H0
Contact: Tyler B
Location: Atlanta
7735 SNMP packets input
    0 Bad SNMP version errors
    9 Unknown community name
    0 Illegal operation for community name supplied
    2 Encoding errors
    51949 Number of requested variables
    2 Number of altered variables
    3740 Get-request PDUs
    3954 Get-next PDUs
    7 Set-request PDUs
    0 Input queue packet drops (Maximum queue size 1000)
7850 SNMP packets output
    0 Too big errors (Maximum packet size 1500)
    0 No such name errors
    0 Bad values errors
    0 General errors
    7263 Response PDUs
    126 Trap PDUs
! Lines omitted for brevity
```

The output in Example D-23 was taken from Router R1 as shown in the earlier examples, after doing some testing from the NMS at address 10.1.3.3. The highlighted items point out the number of SNMP packets received (input) and sent (output), as well as the number of requested MIB variables—that is, the number of variables requested in different SNMP Get requests. (Note that SNMP also supports the GetNext and GetBulk commands, so a single NMS user click can cause the NMS to Get many variables from an agent; thus, it is not unusual for the requested variables counter to get very large.) The output also shows that seven Set requests were received, resulting in two changes to variables. The fact that two Set requests changed variables is a good fact to know if you are wondering if someone has reconfigured something on the device using SNMP.

Implementing SNMP Version 3

SNMPv3 configuration on Cisco routers and switches has some commands in common with SNMPv2c configuration, and some completely different commands. The configuration to support sending Traps and Informs, using the **snmp-server host** and **snmp-server enable traps** commands, works almost identically, with a few small differences. However, SNMPv3 replaces all references to communities, and as a result does not use the **snmp-server community** command at all. Instead, it uses the **snmp-server group** and **snmp-server user** commands to configure the security features available to SNMPv3.

SNMPv3 has many more configuration options, and it is easy to get confused by the details. So, to get started, first look at a short SNMPv3 configuration example, as shown in Example

D-24. The example highlights the values you would have to choose, but the values are either text fields (names and passwords) or the IP address of the NMS. This configuration could be used to replace the SNMPv2c configuration and use username/password authentication. The requirements met in the example are

- Use SNMPv3 authentication (basically replacing SNMPv2 communities).
- Use username Youdda and authentication password madeuppassword (in your network, you would choose your own values).
- Do not use SNMPv3 privacy (that is, message encryption).
- Allow both read (Get) and write (Set) access.
- Send Traps to an NMS (10.1.3.3), authenticating with the same username.

Example D-24 *Configuring SNMPv3 on R1—Authentication Only*

```
R1(config)# snmp-server group BookGroup v3 auth write v1default
R1(config)# snmp-server user Youdda BookGroup v3 auth md5 madeuppassword
R1(config)# snmp-server host 10.1.3.3 version 3 auth Youdda
```

Given the list of requirements, you could probably just read the configuration in Example D-24, compare that to the list of requirements preceding the example, and correctly guess what most of the command parameters mean. However, we need to get into more detail to work through these commands and their options so that you understand the entire configuration, which is exactly what the next few pages do.

SNMPv3 Groups

SNMPv3 authentication uses a username/password combination. When Cisco created its SNMPv3 implementation in IOS, it realized that it might be useful to have groups of users that use some of the same security settings. So, rather than have each **snmp-server user** command (the command that defines a user) define every single security parameter, Cisco put some of the security configuration settings into the **snmp-server group** command. This command holds SNMPv3 security settings that are often the same between a group of SNMPv3 users; each **snmp-server user** command then refers to one SNMP group. This next topic explores those security parameters defined on the **snmp-server group** command.

Figure D-13 shows the entire **snmp-server group** command. The required parameters on the left include a name that the network engineer can make up; it only needs to match other commands on the local router. For SNMPv3 configuration, the **v3** keyword would always be used. The text following this figure then details the rest of the parameters in the figure.

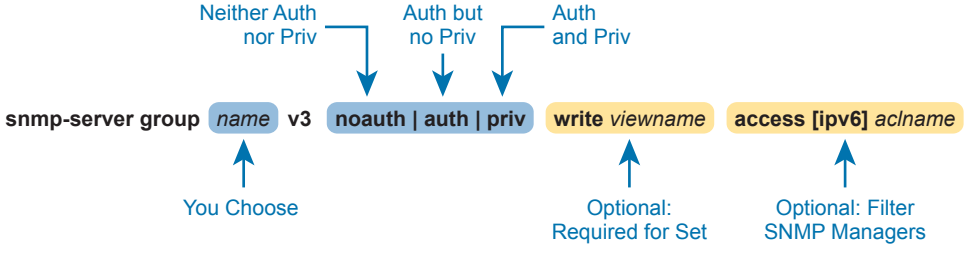


Figure D-13 *SNMPv3 Groups—Configuration Command Parameters*

The next parameter in the command configures this group of users to use one of three SNMPv3 *security levels*. As you can see from the summary in Table D-5, all three security levels provide message integrity for their messages, which confirms that the message has not been changed in transit. The **auth** option adds authentication to message integrity, using a username and password, with IOS storing the password with a hash and never sending the password as clear text. The last increase in security level, configured by using the **priv** security level, causes the SNMP manager and agent to encrypt the entire SNMP packet for all SNMP messages sent, in addition to performing message integrity and authentication.

Table D-5 SNMPv3 Security Levels Keywords and Their Meanings

Command Keyword	Keyword in Messages	Checks Message Integrity?	Performs Authentication?	Encrypts Messages?
noauth	noAuthNoPriv	Yes	No	No
auth	authNoPriv	Yes	Yes	No
priv	authPriv	Yes	Yes	Yes

Continuing to look at the **snmp-server group** command in Figure D-13, notice that it ends with an optional ACL to filter packets. This same idea is used in SNMPv2c to reference an IPv4 or IPv6 ACL to filter incoming messages coming from the SNMP manager.

So far, the discussion has ignored one part of the **snmp-server group** command: the idea of SNMPv3 MIB views. MIB views define a subset of the MIB. IOS supplies a series of MIB views for us, and you can define your own MIB views if you like. However, this book discusses only one predefined MIB view that goes by the name *v1default*, which is a MIB view that includes all the useful parts of the MIB. Instead of focusing on the depths of how you might create different views of a router or switch MIB that has literally thousands of variables, focus on how the **snmp-server group** command uses that one MIB view that includes the majority of the MIB.

By default, each SNMPv3 group, as defined with the **snmp-server group** command, has a read MIB view of v1default, and no write view. As a result, the SNMP agent will process received SNMPv3 Get requests, but not process received SNMPv3 Set requests. That complete lack of a write MIB view basically results in read-only behavior for the SNMP agent, as shown at the top of Figure D-14.

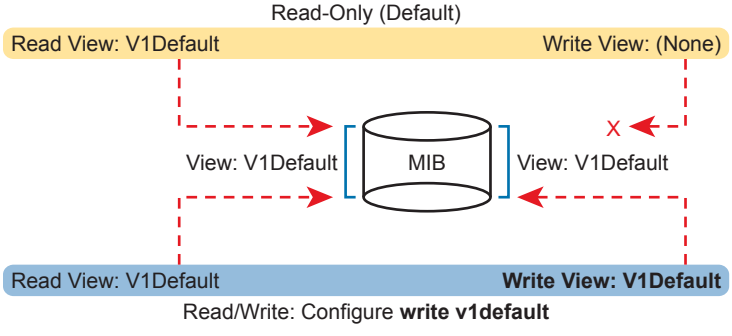


Figure D-14 *SNMPv3 Views Creating Read-Only and Read-Write Effect*

D

The bottom of the figure shows the concept behind configuring an SNMP group with the **write v1default** parameters, causing the group to use the same write view of the MIB that is used for reading the MIB. By including **write v1default** in the **snmp-server group** command, you migrate from a default operation of allowing only Gets to now also allowing Sets.

To pull these ideas together, Example D-25 shows four similar SNMPv3 groups, which could later be referenced by **snmp-server user** commands. Two commands use the parameters **write v1default**, and two do not, so two groups create read-write (Get and Set) support, and two groups create read-only (Get only) support. Also, note that two groups refer to an IPv4 ACL by name (SNMPACL), and two do not. The ends of the lines in the example list comments about each command.

Example D-25 *SNMPv3 Groups—Comparisons with Write Views and ACL Security*

```
ip access-list standard SNMPACL
   permit host 10.1.3.3
!
snmp-server group Group1 v3 noauth                 ! No writes, no ACL
snmp-server group Group2 v3 noauth write v1default ! Allows writes, no ACL
snmp-server group Group3 v3 noauth access SNMPACL  ! No writes, uses ACL
snmp-server group Group4 v3 noauth write v1default access SNMPACL ! Allows writes,
uses ACL
```

Note that while all four examples use an authentication type of **noauth**, groups could be defined that use the **auth** and **priv** types as well. Configuring groups with any one of the security levels does not change the meaning and use of the **write** and **access** keywords and their parameters. The security level simply needs to match the security level configured on the **snmp-server user** commands that refer to the group by name, as seen in the next section.

SNMPv3 Users, Passwords, and Encryption Keys

The **snmp-server user** command configures other security parameters for the SNMP agent. In particular, it configures

- The username
- The authentication password and the authentication hash algorithm (MD5 or SHA)
- The encryption key and the encryption algorithm (DES, 3DES, AES)
- A reference to an **snmp-server group** command by name, which holds more security configuration

The **snmp-server user** command still has plenty of moving parts, even with some of the security configuration sitting in the **snmp-server group** command. Figure D-15 connects these configuration concepts together, showing both commands in one place. Some explanation follows the figure.

The **snmp-server user** command creates the username itself. The network engineer can make up a name. The next two parameters must match the chosen **snmp-server group** command associated with this user, by matching the group name and the **v3** keyword (meaning SNMPv3). Any mistakes here will result in this SNMP user not being associated with the SNMP group.

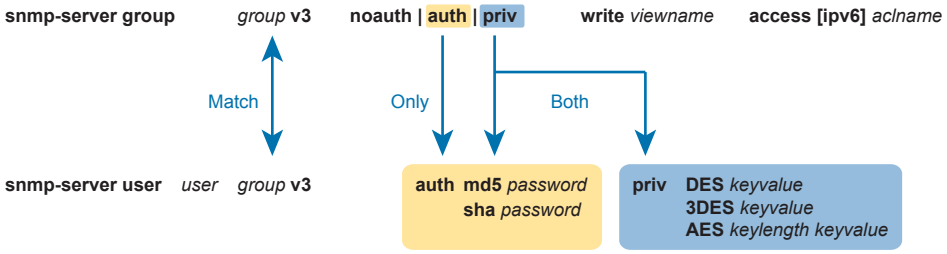


Figure D-15 *SNMPv3 Users and Groups: Configured*

You must pay particular attention to the security type in the associated **snmp-server group** command, because it dictates what parameters must be configured toward the end of the **snmp-server user** command. As noted in Figure D-15 with the arrowed lines, the use of the **auth** keyword in the **snmp-server group** command requires that you configure authentication parameters for the user in the **snmp-server user** command: the password and the choice of authentication hash algorithms. If using the **priv** keyword in the **snmp-server group** command, the **snmp-server user** command must define both authentication and privacy parameters as shown in the figure.

NOTE IOS allows you to misconfigure the **snmp-server user** command so that it omits the **auth** or **priv** keyword, even when the referenced **snmp-server group** command uses the **auth** or **priv** parameter. However, that misconfiguration causes the SNMP agent to not be able to communicate with the SNMP manager. For instance, if the **snmp-server user** command omits the **auth** keyword and associated parameters, but the **snmp-server group** command uses the **auth** keyword, IOS accepts the configuration commands, but authentication fails when the agent and NMS try to communicate.

Example D-26 shows a series of **snmp-server group** and matching **snmp-server user** commands, one after the other, so you can more easily see the parameters. Note that the **snmp-server group** commands do not include the optional parameters to enable writes (**write v1default**) or to use an ACL, just to reduce clutter.

Example D-26 *SNMPv3 Configuration Samples: Groups and Users*

```
! The group uses noauth, so the user Youdda1 has no auth nor priv keyword
snmp-server group BookGroup1 v3 noauth
snmp-server user Youdda1 BookGroup1 v3

! The next group uses auth, so the next two users use the auth keyword, but not priv
snmp-server group BookGroup2 v3 auth
snmp-server user Youdda2 BookGroup2 v3 auth md5 AuthPass2
snmp-server user Youdda3 BookGroup2 v3 auth sha AuthPass3

! The next group uses priv, so the next users use both the auth and priv keywords.
snmp-server group BookGroup3 v3 priv
snmp-server user Youdda4 BookGroup3 v3 auth md5 AuthPass3 priv des PrivPass4
snmp-server user Youdda5 BookGroup3 v3 auth md5 AuthPass3 priv 3des PrivPass5
snmp-server user Youdda6 BookGroup3 v3 auth sha AuthPass4 priv aes 128 PrivPass6
```

Note that the example also shows samples of several authentication and encryption options, as listed in Figure D-15.

Verifying SNMPv3

Verifying SNMPv3 operation begins with confirming the details of the SNMPv3 configuration. You can of course find these with the **show running-config** command, but two commands in particular repeat the configuration settings. Example D-27 shows the output from one of those commands, **show snmp user**, taken from Router R1 after adding the configuration listed in Example D-26.

Example D-27 *Verifying SNMPv3 Configuration Settings*

```
R3# show snmp user
User name: Youdda1
Engine ID: 800000090300D48CB57D8200
storage-type: nonvolatile      active
Authentication Protocol: None
Privacy Protocol: None
Group-name: BookGroup1

User name: Youdda2
Engine ID: 800000090300D48CB57D8200
storage-type: nonvolatile      active
Authentication Protocol: MD5
Privacy Protocol: None
Group-name: BookGroup2

! Skipping Youdda3, Youdda4, and Youdda5 for brevity

User name: Youdda6
Engine ID: 800000090300D48CB57D8200
storage-type: nonvolatile      active
Authentication Protocol: SHA
Privacy Protocol: AES128
Group-name: BookGroup3
```

In particular, work through the highlighted output for users Youdda1, Youdda2, and Youdda6, as compared to the configuration in Example D-26. All the highlighted entries basically repeat the settings from the configuration.

Example D-28 lists output from the **show snmp group** command, which also confirms configuration settings from Example D-26. The most challenging thing to find in this output is what is missing, rather than what is there. Note that this command does not list the SNMP usernames that happen to refer to this group. Also, for groups that do not use an ACL, there is no obvious text that states that no ACL is used. Make sure to compare the output for BookGroup1, which uses an ACL, and the output for BookGroup2, which does not use an ACL.

Example D-28 *Verifying SNMPv3 Using* **show snmp group**

