



Implementing Cisco IP Routing (ROUTE)

Foundation Learning Guide

Foundation learning for the CCNP ROUTE 642-902 Exam



ciscopress.com

Diane Teare

Copyright

Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide

Foundation learning for the ROUTE 642-902 Exam

Diane Teare

Copyright© 2010 Cisco Systems, Inc.

Published by:
Cisco Press
800 East 96th Street
Indianapolis, IN 46240 USA

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without written permission from the publisher, except for the inclusion of brief quotations in a review.

Printed in the United States of America

First Printing June 2010

Library of Congress Cataloging-in-Publication Data is on file.

ISBN-13: 978-1-58705-882-0

Warning and Disclaimer

This book is designed to provide information about Cisco routing. 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 authors 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.

Corporate and Government Sales

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact: U.S. Corporate and Government Sales 1-800-382-3419 corpsales@pearsontechgroup.com

For sales outside the United States please contact: International Sales international@pearsoned.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 e-mail at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Publisher: Paul Boger

Associate Publisher: Dave Dusthimer

Executive Editor: Mary Beth Ray

Managing Editor: Sandra Schroeder

Development Editor: Dayna Isley

Editorial Assistant: Vanessa Evans

Book Designer: Louisa Adair

Cover Designer: Sandra Schroeder

Composition: Mark Shirar

Business Operation Manager, Cisco Press: Anand Sundaram

Manager Global Certification: Erik Ullanderson

Copy Editor: Keith Cline

Proofreader: Leslie Joseph

Project Editor: Mandie Frank

Indexer: Tim Wright

Technical Editors: Sonya Coker, Jeremy Creech, Rick Graziani, Scott Hogg, David Kotfila, Wayne Lewis, Jim Lorenz, Snezhy Neshkova, Allan Reid, Jerold Swan, Bob Vachon



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.

CCDE, CCENT, Cisco Eos, Cisco HealthPresence, the Cisco logo, Cisco Lumin, Cisco Nexus, Cisco StadiumVision, Cisco TelePresence, Cisco WebEx, DCE, and Welcome to the Human Network are trademarks; Changing the Way We Work, Live, Play, and Learn and Cisco Store are service marks; and Access Registrar, Aironet, AsyncOS, Bringing the Meeting To You, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, CCVP,

Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unity, Collaboration Without Limitation, EtherFast, EtherSwitch, Event Center, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, iQuick Study, IronPort, the IronPort logo, LightStream, Linksys, MediaTone, MeetingPlace, MeetingPlace Chime Sound, MGX, Networkers, Networking Academy, Network Registrar, PCNow, PIX, PowerPanels, ProConnect, ScriptShare, SenderBase, SMARTnet, Spectrum Expert, StackWise, The Fastest Way to Increase Your Internet Quotient, TransPath, WebEx, and the WebEx logo are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0812R)

Dedications

This book is dedicated to my inspirational husband, Allan Martin, whose love and encouragement is so welcome; to our delightful and loving son, Nicholas, and his unending curiosity to discover everything about everything; and to my parents, Syd and Beryl, for their constant support and encouragement.

About the Author

Diane Teare is a professional in the networking, training, project management, and e-learning fields. She has more than 25 years of experience in designing, implementing, and troubleshooting network hardware and software, and has been involved in teaching, course design, and project management. She has extensive knowledge of network design and routing technologies, and is an instructor with one of the largest authorized Cisco Learning Partners. She was the director of e-learning for the same company, where she was responsible for planning and supporting all the company's e-learning offerings in Canada, including Cisco courses. Diane has a bachelor's degree in applied science in electrical engineering and a master's degree in applied science in management science. She currently holds her Cisco Certified Network Professional (CCNP), Cisco Certified Design Professional (CCDP), and Project Management Professional (PMP) certifications. She co-authored the Cisco Press titles *Designing Cisco Network Service Architectures (ARCH)*, Second Edition; *Campus Network Design Fundamentals*; the three editions of *Authorized Self-Study Guide Building Scalable Cisco Internetworks (BSCI)*; and *Building Scalable Cisco Networks*. Diane edited the two editions of the *Authorized Self-Study Guide Designing for Cisco Internetwork Solutions (DESGN)* and *Designing Cisco Networks*.

About the Contributor

Catherine Paquet is a practitioner in the field of Internetworking, Network Security, and Security Financials. So far, she has published eight books with Cisco Press. Catherine has in-depth knowledge of security systems, remote access, and routing technology. She is a Cisco Certified Security Professional (CCSP), a Cisco Certified Network Professional (CCNP), and a Certified Cisco Systems Instructor (CCSI) with the largest Cisco training partner, Global Knowledge. Catherine teaches many Cisco security classes such as *Securing Networks with ASA (SNAF,SNAA)*. She also lectures directly with Cisco Systems in emerging countries on the business case for network Security. Her most recent consulting projects include conducting security assessments, performing network designs, configuring and implementing security solutions such as firewalls, virtual private networks, web filters, and intrusion prevention solutions.

About the Technical Reviewers

Sonya Coker has worked in the Cisco Networking Academy program since 1999 when she started a local academy. She has taught student and instructor classes locally and internationally in topics ranging from IT Essentials to CCNP. As a member of the Cisco Networking Academy development team, she has provided subject matter expertise on new courses and course revisions.

Jeremy Creech is a Learning and Development Manager for Cisco Systems with more than 13 years of experience in researching, implementing, and managing data and voice networks. Currently, he is a curriculum development manager for the Cisco Networking Academy Program, leveraging his experience as the Content Development Manager for CCNP Certification exams. He has recently completed curriculum development initiatives for ROUTE, SWITCH, TSHOOT and CCNA Security.

Scott Hogg has been a network computing consultant for more than 18 years. He has a B.S. in computer science, a M.S. in telecommunications, along with his CCIE (No. 5133) and CISSP (No. 4610). For the past ten years, Scott has been researching IPv6 technologies and recently has helped many organizations with their IPv6 planning. Scott has given numerous presentations and demonstrations of IPv6 technologies and authored the book titled *IPv6 Security*. He is also currently the Chair of the Rocky Mountain IPv6 Task Force.

Rick Graziani teaches computer science and computer networking courses at Cabrillo College in Aptos, California. Rick has worked and taught in the computer networking and information technology field for almost 30 years. Before teaching, Rick worked in IT for various companies, including Santa Cruz Operation, Tandem Computers, and Lockheed Missiles and Space Corporation. He holds a Master of Arts degree in computer science and systems theory from California State University Monterey Bay. Rick also does consulting work for Cisco Systems and other companies. When Rick is not working, he is most likely surfing. Rick is an avid surfer who enjoys surfing at his favorite Santa Cruz breaks.