```
R3# show snmp group
groupname: BookGroup1                       security model:v3 noauth
contextname: <no context specified>         storage-type: nonvolatile
readview : v1default                        <no writeview specified>
notifyview: <no notifyview specified>
row status: active       access-list: ACL_PROTECTSNMP

groupname: BookGroup2                       security model:v3 auth
contextname: <no context specified>         storage-type: nonvolatile
readview : v1default                        writeview: <no writeview specified>
notifyview: <no notifyview specified>
row status: active
! Lines omitted for brevity
```

Implementing SNMPv3 Notifications (Traps and Informs)

SNMP agents can use SNMPv3 to send unsolicited notifications—Trap and Inform messages—to SNMP managers. SNMPv2c uses communities, in this case using the SNMPv2c notification community concept. SNMPv3 uses the same security levels just discussed, but as applied to SNMPv3 notifications.

To configure an SNMPv3 agent to send notifications, you add the security level and the username to the **snmp-server host** command. That configuration links to the same kinds of **snmp-server user** commands discussed earlier in this section, which in turn link to an **snmp-server group** command. Figure D-16 shows how the commands connect to each other.

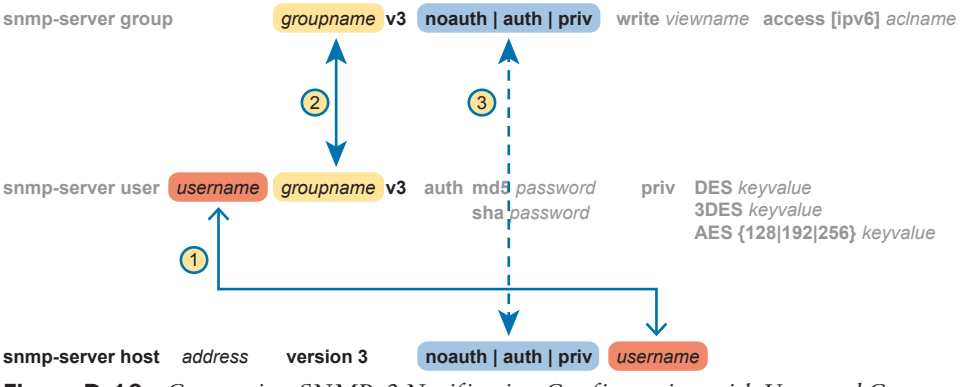


Figure D-16 *Connecting SNMPv3 Notification Configuration with User and Group*

NOTE IOS allows you to configure commands that refer to the correct username and group name, but with different security levels, with no error messages. However, communication with the NMS then fails.

Example D-29 shows a few samples of configuration notifications that use SNMPv3. The samples rely on the SNMPv3 usernames and groups as defined in Example D-26. Feel free

D

to refer back to that example, and check to make sure that each **snmp-server host** command in Example D-29 refers to the correct SNMP security level used by each linked **snmp-server group** command.

Example D-29 *Verifying SNMPv3 Configuration Settings*

```
! The group uses noauth, so the user Youdda1 has no auth nor priv keyword
snmp-server enable traps
snmp-server host 10.1.3.3 version 3 noauth Youdda1          ! Traps w/ noauth
snmp-server host 10.1.3.4 informs version 3 auth Youdda2    ! Informs w/ auth
snmp-server host 10.1.3.5 version 3 priv Youdda4            ! Traps w/ priv
```

As always, the **show snmp** command lists the counters that show how many messages flow, including the number of Trap and Inform messages sent by the SNMP agent. To verify the configuration of SNMPv3 notification to NMS hosts, use the **show snmp host** command. Example D-30 shows the results after configuring Example D-29; note that almost all the fields in Example D-30 repeat the configuration parameters from Example D-29.

Example D-30 *Verifying SNMPv3 Configuration Settings*

```
R3# show snmp host
Notification host: 10.1.3.4    udp-port: 162   type: inform
user: Youdda2 security model:  v3 auth

Notification host: 10.1.3.3    udp-port: 162   type: trap
user: Youdda1 security model:  v3 noauth