David Kotfila, CCNA, CCDA, CCNP, CCDP, CCSP, CCVP, CCAI, teaches in the Computer Science department at Rensselaer Polytechnic Institute, Troy, New York. More than 550 of his students have received their CCNA, 200 have received their CCNP, and 14 have received their CCIE. David likes to spend time with his wife, Kate, his daughter, Charis, and his son, Chris. David enjoys hiking, kayaking, and reading.

Wayne Lewis has been a faculty member at Honolulu Community College since receiving a Ph.D. in math from the University of Hawaii at Manoa in 1992, specializing in finite rank torsion-free modules over a Dedekind domain. Since 1992, he has served as a math instructor, as the state school-to-work coordinator, and as the legal main contact for the Cisco Academy Training Center (CATC).

Dr. Lewis manages the CATC for CCNA, CCNP, and Security, based at Honolulu Community College, which serves Cisco Academies at universities, colleges, and high schools in Hawaii, Guam, and American Samoa. Since 1998, he has taught routing, multilayer switching, remote access, troubleshooting, network security, and wireless networking to instructors from universities, colleges, and high schools in Australia, Britain, Canada, Central America, China, Germany, Hong Kong, Hungary, Indonesia, Italy, Japan, Korea, Mexico, Poland, Singapore, Sweden, Taiwan, and South America, both onsite and at Honolulu Community College.

Jim Lorenz is an instructor and curriculum developer for the Cisco Networking Academy Program. Jim has co-authored Lab Companions for the CCNA courses and the textbooks for the Fundamentals of UNIX course.

He has more than 25 years of experience in information systems, ranging from programming and database administration to network design and project management. Jim has developed and taught computer and networking courses for both public and private institutions. As the Cisco Academy Manager at Chandler-Gilbert College in Arizona, he was instrumental in starting the Information Technology Institute (ITI) and developed several certificates and degree programs.

Jim co-authored the CCNA Discovery online academy courses, Networking for Home and Small Businesses, and Introducing Routing and Switching in the Enterprise, with Allan Reid. Most recently, he developed the hands-on labs for the CCNA Security course and the CCNPv6 Troubleshooting course.

Snezhy Neshkova is a Cisco Certified Internetwork Expert (CCIE No. 11931) since 2003. She has more than 20 years of networking experience, including IT field services and support, management of information systems, and all aspects of networking education. Snezhy has developed and taught CCNA and CCNP networking courses to instructors from universities, colleges, and high schools in Canada, the United States, and Europe. Snezhy's passion is to empower students to become successful and compassionate lifelong learners. Snezhy holds a Master of Science degree in computer science from Technical University, Sofia, Bulgaria.

Allan Reid (CCNA, CCNA-W, CCDA, CCNP, CCDP, CCAI, MLS) is a professor in information and communications engineering technology and the lead instructor at the Centennial College CATC in Toronto, Canada. He has developed and taught networking courses for both private and public organizations and has been instrumental in the development and implementation of numerous certificate, diploma, and degree programs in networking. Outside of his academic responsibilities, Allan has been active in the computer and

networking fields for more than 25 years and is currently a principal in a company specializing in the design, management, and security of network solutions for small and medium-sized companies. Allan is a curriculum and assessment developer for the Cisco Networking Academy program and has authored several Cisco Press titles.

Jerold Swan, CCIE No. 17783, CCSP, works as a senior network engineer for the Southern Ute Indian Tribe Growth Fund in southwest Colorado. Before that, he was a Cisco instructor for Global Knowledge. He has also worked in IT in the service provider and higher-education sectors. His areas of interest include routing protocols, security, and network monitoring. He is a graduate of Stanford University. His other interests include trail running, mountain biking, and volunteer search and rescue.

Bob Vachon, CCNP, CCNA-S, CCAI, is a professor in the Computer Systems Technology program at Cambrian College and has more than 20 years of experience in the networking field. In 2001, he began collaborating with the Cisco Networking Academy on various curriculum development projects, including CCNA, CCNA Security, and CCNP courses. For 3 years, Bob was also part of an elite team authoring CCNP certification exam questions. In 2007, Bob co-authored the *CCNA Exploration: Accessing the WAN* Cisco Press book.

Acknowledgments

I want to thank many people for helping to put this book together:

The Cisco Press team: Mary Beth Ray, the executive editor, coordinated the whole project, steered the book through the necessary processes, and understood when the inevitable snags appeared. Patrick Kanouse, the managing editor, brought the book to production. Vanessa Evans was once again instrumental in organizing the logistics and administration. Dayna Isley, the development editor, has been invaluable in coordinating and producing a high-quality manuscript.

I also want to thank Mandie Frank, the project editor, and Keith Cline, the copy editor, for their excellent work in steering this book through the editorial process.

The Cisco ROUTE course development team: Many thanks to the members of the team who developed the ROUTE course.

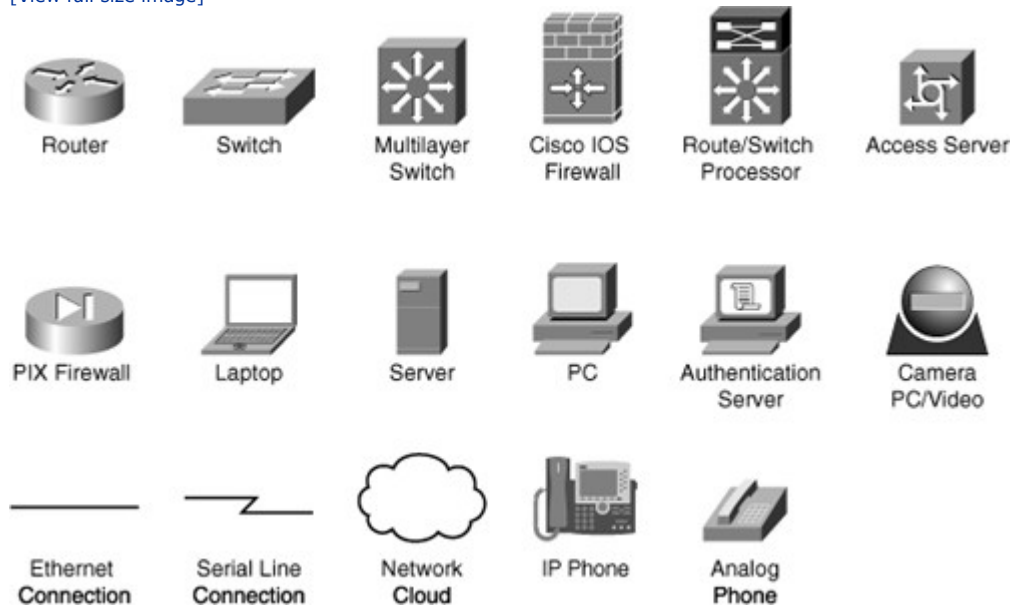
The contributing author: I want to thank my friend and colleague, Catherine Paquet, for agreeing to contribute a chapter to this book, enriching it with her expertise and ensuring that the schedule did not suffer. I owe you, Catherine!