Notification host: 10.1.3.5    udp-port: 162   type: trap
user: Youdda4 security model:  v3 priv
```

Summarizing SNMPv3 Configuration

SNMPv3 configuration has many parameters to choose from in several commands. As a result, putting the commands into a configuration checklist earlier in this section did not work as well for learning, so the text instead spelled out the pieces little by little. Now that you have seen how to configure the individual pieces, this configuration checklist summarizes all the different SNMPv3 configuration options discussed in this chapter, for easier review.

Step 1. Use the **snmp-server group** *groupname* **v3** {**noauth** | **auth** | **priv**} [**write v1default**] [**access** [**ipv6**] *acl-name*] command in global configuration mode to enable the SNMP agent (if not already started), create a named SNMPv3 group of security settings, set the security level, optionally override the default write view with the same view as defaulted for use as the read MIB view (v1default), and optionally restrict incoming SNMP messages based on the optional referenced IPv4 or IPv6 ACL.

Step 2. To configure users whose referenced SNMPv3 group has a security level of **noauth**, use the **snmp-server user** *username groupname* **v3** command in global configuration mode, making sure to reference an SNMPv3 group with security level of **noauth** configured.

Step 3. To configure users whose referenced SNMPv3 group use the security level of **auth**:

A. Use the **snmp-server user** *username groupname* **v3 auth md5** *password* command in global configuration mode to configure the user and authentication password, and to choose to use MD5 as the authentication hash algorithm.

B. Alternatively, use the **snmp-server user** *username groupname* **v3 auth sha** *password* command in global configuration mode to configure the user and authentication password, and to choose to use SHA as the authentication hash algorithm.

Step 4. To configure users that use the security level of **priv**, you will add parameters to the end of the **snmp-server user** command syntax as configured in step 3, as follows:

A. Add the **priv des** *encryption-key* parameters in global configuration mode to the end of the **snmp-server user** command, to enable the use of DES as the encryption algorithm and to set the encryption key.

B. Add the **priv 3des** *encryption-key* parameters in global configuration mode to the end of the **snmp-server user** command, to enable the use of triple DES (3DES) as the encryption algorithm and to set the encryption key.

C. Add the **priv aes** {**128** | **192** | **256**} *encryption-key* parameters in global configuration mode to the end of the **snmp-server user** command, to enable the use of AES as the encryption algorithm, to set the length of the encryption key in bits, and to set the seed for the encryption key.

Step 5. Enable the SNMP agent to send notification messages (Traps and/or Informs) to an NMS as follows:

A. Use the **snmp-server host** {*hostname* | *ip-address*} [**informs** | **traps**] **version 3** {**noauth** | **auth** | **priv**} *username* command in global configuration mode to configure the SNMP agent to send SNMPv3 Traps to the listed host, using the listed username. Use this command once for each host to which this device should send Traps. Include the **informs** keyword to send Informs; the **traps** keyword is the default setting. Use the same security level setting as the link SNMPv3 group.

B. Use the **snmp-server enable traps** command in global configuration mode to enable the sending of all supported notifications to all hosts defined in **snmp-server host** commands.

Note that if you review this checklist and get lost, make sure to review and study this section again. SNMPv3 configuration uses a lot of different parameters on three different commands, so it is easy to get lost. The checklist is best used for review once you have a good understanding of the commands.

NOTE The content under the heading "Analyzing LAN Physical Standard Choices" was most recently published for the 100-105 Exam in 2016, in Chapter 10 of the *Cisco CCENT/CCNA ICND1 100-105 Official Cert Guide*.

Analyzing LAN Physical Standard Choices

When you look at the design of a network designed by someone else, you can look at all the different types of cabling used, the different types of switch ports, and the Ethernet standards used in each case. Then ask yourself: Why did they choose a particular type of Ethernet link for each link in the network? Asking that question, and investigating the answer, starts to reveal much about building the physical campus LAN.

The IEEE has done an amazing job developing Ethernet standards that give network designers many options. Two themes in particular have helped Ethernet grow over the long term:

- The IEEE has developed many additional 802.3 standards for different types of cabling, different cable lengths, and for faster speeds.
- All the physical standards rely on the same consistent data-link details, with the same standard frame formats. That means that one Ethernet LAN can use many types of physical links to meet distance, budget, and cabling needs.

For example, think about the access layer of the generic design drawings, but now think about cabling and Ethernet standards. In practice, access layer switches sit in a locked wiring closet somewhere on the same floor as the end user devices. Electricians have installed unshielded twisted-pair (UTP) cabling used at the access layer, running from that wiring closet to each wall plate at each office, cubicle, or any place where an Ethernet device might need to connect to the LAN. The type and quality of the cabling installed between the wiring closet and each Ethernet outlet dictate what Ethernet standards can be supported. Certainly, whoever designed the LAN at the time the cabling was installed thought about what type of cabling was needed to support the types of Ethernet physical standards that were going to be used in that LAN.

Ethernet Standards

Over time, the IEEE has continued to develop and release new Ethernet standards, for new faster speeds and to support new and different cabling types and cable lengths. Figure D-17 shows some insight into Ethernet speed improvements over the years. The early standards up through the early 1990s ran at 10 Mbps, with steadily improving cabling and topologies. Then, with the introduction of Fast Ethernet (100 Mbps) in 1995, the IEEE began ramping up the speeds steadily over the next few decades, continuing even until today.

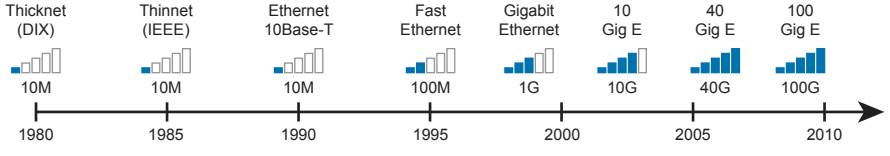


Figure D-17 *Ethernet Standards Timeline*

NOTE Often, the IEEE first introduces support for the next higher speed using some forms of fiber optic cabling, and later, sometimes many years later, the IEEE completes the work to develop standards to support the same speed on UTP cabling. Figure D-17 shows the earliest standards for each speed, no matter what cabling.

When the IEEE introduces support for a new type of cabling, or a faster speed, they create a new standard as part of 802.3. These new standards have a few letters behind the name. So, when speaking of the standards, sometimes you might refer to the standard name (with letters). For instance, the IEEE standardized Gigabit Ethernet support using inexpensive UTP cabling in standard 802.3ab. However, more often, engineers refer to that same standard as 1000BASE-T or simply Gigabit Ethernet. Table D-6 lists some of the IEEE 802.3 physical layer standards and related names for perspective.

Table D-6 IEEE Physical Layer Standards

Original IEEE Standard	Shorthand Name	Informal Names	Speed	Typical Cabling
802.3i	10BASE-T	Ethernet	10 Mbps	UTP
802.3u	100BASE-T	Fast Ethernet	100 Mbps	UTP
802.3z	1000BASE-X	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	Fiber
802.3ab	1000BASE-T	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	UTP
802.3ae	10GBASE-X	10 GigE	10 Gbps	Fiber
802.3an	10GBASE-T	10 GigE	10 Gbps	UTP
802.3ba	40GBASE-X	40 GigE	40 Gbps	Fiber
802.3ba	100GBASE-X	100 GigE	100 Gbps	Fiber

Choosing the Right Ethernet Standard for Each Link

When designing an Ethernet LAN, you can and should think about the topology, with an access layer, a distribution layer, and possibly a core layer. But thinking about the topology does not tell you which specific standards to follow for each link. Ultimately, you need to pick which Ethernet standard to use for each link, based on the following kinds of facts about each physical standard:

- The speed
- The maximum distance allowed between devices when using that standard/cabling
- The cost of the cabling and switch hardware
- The availability of that type of cabling already installed at your facilities

Consider the three most common types of Ethernet today (10BASE-T, 100BASE-T, and 1000BASE-T). They all have the same 100-meter UTP cable length restriction. They all use UTP cabling. However, not all UTP cabling meets the same quality standard, and as it turns out, the faster the Ethernet standard, the higher the required cable quality category needed to support that standard. As a result, some buildings might have better cabling that supports speeds up through Gigabit Ethernet, whereas some buildings may support only Fast Ethernet.

The Telecommunications Industry Association (TIA; tiaonline.org) defines Ethernet cabling quality standards. Each Ethernet UTP standard lists a TIA cabling quality (called a *category*) as the minimum category that the standard supports. For example, 10BASE-T allows for Category 3 (CAT3) cabling or better. 100BASE-T requires higher-quality CAT5 cabling, and 1000BASE-T requires even higher-quality CAT5e cabling. (The TIA standards follow a general "higher number is better cabling" in their numbering.) For instance, if an older facility had only CAT5 cabling installed between the wiring closets and each cubicle, the engineers

would have to consider upgrading the cabling to fully support Gigabit Ethernet. Table D-7 lists the more common types of Ethernet and their cable types and length limitations.

Table D-7 Ethernet Types, Media, and Segment Lengths (Per IEEE)

Ethernet Type	Media	Maximum Segment Length
10BASE-T	TIA CAT3 or better, 2 pairs	100 m (328 feet)
100BASE-T	TIA CAT5 UTP or better, 2 pairs	100 m (328 feet)
1000BASE-T	TIA CAT5e UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T	TIA CAT6a UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T[1]	TIA CAT6 UTP or better, 4 pairs	38–55 m (127–180 feet)
1000BASE-SX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	9-micron single-mode fiber	5 km (3.1 miles)

[1] The option for 10GBASE-T with slightly less quality CAT6 cabling, but at shorter distances, is an attempt to support 10Gig Ethernet for some installations with CAT6 installed cabling.

Ethernet defines standards for using fiber optic cables as well. Fiber optic cables include ultrathin strands of glass through which light can pass. To send bits, the switches can alternate between sending brighter and dimmer light to encode 0s and 1s on the cable.

Generally comparing optical cabling versus UTP cabling Ethernet standards, two obvious points stand out. Optical standards allow much longer cabling, while generally costing more for the cable and the switch hardware components. Optical cables experience much less interference from outside sources compared to copper cables, which allows for longer distances.

When considering optical Ethernet links, many standards exist, but with two general categories. Comparing the two, the cheaper options generally support distances into the hundreds of meters, using less expensive light-emitting diodes (LED) to transmit data. Other optical standards support much longer distances into multiple kilometers, using more expensive cabling and using lasers to transmit the data. The trade-off is basic: For a given link, how long does the cable need to run, what standards support that distance, and which is the least expensive to meet that need?

In reality, most engineers remember only the general facts from tables like Table 10-3: 100 meters for UTP, about 500 meters for multimode fiber, and about 5000 meters for some single mode fiber Ethernet standards. When it is time to get serious about designing the details of each link, the engineer must get into the details, calculating the length of each cable based on its path through the building, and so on.

NOTE The content under the heading "Metro Ethernet" was most recently published for the 200-105 Exam in 2016, in Chapter 14 of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

Metro Ethernet

This section discusses virtual circuits in Ethernet WANs.

Ethernet Virtual Circuit Bandwidth Profiles

Before leaving MetroE to move on to MPLS, it helps to consider some ideas about data usage over the WAN links and a whole topic area related to EVC Bandwidth Profiles (BWP).

First, ignoring MetroE for a moment, anyone who has shopped for mobile phone data plans in the 2010s has already thought about data usage with carrier networks. With mobile phones, many carriers offer some kind of tiered pricing: the more data you want to send and receive, the more money you spend per month. Why do they charge more based on usage? The SP spends a lot of capital and a lot of ongoing operational expense to build and operate its network. It seems fair to charge those who use less of the network a little less money, and those who use more a little more money. Simple enough.

Most private WAN services use the same kind of usage-based pricing, and this last MetroE topic discusses some of the terminology and concepts.

The first big idea is this: The access links transmit bits at a set predefined speed based on Ethernet standards. Each Ethernet access link on a MetroE WAN uses a specific Ethernet standard that runs at a specific speed. Those speeds are 10 Mbps, 100 Mbps, 1000 Mbps (that is, 1 Gbps), 10 Gbps, and so on. And while the IEEE has begun adding some new speeds for Ethernet standards, speeds that are not a multiple of 10 versus the next slower speed, the point is this: If a site's MetroE access link is using an Ethernet standard that is a 100-Mbps standard, then the bits are transmitted at 100 Mbps.

At the same time, the MetroE SP wants to be able to charge customers based on usage, and to be a little more flexible than pricing based on the speed of the access links. These final few pages of the MetroE topics in this chapter show how a MetroE SP can charge for speeds other than the access link speeds.

Charging for the Data (Bandwidth) Used

Think through this scenario. A potential customer looks at a MetroE provider's pricing. This customer wants an E-Line service between two sites only. They know that they need at least 100 Mbps of capacity (that is, bandwidth) between the sites. But because the service has the word "Ethernet" in it, the potential customer thinks the service is either 10 Mbps, 100 Mbps, 1 Gbps, and so on. So they look up pricing for an E-Line service at those prices, and think:

- **100 Mbps:** Reasonably good price, but we need more capacity
- **1000 Mbps:** More than we want to spend, it's enough capacity, but probably too much

As it turns out, what this customer really wants is 200 Mbps between the two sites. However, there is no Ethernet standard that runs at 200 Mbps, so there is no way to use access links that run at 200 Mbps. But there is a solution: an E-Line service, with a Bandwidth Profile that defines a 200-Mbps committed information rate (CIR) over the point-to-point EVC between the customer's two routers. Figure D-18 shows the ideas and terms.

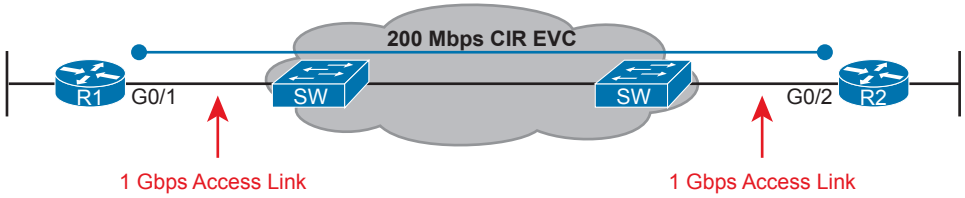


Figure D-18 *Example: 200-Mbps CIR Supported by 1-Gbps Access Links*

The big ideas are simple, although the methods to control the data are new. The SP, per the contract with the customer, agrees to not only forward Ethernet frames between the two E-Line sites, but commits to a CIR of 200 Mbps. That is, the carrier commits to pass 200 Mbps worth of Ethernet frames over time.

When a customer asks for a new E-Line with a 200-Mbps CIR, they could send lots more data than 200 Mbps. Remember, the literal transmission rate would be 1 Gbps in this example, because the access links are 1-Gbps links. But over time, if all the customers that asked for a 200-Mbps CIR E-Line sent lots more than 200 Mbps worth of data, the SP's network could become too congested. The SP builds its network to support the traffic it has committed to send, plus some extra for expected overuse, and some extra for growth. But it is too expensive to build a network that allows customers that ask for and pay for 200 Mbps to send at 1 Gbps all the time.

Controlling Overages with Policing and Shaping

To make the idea of fast access links with a slower CIR on the EVCs work, and work well, both the SP and the customer have to cooperate. The tools are two Quality of Service (QoS) tools called policing and shaping.

Historically, in some similar WAN services (like Frame Relay), the SP would actually let you send more data than your CIR, but MetroE networks typically use policing to discard the excess. A policer can watch incoming frames and identify the frames associated with each EVC. It counts the bytes in each frame, and determines a bit rate over time. When the customer has sent more bits than the CIR, the SP discards enough of the currently arriving frames to keep the rate down to the CIR. Figure D-19 shows the location of policing in the same example shown in Figure D-18.

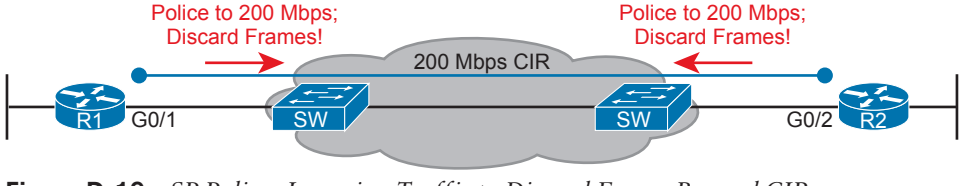


Figure D-19 *SP Polices Incoming Traffic to Discard Excess Beyond CIR*

Recapping this scenario, the customer decides to ask the MetroE SP for an E-Line. The customer's routers use a 1-Gbps access link that allows the E-Line to support a 200-Mbps CIR. To protect the SP's network, the SP now uses ingress policing to monitor the bits/second received over each end of the E-Line's point-to-point EVC. And the SP discards some incoming frames when the rate gets too high.

Having the SP discard a few frames is actually not that harmful if QoS is implemented correctly, but with MetroE, if the SP is policing as shown in Figure D-19, the customer needs

to use the other QoS tool: shaping. Shaping, as implemented on the customer routers, lets the routers slow down. Shaping tells the routers, on the MetroE access link, to send some frames, and then wait; then send more, then wait; and to do that repeatedly. Shaping can be configured for that same rate as the CIR (200 Mbps in this case), so that the SP does not have to discard any traffic.

Summarizing some of these key points:

- MetroE uses the concept of an Ethernet Virtual Connection (EVC), tying a committed number of bits/second called the committed information rate (CIR) to the EVC.
- The access links need to be fast enough to handle the combined CIRs for all EVCs that cross the link.
- For each EVC, the SP commits to forward the bits/second defined as the CIR for that EVC.
- To protect its network from being overrun with too much traffic, the SP can use policing, monitoring the incoming traffic rate on each EVC and discarding traffic that goes beyond the CIR.
- To prevent too much of its traffic from being discarded by the SP, the customer slows down its rate of sending over the EVC to match that same CIR, using shaping on the customer router.

NOTE The content under the heading "MPLS VPNs" was most recently published for the 200-105 Exam in 2016, in Chapter 14 of the *Cisco CCNA ICND2 200-105 Official Cert Guide*.

MPLS VPNs

This section discusses an OSPF design issue that exists when using MPLS VPNs.

OSPF Area Design with MPLS VPN

Now that you know the basics about what happens with routing protocols at the edge of an MPLS network, take a step back and ponder OSPF area design. For all the other WAN services discussed in the book, the WAN service is just one more data link, so the WAN sits inside one area. With MPLS, the MPLS service acts like a bunch of routers. If you use OSPF as the PE-CE routing protocol, some choices must be made about OSPF areas, and about which WAN links are in which area, and where the backbone area can and should be.

MPLS allows for a couple of variations on OSPF area design, but they all use an idea that was added to OSPF for MPLS VPNs, an idea that has come to be known informally as the OSPF *super backbone*. The idea is an elegant solution that meets OSPF needs and the requirement that the MPLS PEs, when using OSPF, must be in some OSPF area:

- The MPLS PEs form a backbone area by the name of a super backbone.
- Each PE-CE link can be any area—a non-backbone area or the backbone area.

Although the super backbone supports some functions and logic beyond the scope of this book, for the purposes of getting a basic understanding of OSPF's use with MPLS, you can think of the super backbone as simply the majority of an enterprise's OSPF backbone area,

but with the option to make the backbone area larger. The CE routers at a customer site may not be part of the backbone area, or may be, at the choice of the customer network engineers.

For example, for a nice clean design, each of the four customer sites in Figure D-20 uses a different area. The PE-CE links are part of those individual areas. The OSPF backbone area still exists, and each area connects to the backbone area, but the backbone exists in the MPLS PE routers only.

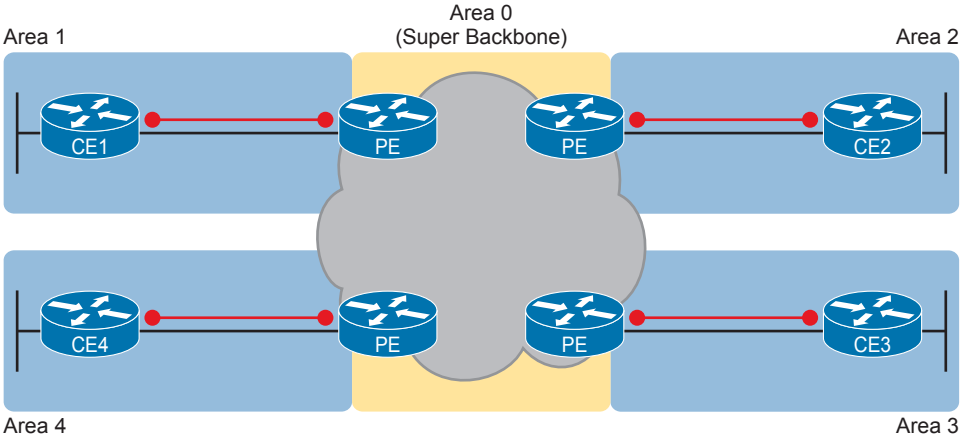


Figure D-20 *MPLS Design with (Super Backbone) Area 0, Non-Backbone Area for Each Site*

The area design in Figure D-20 provides a clean OSPF area design. However, if migrating from some other type of WAN service, with an existing OSPF design, the network engineers may prefer to keep parts of an existing OSPF design, which means some sites may still need to include the backbone area. In fact, multiple WAN sites can be configured to be in the backbone area, and still function correctly. Figure D-21 shows one such example.

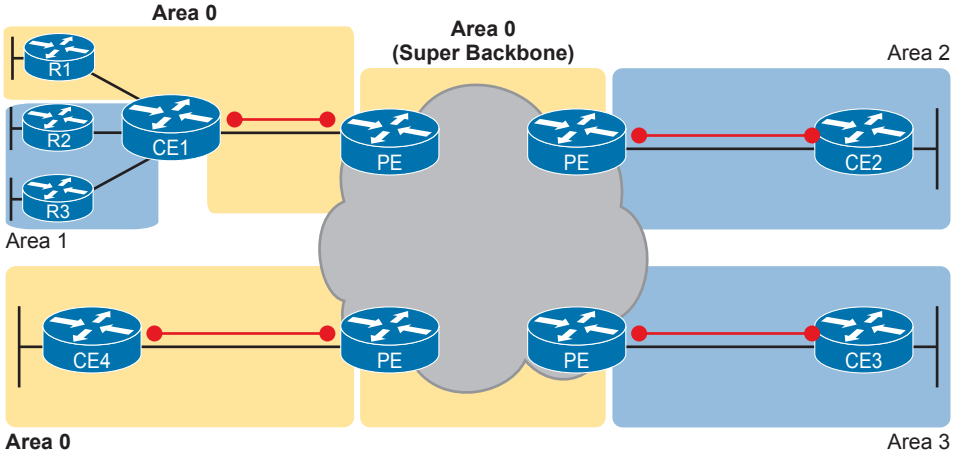


Figure D-21 *Using Area 0 on CE-PE Link, or for Entire Site*

In effect, the super backbone combines with the two other parts of the network configured as area 0 for one contiguous backbone area. Notice on the left side of Figure D-21 the two sites with area 0 noted. Normally, if both customer sites implement area 0, but there were links from some other area between them, the design would break OSPF design rules. However, the OSPF backbone (area 0) links on the left, plus the OSPF super backbone area 0 created by MPLS, act together in regard to OSPF design.

Next, focus on the site at the upper left. That site represents what might have existed before migrating to an MPLS design, with Router R1's links in area 0, and the links connected to Routers R2 and R3 in area 1. The enterprise network engineer may have decided to leave the OSPF area design alone when connecting to the MPLS network. To support those backbone area links off Router R1, the engineer put the CE1-PE1 link into area 0. As a result, the combined customer area 0 instances and the super backbone area 0 creates one contiguous backbone area.

APPENDIX E

Practice for Chapter 2: Basic IPv4 Access Control Lists

Practice Problems

This appendix includes two sets of practice problems. The first question set lists requirements for a single-line access control list (ACL), with your task being to create a standard numbered ACL that meets the requirements. The second question set shows an existing **access-list** command, with your job being to determine the range of IP addresses matched by the ACL.

Note that you can find additional practice on the author's blog, which is linked from the author's website, www.certskills.com.

Practice Building access-list Commands

Table E-1 lists the criteria for several practice problems. Your job: Create a one-line standard ACL that matches the packets. The answers are listed later in this appendix.

Table E-1 Building One-Line Standard ACLs: Practice

Problem	Criteria
1	Packets from 10.1.1.1
2	Packets from hosts with 10.1.1 as the first 3 octets
3	Packets from hosts with 10.1 as the first 2 octets
4	Packets from any host
5	Packets from subnet 192.168.3.128/29
6	Packets from subnet 192.168.3.192/28
7	Packets from subnet 192.168.3.64/27
8	Packets from subnet 172.20.192.192/26
9	Packets from subnet 172.20.200.0/22
10	Packets from subnet 172.20.203.0/25
11	Packet from subnet 192.168.99.0/30
12	Packet from subnet 192.168.99.0/28
13	Packet from subnet 172.28.28.0/23
14	Packet from subnet 172.28.28.0/22
15	Packet from subnet 172.28.28.0/24

Reverse Engineering from ACL to Address Range

For this second question set, look at the existing **access-list** commands in Table E-2. In each case, make a notation about the exact IP address, or range of IP addresses, matched by the command.

Table E-2 Finding IP Addresses/Ranges Matching by Existing ACLs

Problem	Commands for Which to Predict the Source Address Range
1	access-list 1 permit 192.168.4.5
2	access-list 2 permit 192.168.4.128 0.0.0.3
3	access-list 3 permit 192.168.4.128 0.0.0.127
4	access-list 4 permit 172.25.96.0 0.0.0.255
5	access-list 5 permit 192.168.4.128 0.0.0.31
6	access-list 6 permit 192.168.4.128 0.0.0.7
7	access-list 7 permit 172.25.96.0 0.0.7.255
8	access-list 8 permit 172.25.96.0 0.0.0.63
9	access-list 9 permit 10.10.16.0 0.0.7.255
10	access-list 10 permit 10.10.16.0 0.0.0.127
11	access-list 11 permit 192.168.17.112 0.0.0.7
12	access-list 12 permit 192.168.17.112 0.0.0.15
13	access-list 13 permit 172.19.200.0 0.0.0.63
14	access-list 14 permit 172.19.200.0 0.0.1.255
15	access-list 15 permit 10.1.0.0 0.0.255.255

NOTE You can only rely on the method of adding these numbers together (as shown in Chapter 2, "Basic IPv4 Access Control Lists") if you know that the **access-list** command comes from the router and specifically is not what someone simply wrote on a piece of paper. In this case, you can assume that the statements in Table E-2 came from a router.

Answers to Earlier Practice Problems

This section contains the answers to the two sets of practice problems.

Answers: Practice Building access-list Commands

Table E-3 lists the answers to the problems listed in Table E-1.

Table E-3 Building One-Line Standard ACLs: Answers

Problem	Answer
1	access-list 1 permit 10.1.1.1
2	access-list 2 permit 10.1.1.0 0.0.0.255
3	access-list 3 permit 10.1.0.0 0.0.255.255
4	access-list 4 permit any
5	access-list 5 permit 192.168.3.128 0.0.0.7
6	access-list 6 permit 192.168.3.192 0.0.0.15
7	access-list 7 permit 192.168.3.64 0.0.0.31
8	access-list 8 permit 172.20.192.192 0.0.0.63
9	access-list 9 permit 172.20.200.0 0.0.3.255
10	access-list 10 permit 172.20.203.0 0.0.0.127
11	access-list 11 permit 192.168.99.0 0.0.0.3
12	access-list 12 permit 192.168.99.0 0.0.0.15
13	access-list 13 permit 172.28.28.0 0.0.1.255
14	access-list 14 permit 172.28.28.0 0.0.3.255
15	access-list 15 permit 172.28.28.0 0.0.0.255

Answers: Reverse Engineering from ACL to Address Range

Table E-4 lists the answers to the problems listed in Table E-2.

Table E-4 Address Ranges for Problems in Table E-2: Answers

Problem	Address Range
1	One address: 192.168.4.5
2	192.168.4.128 – 192.168.4.131
3	192.168.4.128 – 192.168.4.255
4	172.25.96.0 – 172.25.96.255
5	192.168.4.128 – 192.168.4.159
6	192.168.4.128 – 192.168.4.135
7	172.25.96.0 – 172.25.103.255
8	172.25.96.0 – 172.25.96.63
9	10.10.16.0 – 10.10.23.255
10	10.10.16.0 – 10.10.16.127
11	192.168.17.112 – 192.168.17.119

Problem	Address Range
12	192.168.17.112 – 192.168.17.127
13	172.19.200.0 – 172.19.200.63
14	172.19.200.0 – 172.19.201.255
15	10.1.0.0 – 10.1.255.255

APPENDIX F

Previous Edition ICND1 Chapter 35: Managing IOS Files

NOTE This appendix contains an entire chapter that was published in one of the past editions of a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more; however, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 35 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

Cisco has a wide and complex product catalog. The CCENT and CCNA R&S exams focus on two major branches of the product line: routers that run Cisco IOS software as the operating system (OS) and Catalyst LAN switches that also run IOS. While the IOS for each type of device has some differences, just because routers and switches perform different functions, IOS that runs on these switches and routers has many similarities. Within the exams, Cisco attempts to be generic in that the exam does not ask you to make distinctions between different models of routers and switches.

This chapter looks at some topics that again apply to IOS that runs in both Cisco routers and Cisco Catalyst switches. In particular, this chapter looks at the IOS itself: the file systems where the IOS stores files, how to upgrade IOS, and what happens when you reboot the router or switch to upgrade the IOS. This chapter also looks at how to manage configuration files beyond simply keeping them inside router or switch memory in the startup-config file. This chapter also includes a brief discussion about how to recover if you lose the password for a router or switch.

Note that this chapter focuses on features on Cisco routers. However, many of the same features work either exactly the same, or in a very similar way, on Cisco Catalyst switches.

Foundation Topics

Managing Cisco IOS Images and Upgrades

IOS exists as a file—a single file—that the router then loads into RAM to use as its operating system (OS). This first major section of the chapter works through the story of how to upgrade to a new version of IOS.

This first section has one primary purpose but many secondary purposes. Primarily, this section shows how to upgrade IOS on a router. As a secondary goal, this section works through

a variety of small IOS features that engineers use during that upgrade process—features not covered in any detail until this point in the book. This section explains these topics, in order:

1. The IOS File System
2. Upgrading IOS Images
3. The Cisco IOS Boot Sequence

The IOS File System

Every OS creates file systems to store files. A computer needs some type of permanent storage, but it needs more than just a place to store bytes. The OS organizes the storage into a file system, which includes directories, structure, and filenames, with the associated rules. By using a file system, the OS can keep data organized so the user and the applications can find the data later.

Every OS defines its own file system conventions. Windows OSs, for instance, use a left-leaning slash (\) in directory structures, like \Desktop\Applications. Linux and OS X use a right-leaning slash, for example, /Desktop. Each OS refers to physical disks slightly differently as well, and IOS is no different.

As for the physical storage, Cisco routers typically use flash memory, with no hard disk drive. Flash memory is rewriteable, permanent storage, which is ideal for storing files that need to be retained when the router loses power. Cisco purposefully uses flash memory rather than hard disk drives in its products because there are no moving parts in flash memory, so there is a smaller chance of failure as compared with disk drives. Some routers have flash memory on the motherboard. Others have flash memory slots that allow easy removal and replacement of the flash card, but with the intent that the card remain in the device most of the time. Also, many devices have USB slots that support USB flash drives.

For each physical memory device in the router, IOS creates a simple IOS file system (IFS) and gives that device a name. Example F-1 lists the surprisingly long list of IOS file systems. Note that the entries of type *disk* and *usbflash* are the physical storage devices in that router. In this case, the router has one of two of the 2901's compact flash slots populated with a 256 MB flash card, and one of the two USB flash slots populated with an 8 GB USB flash drive. Look at the size column and prefixes column in the output to find these devices, based on their types as *disk* and *usbflash*.

Example F-1 *Cisco IOS File Systems on a Router*

```
R2# show file systems
File Systems:

       Size(b)       Free(b)      Type  Flags  Prefixes
             -             -    opaque     rw   archive:
             -             -    opaque     rw   system:
             -             -    opaque     rw   tmpsys:
             -             -    opaque     rw   null:
             -             -   network     rw   tftp:
*    256487424      49238016      disk     rw   flash0: flash:#
```

```
               -               -       disk      rw   flash1:
          262136          253220      nvram      rw   nvram:
               -               -     opaque      wo   syslog:
               -               -     opaque      rw   xmodem:
               -               -     opaque      rw   ymodem:
               -               -    network      rw   rcp:
               -               -    network      rw   pram:
               -               -    network      rw   http:
               -               -    network      rw   ftp:
               -               -    network      rw   scp:
               -               -     opaque      ro   tar:
               -               -    network      rw   https:
               -               -     opaque      ro   cns:
      7794737152      7483719680  usbflash      rw   usbflash0:
74503236 bytes copied in 187.876 secs (396555 bytes/sec)
```

The example lists 20 different IOS file systems in this case, but the router does not have 20 different physical storage devices. Instead, IOS uses these file systems for other purposes as well, with these types:

- **Opaque:** To represent logical internal file systems for the convenience of internal functions and commands
- **Network:** To represent external file systems found on different types of servers for the convenience of reference in different IOS commands
- **Disk:** For flash
- **Usbflash:** For USB flash
- **NVRAM:** A special type for NVRAM memory, the default location of the startup-config file

Many IOS commands refer to files in an IFS, but only some commands refer directly to the files by their formal names. The formal names use the prefix as seen in the far right column of Example F-1. For instance, the command **more flash0:/wotemp/fred** would display the contents of file *fred* in directory */wotemp* in the first flash memory slot in the router. (The **more** command itself displays the contents of a file.) However, many commands use a keyword that indirectly refers to a formal filename, to reduce typing. For example:

- **show running-config** command: Refers to file system:running-config
- **show startup-config** command: Refers to file nvram:startup-config
- **show flash** command: Refers to default flash IFS (usually flash0:)

Upgrading IOS Images

One of the first steps to upgrade a router's IOS to a new version is to obtain the new IOS image and put it in the right location. Typically, Cisco routers have their IOS in one of the local physical file systems, most often in permanent flash. The only requirement is that the IOS be in some reachable file system—even if the file sits on an external server and the device loads the OS over the network. However, the best practice is to store each device's IOS file in flash that will remain with the device permanently.

Figure F-1 illustrates the process to upgrade an IOS image into flash memory, using the following steps:

Step 1. Obtain the IOS image from Cisco, usually by downloading the IOS image from cisco.com using HTTP or FTP.

Step 2. Place the IOS image someplace that the router can reach. Locations include TFTP or FTP servers in the network or a USB flash drive that is then inserted into the router.

Step 3. Issue the **copy** command from the router, copying the file into the flash memory that usually remains with the router on a permanent basis. (Routers usually cannot boot from the IOS image in a USB flash drive.)

Figure F-1 *Copying IOS Image as Part of the Cisco IOS Software Upgrade Process*

Copying a New IOS Image to a Local IOS File System Using TFTP

Example F-2 provides an example of Step 3 from the figure, copying the IOS image into flash memory. In this case, Router R2, a 2901, copies an IOS image from a TFTP server at IP address 2.2.2.1.

Example F-2 **copy tftp flash** *Command Copies the IOS Image to Flash Memory*

```
R2# copy tftp flash
Address or name of remote host []? 2.2.2.1
Source filename []? c2900-universalk9-mz.SPA.152-4.M1.bin
Destination filename [c2900-universalk9-mz.SPA.152-4.M1.bin ]?
Accessing tftp://2.2.2.1/c2900-universalk9-mz.SPA.152-4.M1.bin ...
Loading c2900-universalk9-mz.SPA.152-4.M1.bin from 2.2.2.1 (via GigabitEthernet0/1):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 97794040 bytes]


97794040 bytes copied in 187.876 secs (396555 bytes/sec)
R2#
```

The **copy** command does a simple task—copy a file—but the command also has several small items to check. It needs a few pieces of information from the user, so the command prompts the user for that information by showing the user some text and waiting for the user's input. The bold items in the example show the user's input. The router then has to check to make sure the copy will work. The command works through these kinds of questions:

1. What is the IP address or host name of the TFTP server?
2. What is the name of the file?

3. Ask the server to learn the size of the file, and then check the local router's flash to ask whether enough space is available for this file in flash memory.
4. Does the server actually have a file by that name?
5. Do you want the router to erase any old files in flash?

The router prompts you for answers to some of these questions, as necessary. For each question, you should either type an answer or press **Enter** if the default answer (shown in square brackets at the end of the question) is acceptable. Afterward, the router erases flash memory if directed, copies the file, and then verifies that the checksum for the file shows that no errors occurred in transmission.

NOTE Most people use the IOS filenames that Cisco supplies because these names embed information about the IOS image, like the version. Also, if you want to use the same destination filename as the source, avoid the mistake of typing "y" or "yes" to confirm the selection; instead, you would be setting the destination filename to "y" or "yes." Simply press **Enter** to confirm the selection listed in brackets.

You can view the contents of the flash file system to see the IOS file that was just copied by using a couple of commands. The **show flash** command shows the files in the default flash file system (flash0:), as seen at the top of Example F-3. Below it, the more general **dir flash0:** command lists the contents of that same file system, with similar information. (You can use the **dir** command to display the contents of any local IFS.)

Example F-3 *Command Copies the IOS Image to Flash Memory*

```
R4# show flash
-#- --length-- -----date/time------ path
1     104193476 Jul 21 2015 13:38:06 +00:00 c2900-universalk9-mz.SPA.154-3.M3.bin
3       3000320 Jul 10 2012 00:05:44 +00:00 cpexpress.tar
4          1038 Jul 10 2012 00:05:52 +00:00 home.shtml
5        122880 Jul 10 2012 00:06:02 +00:00 home.tar
6       1697952 Jul 10 2012 00:06:16 +00:00 securedesktop-ios-3.1.1.45-k9.pkg
7        415956 Jul 10 2012 00:06:28 +00:00 sslclient-win-1.1.4.176.pkg
8          1153 Aug 16 2012 18:20:56 +00:00 wo-lic-1
9      97794040 Oct 10 2014 21:06:38 +00:00 c2900-universalk9-mz.SPA.152-4.M1.bin

49238016 bytes available (207249408 bytes used)

R4# dir flash0:
Directory of flash0:/

    1  -rw-    104193476  Jul 21 2015 13:38:06 +00:00  c2900-universalk9-mz.SPA.154-3.
                                                       M3.bin
    3  -rw-      3000320  Jul 10 2012 00:05:44 +00:00  cpexpress.tar
    4  -rw-         1038  Jul 10 2012 00:05:52 +00:00  home.shtml
    5  -rw-       122880  Jul 10 2012 00:06:02 +00:00  home.tar
    6  -rw-      1697952  Jul 10 2012 00:06:16 +00:00  securedesktop-ios-3.1.1.45-k9.
                                                       pkg
    7  -rw-       415956  Jul 10 2012 00:06:28 +00:00  sslclient-win-1.1.4.176.pkg
    8  -rw-         1153  Aug 16 2012 18:20:56 +00:00  wo-lic-1
```

```
    9  -rw-     97794040  Oct 10 2014 21:06:38 +00:00  c2900-universalk9-mz.SPA.152-4.
                                                       M1.bin

256487424 bytes total (49238016 bytes free)
```

Pay close attention to the memory usage per file and for the IFS as shown in the example. The output lists the size in bytes for each file. Note that the IOS file is about 104 MB. Note that the size of the IOS file matches the size shown earlier in the TFTP transfer in Example F-2. The end of each of the commands then lists the amount of space available for new files to be added to flash (one lists it as "bytes available"; the other as "bytes free"). However, that same ending line of each command shows slightly different information about usage: **show flash** lists the bytes used, whereas the **dir** command lists the total bytes (bytes used plus bytes free). Play around with the numbers in this example to make sure you know which command lists which particular total.

Verifying IOS Code Integrity with MD5

You download the IOS from Cisco, copy it to your router, and run it. Is it really the code from Cisco? Or did some nefarious attacker somehow get you to download a fake IOS that has a virus?

Cisco provides a means to check the integrity of the IOS file to prevent this type of problem. Figure F-2 shows the basic mechanics of the process. First, when Cisco builds a new IOS image, it calculates and publishes an MD5 hash value for that specific IOS file. That is, Cisco uses as input the IOS file itself, runs the MD5 math algorithm against that file, producing a hex code. Cisco places that code at the download site for all to see. Then, you run that same MD5 math on your router against the IOS file on the router, using the IOS **verify** command. That command will list the MD5 hash as recalculated on your router. If both MD5 hashes are equal, the file has not changed.

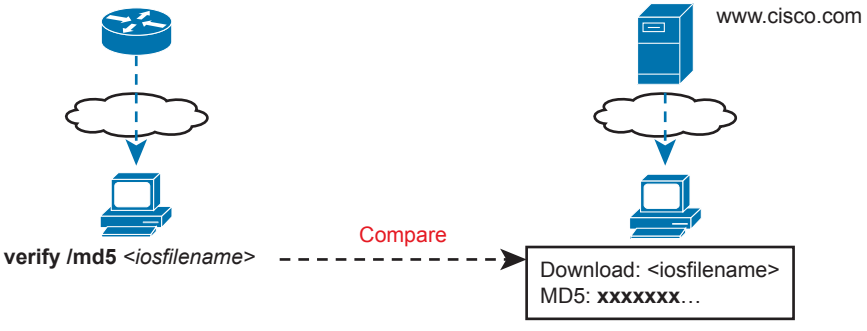


Figure F-2 *MD5 Verification of IOS Images—Concepts*

The **verify /md5** command generates the MD5 hash on your router, as shown in Example F-4. Note that you can include the hash value computed by Cisco as the last parameter (as shown in the example), or leave it off. If you include it, IOS will tell you if the locally computed value matches what you copied into the command. If you leave it out, the **verify** command lists the locally computed MD5 hash, and you have to do the picky character-by-character check of the values yourself.

Key Topic

Example F-4 *Verifying Flash Memory Contents with the* **show flash** *Command*