The technical reviewers: I want to thank the technical reviewers of this book for their thorough, detailed review and valuable input. Special thanks to Bob Vachon for his invaluable (and tedious, I'm sure) screen-capture work.

My family: Of course, this book would not have been possible without the endless understanding and patience of my family. They have always been there to motivate and inspire me. I am forever grateful.

Icons Used in This Book

[\[View full size image\]](#)



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.

Chapter 1. Routing Services

This chapter covers the following topics:

- [Complex Enterprise Network Frameworks, Architectures, and Models](#)
- [Creating, Documenting, and Executing an Implementation Plan](#)
- [Reviewing IP Routing Principles](#)

This chapter first describes the frameworks, architectures, and models used in complex enterprise network designs. The next section explores the process of creating, documenting, and executing a network implementation plan. The chapter concludes with a review of IP routing principles.

Complex Enterprise Network Frameworks, Architectures, and Models

This section introduces converged networks and the variety of traffic within them. To accommodate the requirements of such networks, Cisco has introduced the Intelligent Information Network (IIN) strategy, along with the Service-Oriented Network Architecture (SONA) framework, which guides the evolution of enterprise networks toward an IIN, both of which are described in this section.

This section also introduces the components of the Cisco Enterprise Architecture, and describes the traditional hierarchical network model and the Enterprise Composite Network Model.

Traffic Conditions in a Converged Network

A converged network is one in which data, voice, and video traffic coexist on a single network. When voice and video are transported across a network, the voice and video are seen by the network as being just like any other application data.

Converged networks contain a variety of different types of traffic, including the following:

- **Voice and video traffic**— Examples include IP telephony, video broadcast, and conferencing.
- **Voice applications traffic**— Generated by voice-related applications, such as contact centers.
- **Mission-critical traffic**— Generated by applications critical to an organization (for example, information generated by a stock exchange application at a finance company, patient records at a hospital, and so forth).
- **Transactional traffic**— Generated by applications such as those for e-commerce.
- **Routing protocol traffic**— Data from whichever routing protocols are running in the network, such as the Routing Information Protocol (RIP), Open Shortest Path First Protocol (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), Intermediate System-to-Intermediate System Protocol (IS-IS), and Border Gateway Protocol (BGP).
- **Network management traffic**— Including information about the status of the network and its devices.

Note

Although IS-IS is not covered further in this book, it is included in this chapter for completeness.

The requirements on the network differ significantly depending on the mix of traffic types, especially in terms of security and performance. For example, voice and video performance requirements include constant bandwidth and low delay and jitter (variation in delay), whereas transactional traffic requires high reliability and security with relatively low bandwidth. Voice applications, such as IP telephony, also require high

reliability and availability because users expect to hear a “dial tone” sound when they pick up their phone in an IP network, just as they do in a traditional telephone network. Video traffic is frequently carried as IP multicast traffic, requiring multicast features to be enabled on the network. To meet these traffic requirements, converged networks use quality of service (QoS) mechanisms so that, for example, voice and video traffic are given priority over web-based traffic.

Several security strategies, such as device hardening with strict access control and authentication, intrusion protection, intrusion detection, and traffic protection with encryption, can minimize or possibly eliminate network security threats. Security is a key issue in all networks and becomes even more important in wireless networks.

Cisco IIN and SONA Framework

To accommodate today’s and tomorrow’s network requirements, the Cisco vision of the future includes the IIN, a strategy that addresses how the network is integrated with businesses and business priorities. The Cisco SONA is an architectural framework that illustrates how to build integrated systems and guides the evolution of enterprise networks toward an IIN.

Cisco IIN

The IIN encompasses the following features:

- **Integration of networked resources and information assets that have been largely unlinked**— The modern converged networks with integrated voice, video, and data require that IT departments (and other departments that were traditionally responsible for other technologies) more closely link the IT infrastructure with the network.
- **Intelligence across multiple products and infrastructure layers**— The intelligence built in to each component of the network is extended networkwide and applies end to end.
- **Active participation of the network in the delivery of services and applications**— With added intelligence, the IIN makes it possible for the network to actively manage, monitor, and optimize service and application delivery across the entire IT environment.

The IIN offers much more than basic connectivity, bandwidth for users, and access to applications. It offers an end-to-end functionality and centralized, unified control that promotes true business transparency and agility.

With the IIN, Cisco is helping organizations to address new IT challenges, such as the deployment of service-oriented architectures, web services, and virtualization (as described in the upcoming “Phase 2” bullet). The IIN technology vision offers an evolutionary approach that consists of three phases in which functionality can be added to the infrastructure as required, as follows:

- **Phase 1: Integrated transport**— Everything (data, voice, and video) consolidates onto an IP network for secure network convergence. By integrating data, voice, and video transport into a single, standards-based, modular network, organizations can simplify network management and generate enterprisewide efficiencies. Network convergence also lays the foundation for a new class of IP-enabled applications, delivered through Cisco Unified Communications solutions.

Note

Cisco Unified Communications is the name, launched in March 2006, for the entire range of what were previously known as *Cisco IP Communications* products. These include all call control, conferencing, voice-mail and messaging, customer contact, IP phone, video telephony, videoconferencing, rich media clients, and voice application products.

- **Phase 2: Integrated services**— When the network infrastructure is converged, IT resources can be pooled and shared, or virtualized, to flexibly address the changing needs of the organization. Integrated services help to unify common elements, such as storage and data center server capacity. By extending this virtualization concept to encompass server, storage, and network elements, an organization can transparently use all of its resources more efficiently. Business continuity is also

enhanced because in the event of a local systems failure, shared resources across the IIN can provide needed services.