```
R2# verify /md5 flash0:c2900-universalk9-mz.SPA.154-3.M3.bin a79e325e6c498b70829d4d
  b0afba5041
 .....................................................................................
 .....................................................................................
….MD5 of flash0:c2900-universalk9-mz.SPA.154-3.M3.bin Done!
Verified (flash0:c2900-universalk9-mz.SPA.154-3.M3.bin) = a79e325e6c498b70829d4d
  b0afba5041
```

Copying Images with FTP

The networking world has many options for file transfer, several of which IOS supports for the transfer of files into and out of the IOS file systems that reside on the router. TFTP and FTP have been supported for the longest time, with more recent support added for protocols like SCP. Table F-1 lists some of the names of file transfer protocols that you might come across when working with routers.

Table F-1 Common Methods to Copy Files Outside a Router

Method	Method (Full Name)	Router's Role	Encrypted?
TFTP	Trivial File Transfer Protocol	Client	No
FTP	File Transfer Protocol	Client	No
SCP	Secure Copy Protocol	Server	Yes

To copy files with FTP, you follow the same kind of process you use with TFTP (see Example F-5). You can follow the interactive prompts after using an EXEC command like **copy ftp flash**. However, the **copy** command allows you to use a URI for the source and/or destination, which lets you put most or all of the information in the command line itself. Each URI refers to the formal name of a file in the IFS.

Example F-5 *Installing a New IOS with FTP*

```
R1# copy ftp://wendell:odom@192.168.1.170/c2900-universalk9-mz.SPA.155-2.T1.bin flash
Destination filename [c2900-universalk9-mz.SPA.155-2.T1.bin]?
Accessing ftp://192.168.1.170/c2900-universalk9-mz.SPA.155-2.T1.bin...
Loading c2900-universalk9-mz.SPA.155-2.T1.bin !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 107410736/4096 bytes]


107410736 bytes copied in 119.604 secs (898053 bytes/sec)
```

First, take a close look at the long URI in the command that begins with "ftp". The "ftp" part identifies the protocol, of course. After the //, the text references the username (wendell) and password (odom), as well as the FTP server's IP address. After the single / comes the filename on the server.

Although the command is long, it has only two parameters, with the long first parameter and the short keyword **flash** as the second parameter. The **copy** command lists the source

location as the first parameter and the destination as the second. The destination in this case, **flash**, is a keyword that refers to the default flash, typically flash0:, but it does not identify a specific filename. As a result, IOS prompts the user for a specific destination filename, with a default (in brackets) to keep the source filename. In this case, the user just pressed Enter to accept the default. To avoid being prompted at all, the command could have listed **flash:c2900-universalk9-mz.SPA.155-2.T1.bin** as that second parameter, fully defining the destination file.

Finally, with another twist, you can configure the FTP username and password on the router so that you do not have to include them in the **copy** command. For instance, the global configuration commands **ip ftp username wendell** and **ip ftp password odom** would have configured those values. Then the **copy** command would have begun with **copy ftp://192.168.1.170/...**, omitting the username:password in the command, without needing to then prompt the user for the username and password.

Copying Images with SCP

SSH Copy Protocol (SCP) provides a secure way to transfer files, but with a small twist as compared to other methods mentioned in this chapter: the router acts as the server, and you do not use the **copy** command on the router. Instead, you configure the router to act as an SCP server and then use an SCP client command or application on a desktop computer to transfer the files.

SCP uses SSH for two key parts of the work to securely transfer files: to authenticate the user and to encrypt all data transfer. SSH already does those tasks anyway, so SCP, defined after SSH was well established, simply relies on SSH to do those tasks. SCP then defines a method to transfer files.

To make SCP work on a router, the router first needs configuration to support SSH login as normal, as discussed in detail back in Chapter 8, "Configuring Basic Switch Management." Then you just need to change one command plus add another, as follows:

Key Topic

- Give the SSH user direct access to privileged mode by adding parameters to the **username** command, for example, **username fred privilege-level 15 password barney**.
- Enable the SCP server with the **ip scp server enable** global command.

NOTE While this book does not go into details about IOS privilege levels, enable mode is considered to be privilege level 15. The **username privilege 15** command means that the user would be granted enable mode access at login, without first being placed into user mode.

Then to use SCP to transfer files, the network engineer must use an SCP client on some computer that has network connectivity to the router. You can search the web for SCP clients, many of which are integrated as part of SSH clients. However, for the purpose of transferring files with Cisco devices, a command-line SCP client may actually be the best choice.

Example F-6 shows an SCP file copy with a router, using the Mac OS X built-in **scp** command. The command again copies an IOS file from the computer to the router, like the earlier examples. Note that it uses the full URI of the destination, with the username (wendell), router IP address (192.168.1.9), and IOS filename. The command then prompts the user for the password and begins transferring the file.

Example F-6 *SCP Client IOS Copy from a Mac to a Router*

```
WO-iMac:Desktop wendellodom$ scp c2900-universalk9-mz.SPA.155-2.T1.bin
wendell@192.168.1.9:flash0:c2900-universalk9-mz.SPA.155-2.T1.bin
 Password:
c2900-universalk9-mz.SPA.155-2.T1.bin                      100%  102MB 322.8KB/s
  05:25
```

Once you copy the IOS file into a local IOS file system on the router, you must **reload** the router to start using the new IOS. The next topic looks at the entire IOS boot process, including how to make a router start using the new version of IOS.

The Cisco IOS Software Boot Sequence

Cisco routers perform the same types of tasks that a typical computer performs when you power it on or reboot (reload) it. However, most end-user computers have a single instance of the OS installed, so the computer does not have to choose which OS to load. In contrast, a router can have multiple IOS images available both in flash memory and on external servers, so the router needs a process to pick which IOS image to load into RAM and use. This section examines the entire boot process, with extra emphasis on the options that impact a router's choice of what IOS image to load.

NOTE Routers can load IOS or a special-purpose OS called ROMMON. ROMMON is used for special purposes like password recovery. ROMMON can be used to send and receive IP packets to load a new IOS, but it does not route packets. A third very old special-purpose OS, called RXBOOT, is no longer included in this book because it applies only to very old router models.

When a router first powers on, it follows these four steps:

Key Topic

Step 1. The router performs a power-on self-test (POST) process to discover the hardware components and verify that all components work properly.

Step 2. The router copies a bootstrap program from ROM into RAM and runs the bootstrap program.

Step 3. The bootstrap program decides which IOS image (or the ROMMON OS) to load into RAM, and then the bootstrap program loads the OS. After loading the chosen OS image, the bootstrap program hands over control of the router hardware to the newly loaded OS.

Step 4. If the bootstrap program happened to load IOS, once IOS is running, it finds the startup-config file and loads it into RAM as the running-config.

All routers attempt all four steps each time the router is powered on or reloaded. The first two steps do not have any options to choose; either both of these steps succeed or the initialization fails. If it fails, you might need to call the Cisco Technical Assistance Center (TAC) for support. However, Steps 3 and 4 have several configurable options that tell the router what to do next, as noted in Figure F-3.

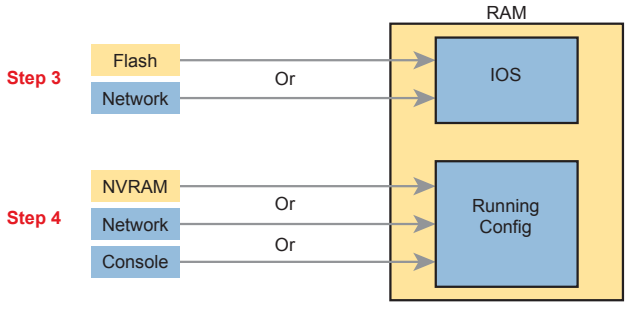


Figure F-3 *Loading IOS and Initial Configuration*

As you can see, the router has options at both Steps 3 and 4 in the figure. However, at Step 4, routers almost always load the configuration from NVRAM (the startup-config file), when it exists. There is no real advantage to storing the initial configuration anywhere else except NVRAM, so this chapter does not look further into the options of Step 4. But there are reasonable motivations for keeping IOS images in flash and on servers in the network, so the rest of this section examines Step 3 in more detail.

The Configuration Register

A router's configuration register has an impact on a router's choice of which OS to load.

Routers use a *configuration register* to find some configuration settings at boot time, before the router has loaded IOS and read the startup-config file. The 16 bits (4 hex digits) in the configuration register set a variety of different parameters. For example, the console runs at a speed of 9600 bps by default, but that console speed is based on the default settings of a couple of bits in the configuration register. By changing specific bits in the configuration register, the next time the router boots, you can change the speed of the console line.

You can set the configuration register value with the **config-register** global configuration command. Engineers set the configuration register to different values for many reasons, but the most common are to help tell the router what IOS image to load, as explained in the next few pages, and in the password recovery process. For example, the global configuration command **config-register 0x2100** sets the value to hexadecimal 2100, which causes the router to load the ROMMON OS rather than IOS the next time the router is reloaded.

Interestingly, Cisco routers automatically save the new configuration register value when you press **Enter** at the end of the **config-register** command; you do not need to use the **copy running-config startup-config** command after changing the configuration register. However, the configuration register's new value has no effect until the next time the router is reloaded.

NOTE On most Cisco routers, the default configuration register setting is hexadecimal 2102, which leaves the console speed at 9600 bps and tells the router to load an IOS image.

How a Router Chooses Which OS to Load

A router chooses the OS to load based on two factors:

- The last hex digit in the configuration register (called the *boot field*)
- Any **boot system** global configuration commands in the startup-config file

The boot field, the fourth hex digit in the configuration register, tells the router the initial instructions about what OS to try to load. The router looks at the boot field's value when the router is powered on or when reloaded. The boot field's value then tells the router how to proceed with choosing which OS to load.

NOTE Cisco represents hexadecimal values by preceding the hex digits with 0x; for example, 0xA would mean a single hex digit A.

The process to choose which OS to load on modern Cisco routers happens as follows:

Key Topic

1. If boot field = 0, use the ROMMON OS.
2. If boot field = 1, load the first IOS file found in flash memory.
3. If boot field = 2-F:
   A. Try each **boot system** command in the startup-config file, in order, until one works.
   B. If none of the **boot system** commands work, load the first IOS file found in flash memory.
4. If all other attempts fail, load ROMMON, from which you can perform further steps to recover by copying a new IOS image into flash.

NOTE The actual step numbers are not important; the list is just numbered for easier reference.

The first two steps are pretty straightforward, but Step 3 then tells the router to look to the second major method to tell the router which IOS to load: the **boot system** global configuration command. This command can be configured multiple times on one router, with each new **boot system** command being added to the end of a list of **boot system** commands. Each command can point to different files in flash memory, and filenames and IP addresses of servers, telling the router where to look for an IOS image to load. The router tries to load the IOS images in the order of the configured **boot system** commands.

Both Step 2 and Step 3B refer to a concept of the "first" IOS file, a concept that needs a little more explanation. Routers number the files stored in flash memory, with each new file usually getting a higher and higher number. When a router tries Step 2 or Step 3B from the preceding list, the router looks in flash memory, starting with file number 1, and then file number 2, and so on, until it finds the lowest numbered file that happens to be an IOS image. The router then loads that file.

Interestingly, most routers end up using Step 3B to find their IOS image. From the factory, Cisco routers do not have any **boot system** commands configured; in fact, they do not have any configuration in the startup-config file at all. Cisco loads flash memory with a single IOS when it builds and tests the router, and the configuration register value is set to 0x2102, meaning a boot field of 0x2. With all these settings, the process tries Step 3 (because boot = 2), finds no **boot system** commands (because the startup-config is empty), and then looks for the first file in flash memory at Step 3B.

NOTE Routers do not search all flash file systems for an IOS image. The details vary depending on the router model, but routers consider one flash file system to be the default IOS file system to look for IOS images.

Figure F-4 summarizes the key concepts behind how a router chooses the OS to load.

Key Topic

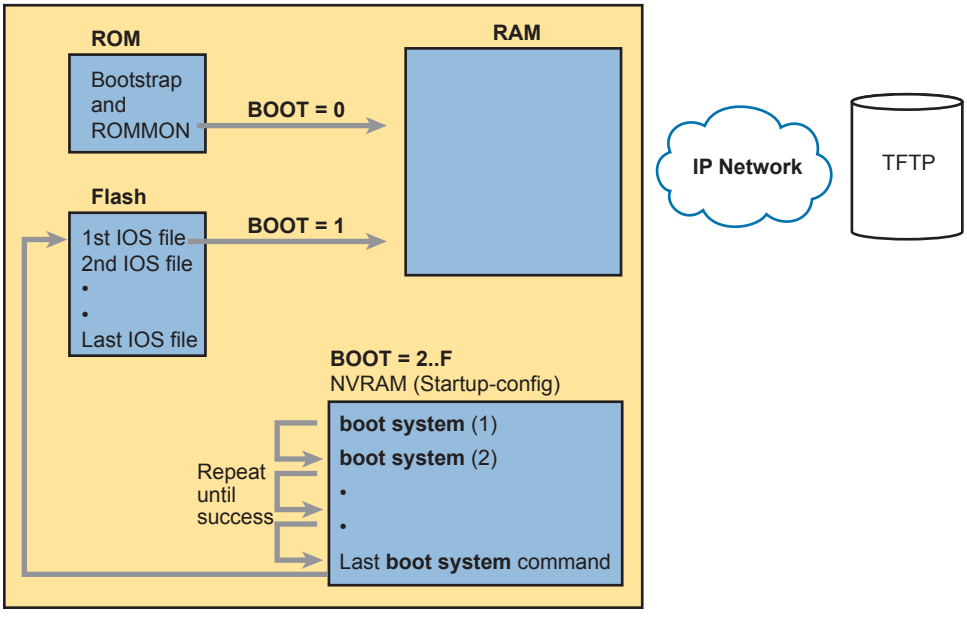


Figure F-4 *Choices for Choosing the OS at Boot Time: Modern Cisco Router*

The **boot system** commands need to refer to the exact file that the router should load. Table F-2 shows several examples of the commands.

Table F-2 Sample **boot system** Commands

Boot System Command	Result
boot system flash	The first file from system flash memory is loaded.
boot system flash *filename*	IOS with the name *filename* is loaded from system flash memory.
boot system tftp *filename* **10.1.1.1**	IOS with the name *filename* is loaded from the TFTP server at address 10.1.1.1.

Finally, remember the process of upgrading the IOS? The whole point of the **boot system** commands and boot field of the configuration register is to control which IOS loads. Once a new IOS has been copied into flash memory on the router, the upgrade process has a few more steps. Add a **boot system** command to refer to the correct new file, save the configuration, and reload the router. The router will now go through the boot sequence discussed in this section, load the new IOS image, and the IOS upgrade is complete. For instance, Example F-2 showed a router copying an IOS image into flash; that router would then also need a **boot system flash:c2900-universalk9-mz.SPA.152-4.M1.bin** command saved into the startup-config.

Verifying the IOS Image Using the show version Command

Once it is upgraded, you should verify the new IOS has loaded using the **show version** command. This command lists not only the version of software but also the source from which the router found the IOS image and the time since it loaded the IOS. As a result, the **show version** command actually identifies some key facts about the results of the previous boot process.

The **show version** command lists many other facts as well, as shown in Example F-7. The example shows output from Router R2, which has been configured with the **boot system flash:c2900-universalk9-mz.SPA.152-4.M1.bin** command and been reloaded, migrating to use the new Version 15.2(4) IOS.

To help point out some of the many important facts in this command, the example shows many highlighted items. The following list describes each of the items in the output in the same order as they are shown in the example, top to bottom:

1. The IOS version
2. The uptime (the length of time that has passed since the last reload)
3. The reason for the last reload of IOS (reload command, power off/on, software failure)
4. The time of the last loading of IOS (if the router's clock has been set)
5. The source from which the router loaded the current IOS
6. The amount of RAM memory
7. The number and types of interfaces
8. The amount of NVRAM memory
9. The amount of flash memory
10. The configuration register's current and future setting (if different)

Key Topic

Example F-7 **show version** *Command Output*