- **Phase 3: Integrated applications—** This phase focuses on making the network application-aware so that it can optimize application performance and more efficiently deliver networked applications to users. With Application-Oriented Networking (AON) technology, Cisco has entered this third IIN phase. In addition to capabilities such as content caching, load balancing, and application-level security, the Cisco AON makes it possible for the network to simplify the application infrastructure by integrating intelligent application message handling, optimization, and security into the existing network.

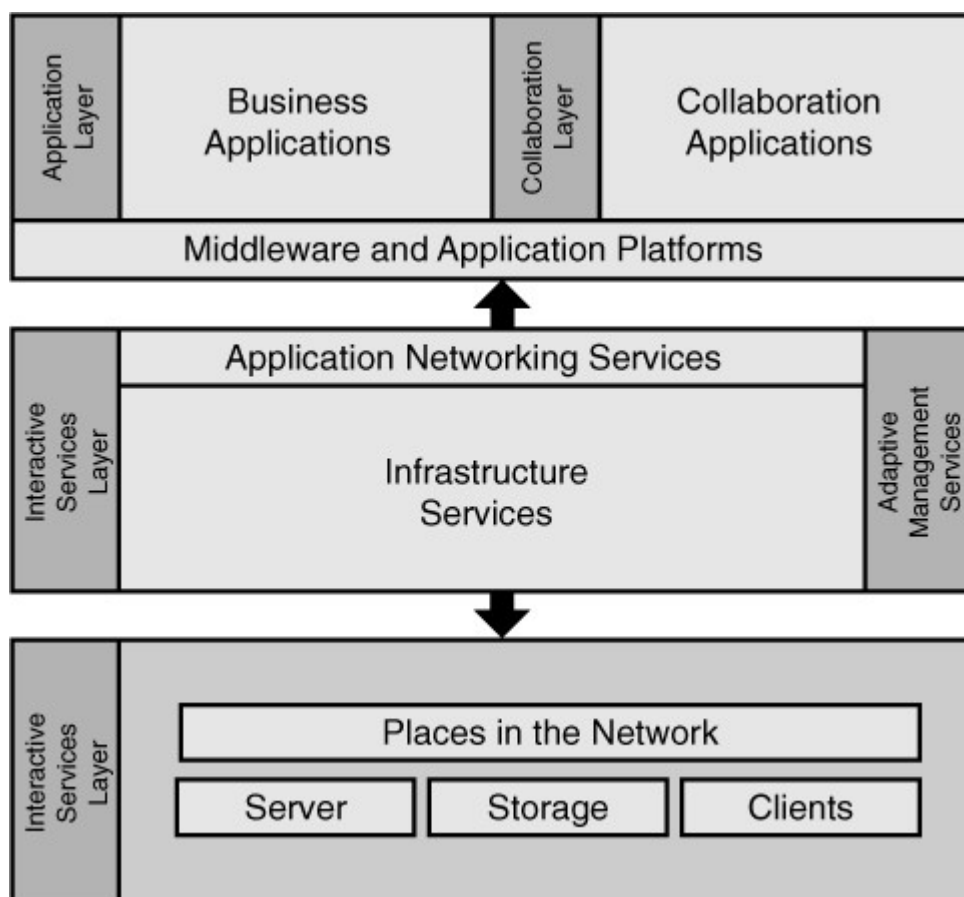
Cisco SONA Framework

The Cisco SONA architectural framework guides the evolution of enterprise networks toward an IIN. Using the SONA framework, enterprises can improve flexibility and increase efficiency by optimizing applications, business processes, and resources to enable IT to have a greater impact on business.

The SONA framework uses the extensive product-line services, proven architectures, and experience of Cisco and its partners to help enterprises achieve their business goals.

The SONA framework, shown in Figure 1-1, shows how integrated systems can allow a dynamic, flexible architecture and provide for operational efficiency through standardization and virtualization. In this framework, the network is the common element that connects and enables all components of the IT infrastructure.

Figure 1-1. Cisco SONA Framework.



The SONA framework outlines the following three layers:

- **Networked infrastructure layer**— Interconnects all the IT resources across a converged network foundation. The IT resources include servers, storage, and clients. The networked infrastructure layer represents how these resources exist in different places in the network, including the campus, branch, data center, wide-area network (WAN), metropolitan-area network (MAN), and with the teleworker. The objective of this layer is to provide connectivity, anywhere and anytime.
- **Interactive services layer**— Enables efficient allocation of resources to applications and business processes delivered through the networked infrastructure. This layer includes these services:
 - Voice and collaboration services
 - Mobility services
 - Security and identity services
 - Storage services
 - Computer services
 - Application networking services
 - Network infrastructure virtualization
 - Services management
 - Adaptive management services
- **Application layer**— Includes business applications and collaboration applications. The objective of this layer is to meet business requirements and achieve efficiencies by leveraging the interactive services layer.

Note

You can access the SONA home page at <http://www.cisco.com/go/sona>.

For example, within an organization with some remote offices, segmentation can be done to the three basic layers of SONA:

- The networked infrastructure layer represents the physical infrastructure (that is, the combination of network, servers, clients, and storage hardware that is deployed throughout an enterprise network).
- The interactive services layer represents the network-based functionality by making resources available to applications and business processes. Application delivery, real-time communication, management, mobility, security, transport, and virtualization are included in the interactive services layer.
- The application layer represents the enterprise software that addresses the needs of organizational processes and data flow, often in a distributed manner.

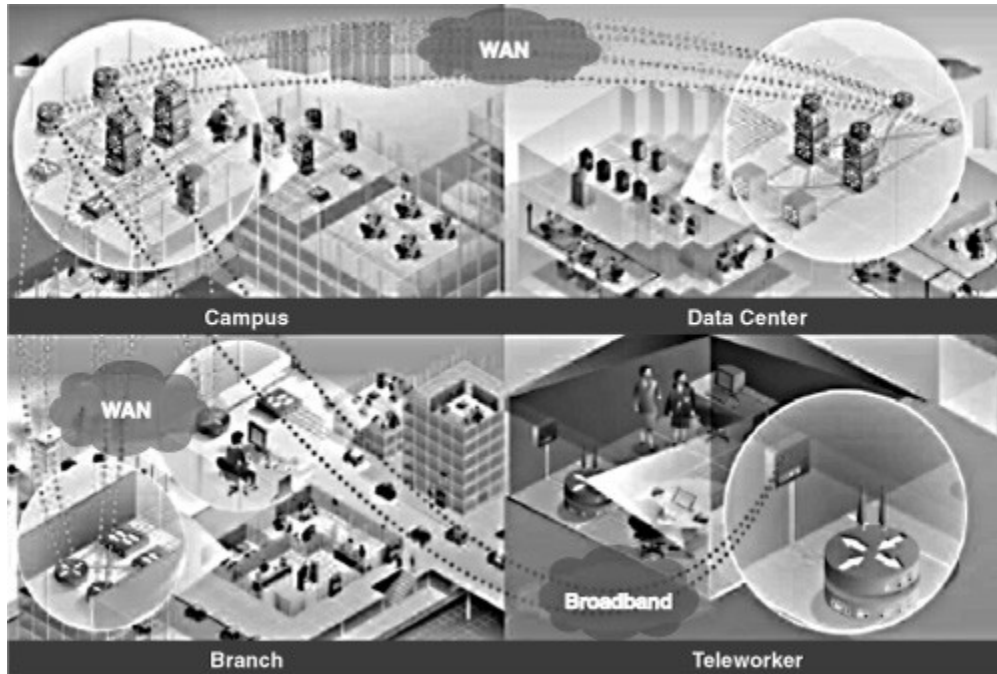
Cisco Network Models

This section describes Cisco network models, starting with the Cisco Enterprise Architecture, followed by the hierarchical network model, and concluding with the Enterprise Composite Network Model.

Cisco Enterprise Architecture

Cisco provides an enterprisewide systems architecture that helps companies to protect, optimize, and grow the infrastructure that supports their business processes. As illustrated in [Figure 1-2](#), the architecture provides for integration of the entire network—campus, data center, branches, teleworkers, and WAN—offering staff secure access to the tools, processes, and services they require.

Figure 1-2. Cisco Enterprise Architecture.



The Cisco Enterprise *Campus* Architecture combines a core infrastructure of intelligent switching and routing with tightly integrated productivity-enhancing technologies, including IP communications, mobility, and advanced security. The architecture provides the enterprise with high availability through a resilient multilayer design, redundant hardware and software features, and automatic procedures for reconfiguring network paths when failures occur. IP multicast capabilities provide optimized bandwidth consumption, and QoS features ensure that real-time traffic (such as voice, video, or critical data) is not dropped or delayed. Integrated security protects against and mitigates the impact of worms, viruses, and other attacks on the network, including at the switch port level. For example, the Cisco enterprisewide architecture extends support for security standards, such as the Institute for Electrical and Electronic Engineers (IEEE) 802.1x port-based network access control standard and the Extensible Authentication Protocol (EAP). It also provides the flexibility to add IP security (IPsec) and Multiprotocol Label Switching virtual private networks (MPLS VPNs), identity and access management, and virtual local-area networks (VLANs) to compartmentalize access. These features help improve performance and security while decreasing costs.

The Cisco Enterprise *Data Center* Architecture is a cohesive, adaptive network architecture that supports requirements for consolidation, business continuance, and security while enabling emerging service-oriented architectures, virtualization, and on-demand computing. Staff, suppliers, or customers can be provided with secure access to applications and resources, simplifying and streamlining management and significantly reducing overhead. Redundant data centers provide backup using synchronous and asynchronous data and application replication. The network and devices offer server and application load balancing to maximize performance. This architecture allows the enterprise to scale without major changes to the infrastructure.

The Cisco Enterprise *Branch* Architecture allows enterprises to extend head-office applications and services (such as security, IP communications, and advanced application performance) to thousands of remote locations and users or to a small group of branches. Cisco integrates security, switching, network analysis, caching, and converged voice and video services into a series of integrated services routers (ISRs) in the branch so that the enterprises can deploy new services without buying new routers. This architecture provides secure access to voice, mission-critical data, and video applications—anywhere, anytime. Advanced routing, VPNs, redundant WAN links, application content caching, and local IP telephony call processing features are available with high levels of resilience for all the branch offices. An optimized network uses the WAN and LAN to reduce traffic and save bandwidth and operational expenses. The enterprise can easily support branch offices with the ability to centrally configure, monitor, and manage devices located at remote sites, including tools, such as AutoQoS, which configures devices to handle congestion and bandwidth issues before they affect network performance.

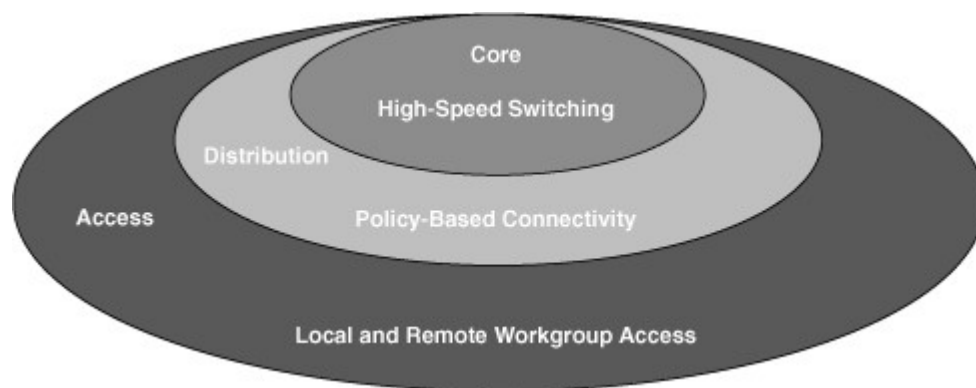
The Cisco Enterprise *Teleworker* Architecture allows enterprises to securely deliver voice and data services to remote small or home offices over a standard broadband access service, providing a business-resiliency solution for the enterprise and a flexible work environment for employees. Centralized management minimizes the IT support costs. Integrated security and identity-based networking services enable the enterprise to extend campus security policies to the teleworker. Staff can securely log in to the network over an *always-on* VPN and gain access to authorized applications and services from a single cost-effective platform. Productivity can further be enhanced by adding an IP phone, thereby providing cost-effective access to a centralized IP communications system with voice and unified messaging services.

The Cisco Enterprise WAN Architecture offers the convergence of voice, video, and data services over a single Cisco Unified Communications network, which enables the enterprise to cost-effectively span large geographic areas. QoS, granular service levels, and comprehensive encryption options help ensure the secure delivery of high-quality corporate voice, video, and data resources to all corporate sites, enabling staff to work productively and efficiently wherever they are located. Security is provided with multiservice VPNs (using IPsec and MPLS) over Layer 2 or Layer 3 WANs, hub-and-spoke, or full-mesh topologies.