```
R2# show version
Cisco IOS Software, C2900 Software (C2900-UNIVERSALK9-M), Version 15.2(4)M1, RELEASE
  SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright  1986-2012 by Cisco Systems, Inc.
Compiled Thu 26-Jul-12 20:54 by prod_rel_team

ROM: System Bootstrap, Version 15.0(1r)M15, RELEASE SOFTWARE (fc1)

R2 uptime is 44 minutes
System returned to ROM by reload at 19:44:01 UTC Tue Feb 12 2013
System restarted at 19:45:53 UTC Tue Feb 12 2013
System image file is "flash:c2900-universalk9-mz.SPA.152-4.M1.bin"
Last reload type: Normal Reload
Last reload reason: Reload Command

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
```

```
! Rest of legal disclaimer omitted

Cisco CISCO2901/K9 (revision 1.0) with 483328K/40960K bytes of memory.
Processor board ID FTX1628837T
2 Gigabit Ethernet interfaces
4 Serial(sync/async) interfaces
1 terminal line
DRAM configuration is 64 bits wide with parity enabled.
255K bytes of non-volatile configuration memory.
3425968K bytes of USB Flash usbflash1 (Read/Write)
250880K bytes of ATA System CompactFlash 0 (Read/Write)


License Info:

License UDI:

-------------------------------------------------
Device#    PID                     SN
-------------------------------------------------
*0         CISCO2901/K9            FTX1628837T


Technology Package License Information for Module:'c2900'
-----------------------------------------------------------------
Technology      Technology-package              Technology-package
                Current         Type            Next reboot
-----------------------------------------------------------------
ipbase          ipbasek9        Permanent       ipbasek9
security        None            None            None
uc              None            None            None
data            None            None            None

Configuration register is 0x2102
```

Password Recovery

Suppose that you are sitting at your desk and you try to Secure Shell (SSH) or Telnet to a router. However, you cannot log in. Or, you can get into user mode but not into enable mode because you forgot the **enable secret** password. You want to recover, or at least reset the passwords, so you can get into the router and change the configuration. What can you do?

Cisco provides a way to reset the passwords on a router when sitting beside the router. With access to the router console and the ability to power the router off and back on, anyone can reset all the passwords on the router to new values.

The details differ from router model to router model. However, if you go to www.cisco.com and search for "password recovery," within the first few hits you should see a master password

recovery page. This page lists instructions on how to perform password recovery (actually password reset) for almost any model of Cisco product.

NOTE Cisco generally refers to the topic in this section as password recovery, but you do not actually recover and learn the password that you forgot. Instead, you change the password to a new value.

The General Ideas Behind Cisco Password Recovery/Reset

Although the details differ from model to model, all the password recovery procedures follow the same general principles. First, the end goal of the process is to make the router boot IOS while ignoring the startup-config file. Of course, this startup configuration holds all the passwords. Once the router boots while ignoring the initial configuration, the router has no passwords at all, so you can log in at the console with no password restrictions and reconfigure all the passwords.

One config-register bit holds the key: the ignore configuration bit. (The bit is the second bit in the third nibble, reading left to right.) When set to binary 1, the router will ignore the startup-config file the next time the router is loaded. To set that value, the default configuration register value of 0x2102 can be changed to 0x2142.

Unfortunately, under normal circumstances, you need to remember the enable password to reach the mode to configure the configuration register's value. When you need to do password recovery, you clearly do not know the passwords, so how can you change the configuration register? The solution is to use ROMMON mode.

ROMMON enables you to set the configuration register. ROMMON contains a small and different set of CLI commands as compared to IOS, with the commands varying from router model to router model. However, each router's ROMMON software supports some command, usually the **confreg** command, that lets you set the configuration register. For instance, the ROMMON command **confreg 0x2142** would set the correct bit to tell the router to ignore the startup-config file at reload.

So, how do you get the router to boot in ROMMON mode? Older routers require you to press the break key at the console during boot of the router. Some newer routers happen to have all removable flash memory—on those, just remove the flash (so there is no IOS available), and turn the router off and back on, and the router has no IOS to load—so it loads ROMMON. (Put the flash back in once ROMMON loads.)

In summary, the big ideas behind password recovery are as follows:

Step 1. Boot ROMMON, either by breaking into the boot process from the console or by first removing all the flash memory.

Step 2. Set the configuration register to ignore the startup-config file (for example, confreg 0x2142).

Step 3. Boot the router with an IOS. The router boots with no configuration. Now you can reach enable mode from the console without needing any passwords.

A Specific Password Reset Example

Example F-8 shows a sample password recovery/reset process on a 2901 router. The example begins with Router R1 powered on and the user connected at the console. These 2901 routers use compact flash slots for the primary flash memory; in this example, I removed the flash memory and rebooted the router so that the normal boot process caused ROMMON to load. Look at the highlighted steps in the example for the specific action that resets the password.

Example F-8 *A Password Recovery/Reset Example*

```
! 1) User walks to the router and powers off the router

! 2) User removes all flash memory

! 3) User turns router back on again

System Bootstrap, Version 15.0(1r)M15, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright  2011 by cisco Systems, Inc.

! 4) Several lines of messages omitted: ROMMON is initializing

Readonly ROMMON initialized

rommon 1> confreg 0x2142

You must reset or power cycle for new config to take effect
rommon 2 >
! 5) Just above, user sets the config register to ignore the startup-config.

! 6) User powers off router and then safely plugs the flash back in.

! 7) User powers on router, so that the router now boots IOS.

System Bootstrap, Version 15.0(1r)M15, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright  2011 by cisco Systems, Inc.

! Lots of IOS initialization messages omitted; watch for these next messages


          --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no
```

```
Press RETURN to get started!

! 8) Just above, IOS asked the user if they wanted to do the initial config dialogue.
! That happens when a router boots with no startup config. That confirms the router
! booted and ignored startup-config. The user answered no, to avoid using setup.

! 9) Below, the console user logs in with no passwords required to reach enable mode.

Router>
Router>enable
Router#

! 10) Next, user copies the starting config to make the router do its normal job
Router# copy startup-config running-config
Destination filename [running-config]?
3297 bytes copied in 0.492 secs (6701 bytes/sec)

! 11) User changes the forgotten enable secret password, and sets config register back
! to the default setting of 0x2102
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# enable secret cisco
R1(config)# config-reg 0x2102
R1(config)# ^Z
R1#

! 12) User saves his changes to the password
R1# copy running-config startup-config
Destination filename [startup-config]?
3297 bytes copied in 0.492 secs (6701 bytes/sec)
R1#
```

Note that those last few steps are pretty important. Remember, this process makes the router boot with no initial configuration, so it is clearly disruptive to the normal working state of the router, even beyond the time required to work through the process. The **copy startup-config running-config** command makes up for the fact that the router ignored the startup-config file when it booted IOS. Also, to be ready for the next time the router reloads, put the configuration register value back to its normal permanent value, usually hex 2102.

NOTE When using this process, at the end, take the time to check the interface state of the router interfaces. The **copy running-config startup-config** command could result in some of the interfaces remaining in a shutdown state, depending on the current state of the cabling and the state of the connected devices. So, make sure to check and enable any interfaces with the **no shutdown** interface subcommand.

Managing Configuration Files

Cisco routers and switches happen to use two different configuration files: a startup-config file to save the configuration to use each time the device boots, and the running-config file that holds the currently used configuration for current use inside RAM. By now, you should be used to changing the running-config file using configuration mode and saving the running-config using the **copy running-config startup-config** command.

This last of three major sections of the chapter takes the discussion of configuration files much further. It begins with a look at the traditional methods to copy configuration files outside the router or switch. It then examines more recent options to archive and restore the configuration. This section ends with a brief example of the setup process by which the router can build an initial configuration file.

Copying and Erasing Configuration Files

A good operational plan includes regular backup of the configuration files. The startup and running-config files reside in the router only, and that poses a risk. If the router configuration is never backed up to an external site and the router fails, then even after you replace the router hardware, you may have difficulty piecing a correct router configuration together based on old project notes.

The IOS **copy** command gives you a way to make a copy of the configuration, and has been around for a long time. This command lets you use any of the IFS references to network protocols, including TFTP, FTP, and SCP.

You can also just copy files to and from removable USB flash memory in the router. The USB slots on most recent models of Cisco routers allow you to insert and remove the USB flash drives with IOS running. For instance, a Cisco 2901 router has two USB flash drive slots (usbflash0: and usbflash1:). As demonstrated in Example F-9, an engineer could easily copy the running-config file to flash.

Example F-9 *Copying a File to USB Flash*

```
R1# copy running-config usbflash1:temp-copy-of-config
Destination filename [temp-copy-of-config]?
3159 bytes copied in 0.944 secs (3346 bytes/sec)

R1# dir usbflash1:
Directory of usbflash1:/

! lines listing other files omitted for brevity.
   74  -rw-          3159  Feb 12 2013 22:17:00 +00:00   temp-copy-of-config

7783804928 bytes total (7685111808 bytes free)
R1#
```

While useful in a lab, using USB flash to back up configuration files does not work well with thousands of devices spread around many sites. More than likely, you would back up the files to a more centralized server over the network. The next topic looks at the overall backup and restore plan for systematically backing up configurations.

Traditional Configuration Backup and Restore with the copy Command

One primary motivation of copying the configuration to an external server is to then later restore the configuration if a problem occurs. Like any backup and restore process, the configuration restore process is just as important as backing up the configuration. However, the IOS **copy** command, which has been in IOS for a long time, has an odd behavior when copying files to the running config file to restore the configuration. That odd behavior impacts how to restore the configuration rather than how to back up the configuration.

The **copy** command does not replace the running-config file when copying a configuration into RAM. Effectively, any copy into the running-config file works just as if you entered the commands in the "from" configuration file while in configuration mode. In some cases, adding the new commands does actually replace the old command; for instance, the **ip address** interface subcommand would simply replace the old value. However, with other commands, the command would not replace the old configuration but add to it instead (for instance, with IP **access-list** commands), creating a different configuration.

To drive the point home with a few examples, Figure F-5 shows the cases that result in a replacement of the configuration versus an addition to the configuration. The figure shows commands to copy to and from a TFTP server. Note that the two commands with an asterisk beside them are the ones that effectively add the configuration.

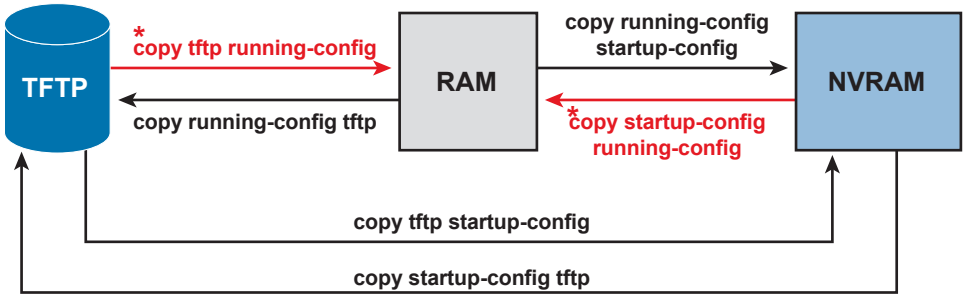


Figure F-5 *Copy into RAM (running-config) Adds Configuration Instead of Replacing*

Because of the effect of copying configurations into the running-config file, the restore process basically avoids using the **copy** command to copy a backup configuration file into running-config. The complete process, using the **copy** command, to both back up and restore configurations, works like this:

Key Topic

Step 1. To back up: Copy the running-config file to some external server; for instance, **copy running-config tftp**.

Step 2. To restore:

A. Copy the backup configuration into the startup-configuration file using the **copy** command, which replaces the startup-config file; for instance, **copy tftp startup-config**.

B. Issue the **reload** command, which reloads, or reboots, the router. That process erases all running config in RAM and then copies the startup-config into RAM as part of the reload process.

Alternatives for Configuration Backup and Restore

Cisco has improved the backup and restore process over the years beyond the basic capabilities of the IOS **copy** command. Two improvements stand out as compared to the use of the **copy** command:

Key Topic

- Create backup configurations, called *archives*, based on the use of the **archive** EXEC command. Archives can be created by command, based on a configured timer, or automatically created each time someone saves the configuration.
- Perform a restore of the archived configuration to the running-config file without requiring a reload by using the **configure replace** command.

The archival process revolves around an IOS file system called the archive. The router just needs to know where to store these configuration files. The router also needs to know whether or not to save the configuration archives automatically. Those rules define the archive—when to automatically save the configuration and where to save them. Example F-10 shows a sample archive configuration, in which the router defines an FTP server at address 192.168.1.170 as the place to store the configurations, with username wendell and password odom. It also defines automatic backup every 1,440 minutes (that is, daily) and stores a copy of the configuration every time the configuration is saved (per the **write-memory** subcommand).

Example F-10 *Creating a Configuration Archive*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# archive
R1(config-archive)# path ftp://wendell:odom@192.168.1.170/
R1(config-archive)# time-period 1440
R1(config-archive)# write-memory
R1(config-archive)# ^Z
R1#
```

NOTE IOS originally used the **write memory** EXEC command to save the configuration; that command was replaced by the **copy running-config startup-config** command. The **archive** feature's **write-memory** command appears to refer to this old EXEC command.

The configuration in the example makes a great improvement over using the **copy** command. First, it improves backups by backing up the configuration automatically. It also improves the restore process because of the **configure replace** command. Basically, the **configure replace** command allows you to copy a configuration archive into the running-config file, so it completely replaces the running-config without requiring a reload of the router. Basically, the router analyzes all the configuration, does a series of comparisons, and determines what sequence of configuration commands would be required to change the configuration correctly—all without reloading the router.

To show the process, Example F-11 shows a sequence in which a router does not have an ACL (141) at the time the archive is made. Then the user changes the configuration to add an ACL 141. Next, the **configure restore** command is used to restore the earlier archived

configuration (which doesn't have ACL 141). Because the restore should replace the running-config file, the running-config should no longer have ACL 141 at the end of the process. The example also shows the hostname being changed as a more obvious confirmation that the **configure replace** command changed the configuration.