Cisco Hierarchical Network Model

Traditionally, the three-layer hierarchical model, illustrated in Figure 1-3, has been used in network design, providing a modular framework that allows design flexibility and facilitates implementation and troubleshooting. The hierarchical model divides networks or modular blocks within a network into the access, distribution, and core layers. The features of the hierarchical layers are as follows:

Figure 1-3. Cisco Hierarchical Network Model.



- **Access layer**— This layer is used to grant users access to network devices. In a network campus, the access layer generally incorporates switched LAN devices with ports that provide connectivity to workstations and servers. In the WAN environment, the access layer at remote sites or at teleworkers' homes provides access to the corporate network across various WAN technologies.
- **Distribution layer**— This layer aggregates the wiring closet connections and uses switches to segment workgroups and isolate network problems in a campus environment. Similarly, the distribution layer aggregates WAN connections at the edge of the campus and provides policy-based connectivity (in other words, it implements the organization's policies).
- **Core layer (also referred to as the backbone)**— The core layer is a high-speed backbone and is designed to switch packets as fast as possible. Because the core is critical for connectivity, it must provide a high level of availability and adapt to changes quickly.

The hierarchical model can be applied to networks that include any type of connectivity, such as LANs, WANs, wireless LANs (WLANs), MANs, and VPNs.

For example, Figure 1-4 demonstrates the model applied to the enterprise campus, and Figure 1-5 demonstrates the model applied to a WAN environment.

Figure 1-4. Hierarchical Model Applied to the Enterprise Campus.

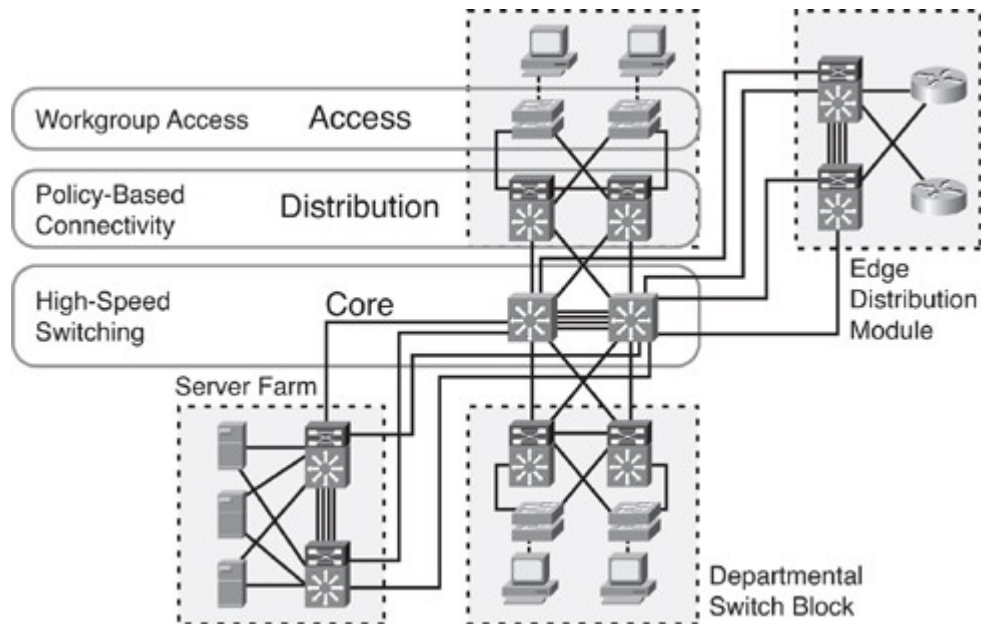
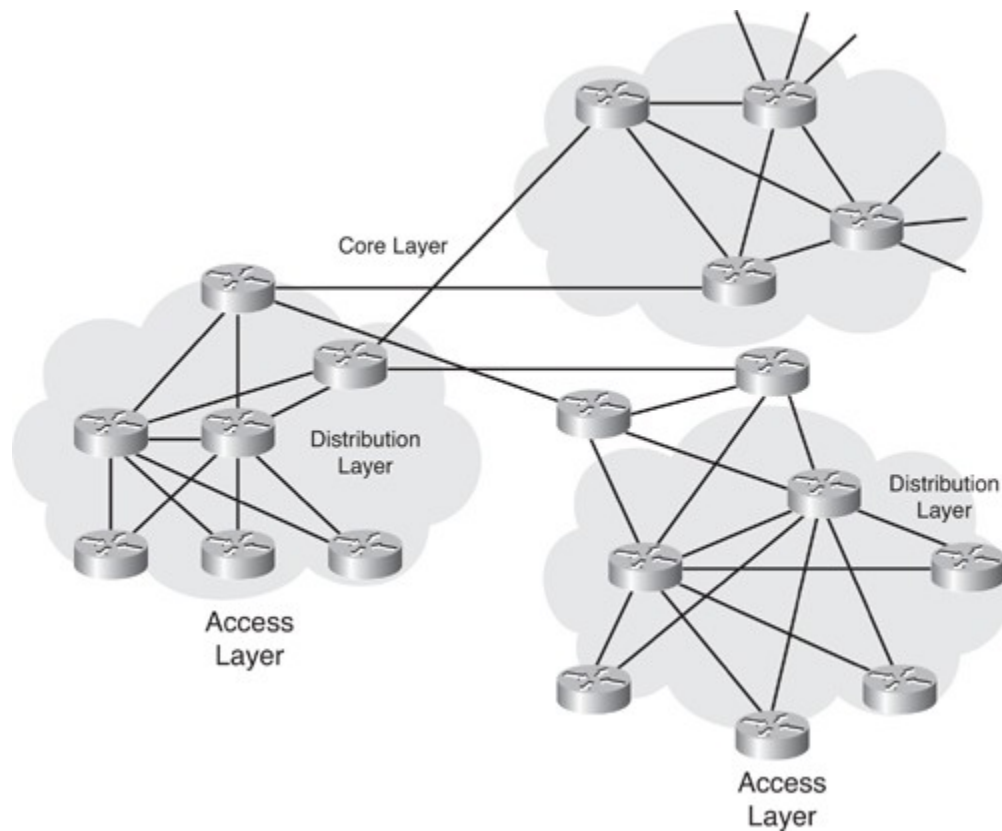


Figure 1-5. Hierarchical Model Applied to a WAN.