Key Topic

Example F-11 *Replacing the Running-config with the* **configure replace** *Command*

```
R1# archive config
Writing -Oct-24-09-46-43.165-2
R1# show archive
The maximum archive configurations allowed is 10.
The next archive file will be named ftp://wendell:odom@192.168.1.170/-<timestamp>-3
 Archive #  Name
   1        ftp://wendell:odom@192.168.1.170/-Oct-24-09-21-38.865-0
   2        ftp://wendell:odom@192.168.1.170/-Oct-24-09-22-22.561-1
   3        ftp://wendell:odom@192.168.1.170/-Oct-24-09-46-43.165-2 <- Most Recent

R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# hostname ridiculousname
ridiculousname(config)# access-list 141 permit ip host 2.2.2.2 host 3.3.3.3
ridiculousname(config)# ^Z
ridiculousname#
*Oct 24 09:47:57.189: %SYS-5-CONFIG_I: Configured from console by console

ridiculousname# configure replace ftp://wendell:odom@192.168.1.170/
  -Oct-24-09-46-43.165-2
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: y
Loading -Oct-24-09-46-43.165-2 !
[OK - 6498/4096 bytes]

Loading -Oct-24-09-46-43.165-2 !
Total number of passes: 1
Rollback Done

R1# show access-list 141
R1#
```

Note that by the end of the example, the hostname has reverted back to the original name (R1) and ACL 141 is no longer configured, as expected.

Erasing Configuration Files

IOS supports three different commands to erase the startup-config file in NVRAM. The **write erase** and **erase startup-config** commands are older, whereas the **erase nvram:** command is the more recent, and recommended, command.

Note that Cisco IOS does not have a command that erases the contents of the running-config file. To clear out the running-config file, simply erase the startup-config file; then **reload** the router so that the router loads an empty startup-config file into the running-config.

Initial Configuration (Setup Mode)

Cisco IOS software supports two primary methods of giving a router or switch an initial basic configuration: configuration mode and setup mode. Setup mode leads a switch administrator through a basic configuration by using questions that prompt the administrator. Because configuration mode is required for most configuration tasks, most networking personnel quickly get comfortable with configuration mode and do not use setup at all. However, new users sometimes like to use setup mode, particularly until they become more familiar with the CLI configuration mode.

Just so you know how to get to setup mode, an engineer can get into setup mode in two ways. Figure F-6 shows one of the methods that occurs during the boot process: If the router boots, with no initial configuration, the router asks if the user wants to enter the "initial configuration dialogue," also known simply as setup mode. You can also enter setup mode by using the **setup** command from privileged mode.

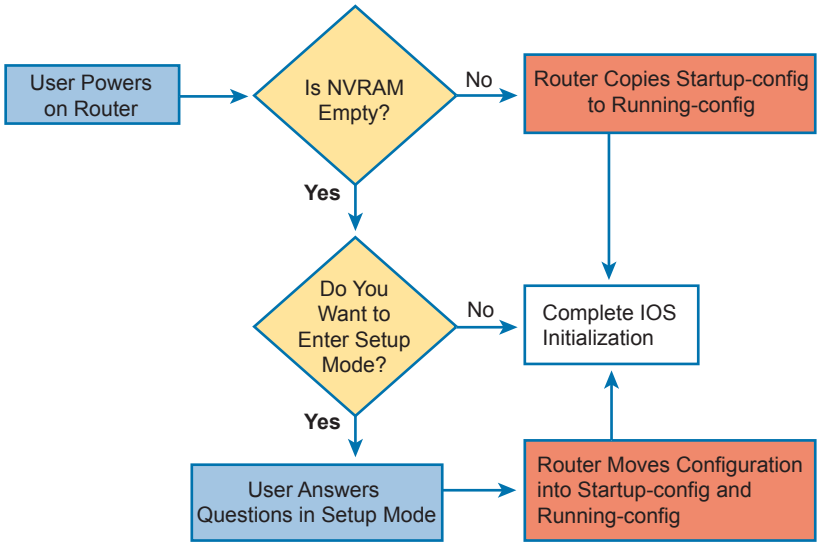


Figure F-6 *Logic and Decisions for Entering Setup Mode After Reload*

NOTE Example F-8, earlier in this chapter, showed the password recovery process. That process caused a router to boot while ignoring the initial configuration, causing the router to ask the user the question shown in Figure F-6.

Command References

Tables F-3 and F-4 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table F-3 Appendix F Configuration Commands

Command	Mode and Purpose
config-register *value*	Global command that sets the hexadecimal value of the configuration register.
boot system {*file-uri* | *filename*}	Global command that identifies an externally located IOS image using a URI.
boot system flash [*flash-fs*:] [*filename*]	Global command that identifies the location of an IOS image in flash memory.
boot system {**tftp** | **ftp**} *filename* [*ip-address*]	Global command that identifies an external server, protocol, and filename to use to load an IOS from an external server.
archive	Global command that moves the user into archive mode.
write-memory	Archive mode command to tell the router to archive the configuration each time the configuration is saved to startup-config.
time-period *minutes*	Archive mode command to define the time between the automatic creation of a new configuration archive.
path *uri*	Archive mode command that defines where to store configurations.
ip ftp username *name*	Global command to define the username used when referencing the **ftp:** IOS file system but not supplying a username.
ip ftp password *pass*	Global command to define the password used when referencing the **ftp:** IOS file system but not supplying a password.
username *name* **privilege-level 15 secret** *pass*	Global command to define a username useful to SCP with a privilege level that enables SCP file transfers.

Table F-4 Appendix F EXEC Command Reference

Command	Purpose
reload	Enable mode EXEC command that reboots the switch or router.
copy *from-location to-location*	Enable mode EXEC command that copies files from one file location to another. Locations include the startup-config and running-config files, files on TFTP and RPC servers, and flash memory.

Command	Purpose
copy running-config startup-config	Enable mode EXEC command that saves the active config, replacing the startup-config file used when the switch initializes.
copy startup-config running-config	Enable mode EXEC command that merges the startup-config file with the currently active config file in RAM.
show running-config	Lists the contents of the running-config file.
write erase **erase startup-config** **erase nvram:**	Each one of the three enable mode EXEC commands erases the startup-config file.
setup	Enable mode EXEC command that places the user in setup mode, in which Cisco IOS asks the user for input on simple switch configurations.
show flash	Lists the names and size of the files in flash memory, as well as noting the amount of flash memory consumed and available.
dir *filesystem:* **dir** *filesystem:directory*	Lists the files in the referenced file system, or file system directory.
verify /md5 *filesystem:name [MD5-hash]*	Performs an MD5 hash of the referenced file and displays the results. If listed, the command compares the MD5 hash in the command with the results of performing MD5 on the local file.
archive config	Creates a copy of the running-config file in the archive.
configure replace *filesystem:name*	Copies the referenced file into running-config, replacing the running-config, without reloading the router.

APPENDIX G

Exam Topics Cross-Reference

This appendix lists the exam topics associated with the CCNA 200-301 exam. Cisco lists the exam topics on its website. Even though changes to the exam topics are rare, you should always review those exam topics for any updates; check www.cisco.com/go/certifications and navigate to the correct exam.

Cisco organizes each list of exam topics by domains, which are major topic areas. Cisco states the percentage of the exam that should come from each domain, so you get some idea of the areas of importance. Traditionally, the score report you receive after taking the exam shows your percentage score in each domain.

This appendix includes two separate types of indices to exam topics:

- **CCNA 200-301 Exam Topic Order:** This section lists the CCNA 200-301 exam topics in the same order Cisco lists them on its website, with a list of associated book chapters. This first list shows a cross-reference from each exam topic to the chapters that include at least some material about each topic.
- **Book Chapter Order Versus CCNA 200-301 Exam Topics:** This lists the same CCNA 200-301 exam topics but indexed by chapter instead of exam topic. This section lists the chapters in this book, along with the exam topics that the chapter includes. This section basically relists the kind of information found on the first page of each chapter, just in condensed form in one place.

CCNA 200-301 Exam Topic Order

The CCNA 200-301 exam includes six major topic areas (domains), each with a percentage listed. Table G-1 lists the domains and their percentages.

Table G-1 CCNA 200-301 Exam Topic Domains

Domain	Percentage
Domain 1: Network Fundamentals	20%
Domain 2: Network Access	20%
Domain 3: IP Connectivity	25%
Domain 4: IP Services	10%
Domain 5: Security Fundamentals	15%
Domain 6: Automation and Programmability	10%

Tables G-2 through G-7 list the exam topics within each of the six domains. Note that the *CCNA 200-301 Official Cert Guide, Volume 1*, covers some of the exam topics, while this book covers the rest. These tables show the chapters in each book that cover each exam topic.

Table G-2 CCNA 200-301 Domain 1 Exam Topics (Network Fundamentals)

Exam Topic	Vol 1 Chapter(s)	Vol 2 Chapter(s)
1.1 Explain the Role and function of Network Components	2, 3, 5, 7, 26	5, 16, 17
1.1.a Routers	3, 15	
1.1.b L2 and L3 Switches	2, 5, 7	
1.1.c Next-generation firewalls and IPS		5
1.1.d Access points	26	
1.1.e Controllers (Cisco DNA Center and WLC)	29	17
1.1.f Endpoints		16
1.1.g Servers		16
1.2 Describe characteristics of network topology architectures	2, 3	13, 14, 15, 16
1.2.a 2 tier		13
1.2.b 3 tier		13
1.2.c Spine-leaf		16
1.2.d WAN	3	14
1.2.e Small office/home office (SOHO)	2, 15	13
1.2.f On-premises and cloud		15
1.3 Compare physical interface and cabling types	1, 2	13
1.3.a Single-mode fiber, multimode fiber, copper	1, 2	
1.3.b Connections (Ethernet shared media and point-to-point)	1, 2	
1.3.c Concepts of PoE		13
1.4 Identify interface and cable issues (collisions, errors, mismatch duplex, and/or speed)	7	
1.5 Compare TCP to UDP		1
1.6 Configure and verify IPv4 addressing and subnetting	6, 11, 12, 13, 14, 15, 17, 22	
1.7 Describe the need for private IPv4 addressing	11, 16	
1.8 Configure and verify IPv6 addressing and prefix	23, 24	
1.9 Compare IPv6 address types	23, 24	
1.9.a Global unicast	23, 24	
1.9.b Unique local	23, 24	
1.9.c Link local	24	
1.9.d Anycast	24	
1.9.e Multicast	24	
1.9.f Modified EUI 64	24	

Exam Topic	Vol 1 Chapter(s)	Vol 2 Chapter(s)
1.10 Identify IP parameters for Client OS (Windows, Mac OS, Linux)		7
1.11 Describe wireless principles	26	
1.11.a Nonoverlapping Wi-Fi channels	26	
1.11.b SSID	26	
111.c RF	26	
1.11.d Encryption	28	
1.12 Explain virtualization fundamentals (virtual machines)		15
1.13 Describe switching concepts	5, 8	
1.13.a MAC learning and aging	5, 8	
1.13.b Frame switching	5, 8	
1.13.c Frame flooding	5, 8	
1.13.d MAC address table	5, 8	

Table G-3 CCNA 200-301 Domain 2 Exam Topics (Network Access)

Exam Topic	Vol 1 Chapter(s)	Vol 2 Chapter(s)
2.1 Configure and verify VLANs (normal range) spanning multiple switches	8	
2.1.a Access ports (data and voice)	8	
2.1.b Default VLAN	8	
2.1.c Connectivity	8	
2.2 Configure and verify interswitch connectivity	8	
2.2.a Trunk ports	8	
2.2.b 802.1Q	8	
2.2.c Native VLAN	8	
2.3 Configure and verify Layer 2 discovery protocols (Cisco Discovery Protocol and LLDP)		9
2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)	8, 9, 10, 17	
2.5 Describe the need for and basic operations of Rapid PVST+ Spanning Tree Protocol and identify basic operations	5, 9, 10	
2.5.a Root port, root bridge (primary/secondary), and other port names	9, 10	
2.5.b Port states (forwarding/blocking)	9, 10	
2.5.c PortFast benefits	9, 10	
2.6 Compare Cisco Wireless Architectures and AP modes	27	

Exam Topic	Vol 1 Chapter(s)	Vol 2 Chapter(s)
2.7 Describe physical infrastructure connections of WLAN components (AP, WLC, access/trunk ports, and LAG)	29	
2.8 Describe AP and WLC management access connections (Telnet, SSH, HTTP, HTTPS, console, and TACACS+/RADIUS)	29	
2.9 Configure the components of a wireless LAN access for client connectivity using GUI only such as WLAN creation, security settings, QoS profiles, and advanced WLAN settings	29	

Table G-4 CCNA 200-301 Domain 3 Exam Topics (IP Connectivity)

Exam Topic	Vol 1 Chapter(s)	Vol 2 Chapter(s)
3.1 Interpret the components of routing table	16	
3.1.a Routing protocol code	16	
3.1.b Prefix	16	
3.1.c Network mask	16	
3.1.d Next hop	16	
3.1.e Administrative distance	16	
3.1.f Metric	16	
3.1.g Gateway of last resort	16	
3.2 Determine how a router makes a forwarding decision by default	16	
3.2.a Longest match	16	
3.2.b Administrative distance	16, 19, 20	
3.2.c Routing protocol metric	19, 20	
3.3 Configure and verify IPv4 and IPv6 static routing	16, 18, 25	
3.3.a Default route	16, 18, 25	
3.3.b Network route	16, 18, 25	
3.3.c Host route	16, 18, 25	
3.3.d Floating static	16, 18, 25	
3.4 Configure and verify single area OSPFv2	19, 20, 21	
3.4.a Neighbor adjacencies	19, 20, 21	
3.4.b Point-to-point	19, 20, 21	
3.4.c Broadcast (DR/BDR selection)	19, 20, 21	
3.4.d Router ID	19, 20, 21	
3.5 Describe the purpose of First Hop Redundancy Protocol		12

Table G-5 CCNA 200-301 Domain 4 Exam Topics (IP Services)

Exam Topics	Vol 1 Chapter(s)	Vol 2 Chapter(s)
4.1 Configure and verify inside source NAT using static and pools		10
4.2 Configure and verify NTP operating in a client and server mode		9
4.3 Explain the role of DHCP and DNS within the network		1, 7
4.4 Explain the function of SNMP in network operations		12
4.5 Describe the use of syslog features including facilities and levels		9
4.6 Configure and verify DHCP client and relay	6	7
4.7 Explain the per-hop behavior (PHB) for QoS such as classification, marking, queuing, congestion, policing, shaping		11
4.8 Configure network devices for remote access using SSH	6	5
4.9 Describe the capabilities and function of TFTP/FTP in the network		12

Table G-6 CCNA 200-301 Domain 5 Exam Topics (Security Fundamentals)

Exam Topics	Vol 1 Chapter(s)	Vol 2 Chapter(s)
5.1 Define key security concepts (threats, vulnerabilities, exploits, and mitigation techniques)		4
5.2 Describe security program elements (user awareness, training, and physical access control)		4
5.3 Configure device access control using local passwords	6	5
5.4 Describe security password policies elements, such as management, complexity, and password alternatives (multifactor authentication, certificates, and biometrics)		4
5.5 Describe remote access and site-to-site VPNs		14
5.6 Configure and verify access control lists		2, 3
5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)		6, 8
5.8 Differentiate authentication, authorization, and accounting concepts		4
5.9 Describe wireless security protocols (WPA, WPA2, and WPA3)	28	
5.10 Configure WLAN using WPA2 PSK using the GUI	29	

Table G-7 CCNA 200-301 Domain 6 Exam Topics (Automation and Programmability)

Exam Topics	Vol 1 Chapter(s)	Vol 2 Chapter(s)
6.1 Explain how automation impacts network management		16
6.2 Compare traditional networks with controller-based networking		16
6.3 Describe controller-based and software-defined architectures (overlay, underlay, and fabric)		16, 17
6.3.a Separation of control plane and data plane		16, 17
6.3.b Northbound and southbound APIs		16, 17
6.4 Compare traditional campus device management with Cisco DNA Center enabled device management		17
6.5 Describe characteristics of REST-based APIs (CRUD, HTTP verbs, and data encoding)		18
6.6 Recognize the capabilities of configuration management mechanisms Puppet, Chef, and Ansible		19
6.7 Interpret JSON encoded data		18

Book Chapters, with Exam Topics Covered in Each

Cisco organizes its exam topics based on the outcome of your learning experience, which is typically not a reasonable order for building the content of a book or course. This section lists this book's chapters in sequence, with the exam topics covered in each chapter.

Book Chapter	Exam Topics Covered
Part I: IP Access Control Lists	
Chapter 1: Introduction to TCP/IP Transport and Applications	**1.0 Network Fundamentals** 1.5 Compare TCP to UDP **4.0 IP Services** 4.3 Explain the role of DHCP and DNS in the network
Chapter 2: Basic IPv4 Access Control Lists	**5.0 Security Fundamentals** 5.6 Configure and verify access control lists
Chapter 3: Advanced IPv4 Access Control Lists	**5.0 Security Fundamentals** 5.6 Configure and verify access control lists

Book Chapter	Exam Topics Covered
Part II: Security Services	
Chapter 4: Security Architectures	**5.0 Security Fundamentals** 5.1 Define key security concepts (threats, vulnerabilities, exploits, and mitigation techniques) 5.2 Describe security program elements (user awareness, training, and physical access control) 5.4 Describe security password policies elements, such as management, complexity, and password alternatives (multifactor authentication, certificates, and biometrics) 5.8 Differentiate authentication, authorization, and accounting concepts
Chapter 5: Securing Network Devices	**1.0 Network Fundamentals** 1.1 Explain the Role of Network Components 1.1.c Next-generation firewalls and IPS **4.0 IP Services** 4.8 Configure network devices for remote access using SSH **5.0 Security Fundamentals** 5.3 Configure device access control using local passwords
Chapter 6: Implementing Switch Port Security	**5.0 Security Fundamentals** 5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)
Chapter 7: Implementing DHCP	**1.0 Network Fundamentals** 1.10 Identify IP parameters for Client OS (Windows, Mac OS, Linux) **4.0 IP Services** 4.3 Explain the role of DHCP and DNS within the network 4.6 Configure and verify DHCP client and relay
Chapter 8: DHCP Snooping and ARP Inspection	**5.0 Security Fundamentals** 5.7 Configure Layer 2 security features (DHCP snooping, dynamic ARP inspection, and port security)

Book Chapter	Exam Topics Covered
Part III: IP Services	
Chapter 9: Device Management Protocols	**2.0 Network Access** 2.3 Configure and verify Layer 2 discovery protocols (Cisco Discovery Protocol and LLDP) **4.0 IP Services** 4.2 Configure and verify NTP operating in a client and server mode 4.5 Describe the use of syslog features including facilities and levels
Chapter 10: Network Address Translation	**4.0 IP Services** 4.7 Configure and verify inside source NAT using static and pools
Chapter 11: Quality of Service (QoS)	**4.0 IP Services** 4.7 Explain the forwarding per-hop behavior (PHB) for QoS such as classification, marking, queuing, congestion, policing, shaping
Chapter 12: Miscellaneous IP Services	**3.0 IP Connectivity** 3.5 Describe the purpose of First Hop Redundancy Protocol **4.0 Infrastructure Services** 4.4 Explain the function of SNMP in network operations 4.9 Describe the capabilities and function of TFTP/FTP in the network
Part IV: Network Architecture	
Chapter 13: LAN Architecture	**1.0 Network Fundamentals** 1.2 Describe characteristics of network topology architectures 1.2.a 2 tier 1.2.b 3 tier 1.2.e Small office/home office (SOHO) 1.3 Compare physical interface and cabling types 1.3.c Concepts of PoE
Chapter 14: WAN Architecture	**1.0 Network Fundamentals** 1.2 Describe the characteristics of network topology architecture 1.2.d WAN **5.0 Security Fundamentals** 5.5 Describe remote access and site-to-site VPNs

Book Chapter	Exam Topics Covered
Chapter 15: Cloud Architecture	**1.0 Network Fundamentals** 1.2 Describe the characteristics of network topology architectures 1.2.f On-premises and cloud 1.12 Explain virtualization fundamentals (virtual machines)
Part V: Network Automation	
Chapter 16: Introduction to Controller-Based Networking	**6.0 Automation and Programmability** 6.1 Explain how automation impacts network management 6.2 Compare traditional networks with controller-based networking 6.3 Describe controller-based and software-defined architectures (overlay, underlay, and fabric) 6.3.a Separation of control plane and data plane 6.3.b Northbound and southbound APIs
Chapter 17: Cisco Software-Defined Access	**1.0 Network Fundamentals** 1.1 Explain the role and function of network components 1.1.e Controllers (Cisco DNA Center and WLC) **6.0 Automation and Programmability** 6.1 Explain how automation impacts network management 6.2 Compare traditional networks with controller-based networking 6.3 Describe controller-based and software-defined architectures (overlay, underlay, and fabric)
Chapter 18: Understanding REST and JSON	**6.0 Automation and Programmability** 6.5 Describe characteristics of REST-based APIs (CRUD, HTTP verbs, and data encoding) 6.7 Interpret JSON encoded data
Chapter 19: Ansible, Puppet, and Chef	**6.0 Automation and Programmability** 6.6 Recognize the capabilities of configuration mechanisms Puppet, Chef, and Ansible

This page intentionally left blank

Appendix H

Study Planner

Practice Test	Reading	Task

Element	Task	Goal Date	First Date Completed	Second Date Completed (Optional)	Notes
Introduction	Read Introduction				
1. Introduction to TCP/IP Transport and Applications	Read Foundation Topics				
1. Introduction to TCP/IP Transport and Applications	Review Key Topics using the book or companion website				
1. Introduction to TCP/IP Transport and Applications	Define Key Terms using the book or companion website				
1. Introduction to TCP/IP Transport and Applications	Repeat DIKTA questions using the book or PTP exam engine				
1. Introduction to TCP/IP Transport and Applications	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
2. Basic IPv4 Access Control Lists	Read Foundation Topics				
2. Basic IPv4 Access Control Lists	Review Key Topics using the book or companion website				
2. Basic IPv4 Access Control Lists	Define Key Terms using the book or companion website				
2. Basic IPv4 Access Control Lists	Repeat DIKTA questions using the book or PTP exam engine				
2. Basic IPv4 Access Control Lists	Complete all memory tables in this chapter using the companion website				
2. Basic IPv4 Access Control Lists	Practice configuring IPv4 access lists using Appendix E on the companion website				
2. Basic IPv4 Access Control Lists	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
3. Advanced IPv4 Access Control Lists	Read Foundation Topics				
3. Advanced IPv4 Access Control Lists	Review Key Topics using the book or companion website				
3. Advanced IPv4 Access Control Lists	Define Key Terms using the book or companion website				

3. Advanced IPv4 Access Control Lists	Repeat DIKTA questions using the book or PTP exam engine				
3. Advanced IPv4 Access Control Lists	Complete all memory tables in this chapter using the companion website				
3. Advanced IPv4 Access Control Lists	Watch video for this chapter using the companion website				
3. Advanced IPv4 Access Control Lists	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part I. IP Access Control Lists	Complete all exercises in Part I Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
4. Security Architectures	Read Foundation Topics				
4. Security Architectures	Review Key Topics using the book or companion website				
4. Security Architectures	Define Key Terms using the book or companion website				
4. Security Architectures	Repeat DIKTA questions using the book or PTP exam engine				
4. Security Architectures	Complete all memory tables in this chapter using the companion website				
4. Security Architectures	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
5. Securing Network Devices	Read Foundation Topics				
5. Securing Network Devices	Review Key Topics using the book or companion website				
5. Securing Network Devices	Define Key Terms using the book or companion website				
5. Securing Network Devices	Repeat DIKTA questions using the book or PTP exam engine				
5. Securing Network Devices	Review command tables for this chapter				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
6. Implementing Switch Port Security	Read Foundation Topics				
6. Implementing Switch Port Security	Review Key Topics using the book or companion website				
6. Implementing Switch Port Security	Define Key Terms using the book or companion website				
6. Implementing Switch Port Security	Repeat DIKTA questions using the book or PTP exam engine				
6. Implementing Switch Port Security	Complete config checklists in this chapter using the companion website				
6. Implementing Switch Port Security	Complete all memory tables in this chapter using the companion website				
6. Implementing Switch Port Security	Watch video for this chapter using the companion website				
6. Implementing Switch Port Security	Review command tables for this chapter				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
7. Implementing DHCP	Read Foundation Topics				
7. Implementing DHCP	Review Key Topics using the book or companion website				
7. Implementing DHCP	Define Key Terms using the book or companion website				
7. Implementing DHCP	Repeat DIKTA questions using the book or PTP exam engine				
7. Implementing DHCP	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
8. Implementing DHCP Snooping and ARP Inspection	Read Foundation Topics				
8. Implementing DHCP Snooping and ARP Inspection	Review Key Topics using the book or companion website				
8. Implementing DHCP Snooping and ARP Inspection	Define Key Terms using the book or companion website				
8. Implementing DHCP Snooping and ARP Inspection	Repeat DIKTA questions using the book or PTP exam engine				
8. Implementing DHCP Snooping and ARP Inspection	Complete config checklists in this chapter using the companion website				
8. Implementing DHCP Snooping and ARP Inspection	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part II. Security Services	Complete all exercises in Part II Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
9. Device Management Protocols	Read Foundation Topics				
9. Device Management Protocols	Review Key Topics using the book or companion website				
9. Device Management Protocols	Define Key Terms using the book or companion website				
9. Device Management Protocols	Repeat DIKTA questions using the book or PTP exam engine				
9. Device Management Protocols	Complete all memory tables in this chapter using the companion website				
9. Device Management Protocols	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
10. Network Address Translation	Read Foundation Topics				
10. Network Address Translation	Review Key Topics using the book or companion website				
10. Network Address Translation	Define Key Terms using the book or companion website				
10. Network Address Translation	Repeat DIKTA questions using the book or PTP exam engine				

10. Network Address Translation	Complete config checklists in this chapter using the companion website				
10. Network Address Translation	Review command tables for this chapter				
10. Network Address Translation	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
11. Quality of Service (QoS)	Read Foundation Topics				
11. Quality of Service (QoS)	Review Key Topics using the book or companion website				
11. Quality of Service (QoS)	Define Key Terms using the book or companion website				
11. Quality of Service (QoS)	Repeat DIKTA questions using the book or PTP exam engine				
11. Quality of Service (QoS)	Complete all memory tables in this chapter using the companion website				
11. Quality of Service (QoS)	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
12. Miscellaneous IP Services	Read Foundation Topics				
12. Miscellaneous IP Services	Review Key Topics using the book or companion website				
12. Miscellaneous IP Services	Define Key Terms using the book or companion website				
12. Miscellaneous IP Services	Repeat DIKTA questions using the book or PTP exam engine				
12. Miscellaneous IP Services	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part III. IP Services	Complete all exercises in Part III Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
13. LAN Architecture	Read Foundation Topics				
13. LAN Architecture	Review Key Topics using the book or companion website				
13. LAN Architecture	Define Key Terms using the book or companion website				
13. LAN Architecture	Repeat DIKTA questions using the book or PTP exam engine				
13. LAN Architecture	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
14. WAN Architecture	Read Foundation Topics				
14. WAN Architecture	Review Key Topics using the book or companion website				
14. WAN Architecture	Define Key Terms using the book or companion website				

14. WAN Architecture	Repeat DIKTA questions using the book or PTP exam engine				
14. WAN Architecture	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
15. Cloud Architecture	Read Foundation Topics				
15. Cloud Architecture	Review Key Topics using the book or companion website				
15. Cloud Architecture	Define Key Terms using the book or companion website				
15. Cloud Architecture	Repeat DIKTA questions using the book or PTP exam engine				
15. Cloud Architecture	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part IV. Network Architecture	Complete all exercises in Part IV Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
16. Introduction to Controller-Based Networking	Read Foundation Topics				
16. Introduction to Controller-Based Networking	Review Key Topics using the book or companion website				
16. Introduction to Controller-Based Networking	Define Key Terms using the book or companion website				
16. Introduction to Controller-Based Networking	Repeat DIKTA questions using the book or PTP exam engine				
16. Introduction to Controller-Based Networking	Complete all memory tables in this chapter using the companion website				
16. Introduction to Controller-Based Networking	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
17. Cisco Software-Defined Access (SDA)	Read Foundation Topics				
17. Cisco Software-Defined Access (SDA)	Review Key Topics using the book or companion website				
17. Cisco Software-Defined Access (SDA)	Define Key Terms using the book or companion website				
17. Cisco Software-Defined Access (SDA)	Repeat DIKTA questions using the book or PTP exam engine				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
18. Understanding REST and JSON	Read Foundation Topics				
18. Understanding REST and JSON	Review Key Topics using the book or companion website				
18. Understanding REST and JSON	Define Key Terms using the book or companion website				

18. Understanding REST and JSON	Repeat DIKTA questions using the book or PTP exam engine				
18. Understanding REST and JSON	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
19. Understanding Ansible, Puppet, and Chef	Read Foundation Topics				
19. Understanding Ansible, Puppet, and Chef	Review Key Topics using the book or companion website				
19. Understanding Ansible, Puppet, and Chef	Define Key Terms using the book or companion website				
19. Understanding Ansible, Puppet, and Chef	Repeat DIKTA questions using the book or PTP exam engine				
19. Understanding Ansible, Puppet, and Chef	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part V. Network Automation	Complete all exercises in Part V Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
Final Review	Take practice test in study mode for all Book Questions in practice test software				
Final Review	Review all Key Topics in all chapters or in the Key Topics App using the companion website				
Final Review	Review all Key Terms in all chapters or using the Key Terms Flashcards on the companion website				
Final Review	Complete all memory tables for all chapters using the companion website				
Final Review	Take practice test in practice exam mode using Exam Bank #1 questions for all chapters				
Final Review	Take practice test in practice exam mode using Exam Bank #2 questions for all chapters				