The hierarchical model is useful for smaller networks, but does not scale well to today's larger, more complex networks. The Enterprise Composite Network Model, introduced in the following section, provides additional modularity and functionality.

Cisco Enterprise Composite Network Model

Cisco has developed a set of best practices for security, comprising a blueprint for network designers and administrators for the proper deployment of security solutions to support network applications and the existing network infrastructure. This blueprint is called "SAFE." SAFE includes the Enterprise Composite Network Model, which network professionals can use to describe and analyze any modern enterprise network. This model supports larger networks than those designed with only the hierarchical model and clarifies the functional boundaries within the network.

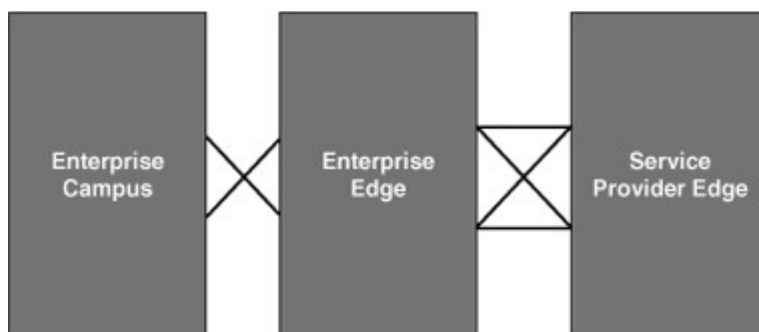
Note

You can access the SAFE blueprint home page at <http://www.cisco.com/go/safe>.

The Enterprise Composite Network Model first divides the network into three functional areas, as illustrated in Figure 1-6 and described as follows:

- **Enterprise Campus**— This functional area contains the modules required to build a hierarchical, highly robust campus network. Access, distribution, and core principles are applied to these modules appropriately.
- **Enterprise Edge**— This functional area aggregates connectivity from the various elements at the edge of the enterprise network, including to remote locations, the Internet, and remote users.
- **Service Provider Edge**— This area is not implemented by the organization; instead, it is included to represent connectivity to service providers such as Internet service providers (ISPs), WAN providers, and the public switched telephone network (PSTN).

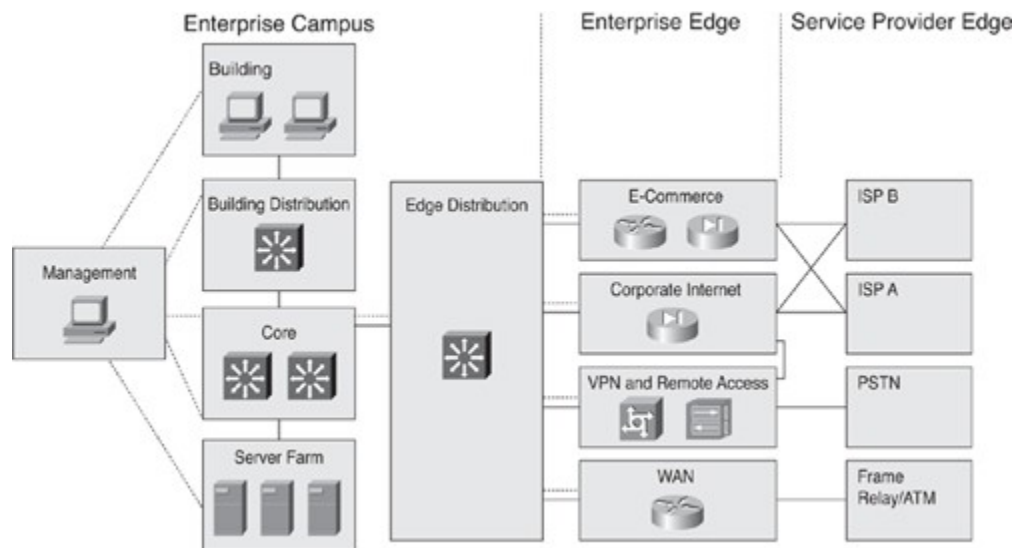
Figure 1-6. Enterprise Composite Network Model Functional Areas.



As illustrated in Figure 1-7, each of these functional areas contains various network modules. These modules can in turn include hierarchical core, distribution, and access layer functionality.

Figure 1-7. Modules Within the Enterprise Composite Network Model.

[\[View full size image\]](#)



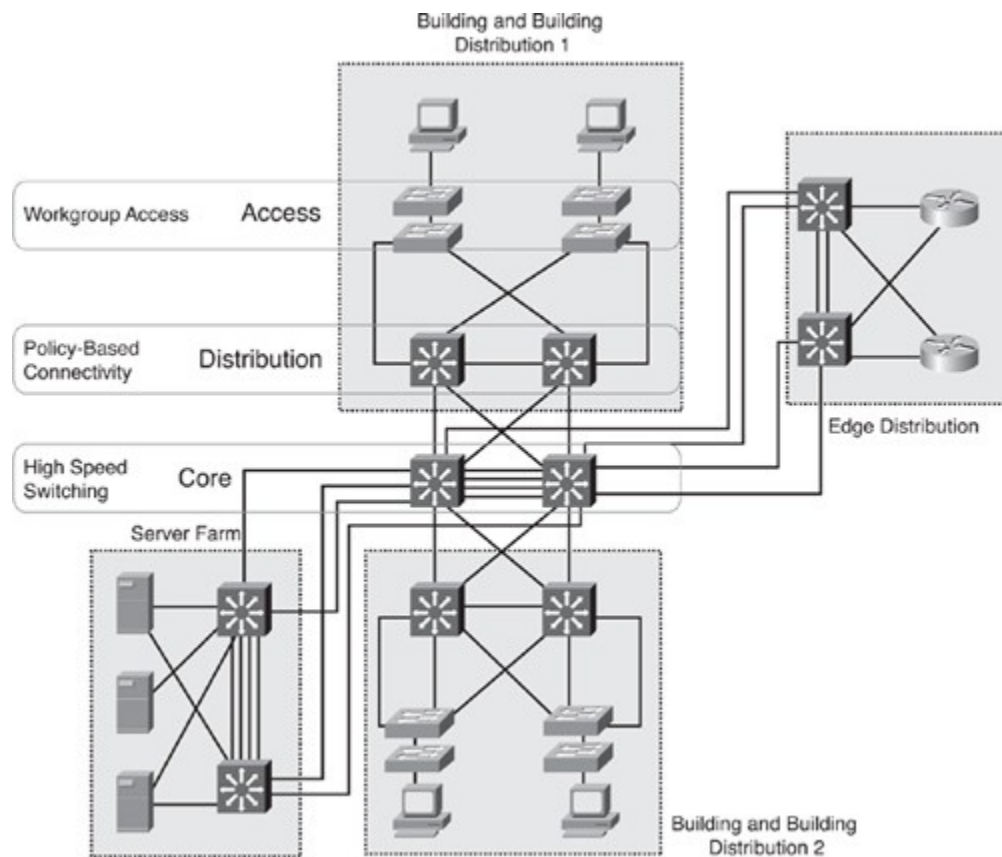
The Enterprise Campus functional area comprises the following modules:

- **Building**— Contains access switches and end-user devices (including PCs and IP phones).
- **Building Distribution**— Includes distribution multilayer switches to provide access between workgroups and to the Core.
- **Core**— Also called the backbone, provides a high-speed connection between buildings themselves, and between buildings and the Server Farm and Edge Distribution modules.
- **Edge Distribution**— The interface between the Enterprise Campus and the Enterprise Edge functional areas. This module concentrates connectivity to and from all branches and teleworkers accessing the campus via a WAN or the Internet.
- **Server Farm**— Represents the campus's data center.
- **Management**— Represents the network management functionality, including monitoring, logging, security, and other management features within an enterprise.

Figure 1-8 illustrates how the Building, Building Distribution, and Core modules map directly onto the hierarchical model's access, distribution, and core layers. The figure also shows how multiple buildings can be represented by multiple sets of a Building and a Building Distribution module, with each connected to the Core.

Figure 1-8. Multiple Buildings Represented Within the Enterprise Campus.

[\[View full size image\]](#)



The Enterprise Edge functional area is the interface between the Enterprise Campus functional area (through the Edge Distribution module) and the Service Provider Edge functional area. It is composed of the following four modules:

- **E-commerce**— Includes the servers, network devices, and so forth necessary for an organization to provide e-commerce functionality, such as online ordering
- **Corporate Internet**— Provides Internet access for the organization, and passes VPN traffic from external users to the VPN and Remote Access module
- **VPN and Remote Access**— Terminates VPN traffic and dial-in connections from external users
- **WAN**— Provides connectivity from remote sites using various WAN technologies

The three modules within the Service Provider Edge functional area are as follows:

- **ISP**— Represents Internet connections (in Figure 1-7, two instances of this module are shown, representing a dual-homed connection to two ISPs)
- **PSTN**— Represents all *nonpermanent* connections, including via analog phone, cellular phone, and Integrated Services Digital Network (ISDN)
- **Frame Relay/Asynchronous Transfer Mode (ATM)**— Represents all *permanent* connections to remote locations, including via Frame Relay, ATM, leased lines, cable, digital subscriber line (DSL), MPLS, and wireless bridging

Note

For further information and details about network design, see the Cisco Press book *Authorized Self-Study Guide: Designing for Cisco Internetwork Solutions (DESGN)*, Second Edition.

During the network design, an implementation plan is created, as detailed in the next section.

Creating, Documenting, and Executing an Implementation Plan

An effective, documented, implementation plan is a result of good processes and procedures during network design, deployment, and performance testing. This section describes approaches to creating an implementation plan and its associated documentation.

Approaches to Creating an Implementation Plan

There are two approaches to implementing changes to a network: using an ad hoc approach or using a structured approach.

In an ad hoc approach, the network engineer identifies the need for a change, such as a routing protocol implementation, and implements the solution without planning any of the tasks. The many tasks such as connectivity, addressing, routing, and security are implemented and configured as required. New equipment may be added, and new offices may be deployed. With such an approach, it is more likely that scalability issues, suboptimal routing, and security issues can occur. A good implementation plan is required to avoid such difficulties.

In a structured approach, the network engineer identifies the need for a network upgrade (for example, a new routing protocol implementation) and starts with planning as the first step. Based on the existing topology, all potential changes are reviewed, and many considerations are taken into account. The design and implementation plan are completed, and may include a new topology, an IP addressing plan, a solution to scalability issues, a link utilization upgrade, remote network connectivity, and changes to other network parameters. The design and implementation plan must meet both technical and business requirements. All details are documented in the implementation plan before the implementation. After the successful implementation, the documentation is updated to include the tools and resources used, and the implementation results.

Many models and methodologies used in IT define a lifecycle approach using various processes to help provide high-quality IT services. Network implementation, including an implementation plan, is just one part of these models. The following are some examples of these models:

- **The Cisco Lifecycle Services** approach defines the minimum set of activities needed to help customers successfully deploy and operate Cisco technologies and optimize their performance throughout the lifecycle of the network. The Cisco Lifecycle Services approach defines six phases in the network lifecycle and is referred to as the Prepare, Plan, Design, Implement, Operate, and Optimize (PPDIOO) model. The implementation plan is part of the Design phase; implementation is of course part of the Implement phase.
- **IT Infrastructure Library (ITIL)** is a framework of best practices for IT service management, providing high-quality IT services that are aligned with business requirements and processes. The implementation plan and implementation are part of ITIL best practices.
- **The Fault, Configuration, Accounting, Performance, and Security (FCAPS) model** is defined by the International Organization for Standardization (ISO) and defines the minimum set of categories needed for successful network management. Five different categories are defined: Fault Management, Configuration Management, Accounting Management, Performance Management, and Security Management. The implementation plan and implementation are part of the Configuration Management category.
- **The Telecommunications Management Network (TMN) model** is similar to the FCAPS model and defines a framework for the management of telecommunication networks. The Telecommunications Standardization Sector (ITU-T) took the main aspects of the FCAPS Model and refined it to create the TMN framework. The implementation plan and implementation are one of the building blocks within the framework.

Each organization has unique requirements. The model and its elements chosen should fit the organization, and its business and technical requirements. Different models may be combined and adapted for an optimal fit. For example, the service components, processes, and procedures needed to create an implementation plan successfully can be chosen to help ensure a successful implementation.

After the model has been selected, the cost-effective tools that support the model are chosen to allow a successful deployment of Cisco technologies with optimized performance.

After the requirements, models, and tools have been defined and collected, the implementation plan can be created.

Note

The Cisco Lifecycle Services (PPDIOO) approach is used in examples throughout this book.

Creating an Implementation Plan

As detailed in the Cisco Press book *Authorized Self-Study Guide: Designing for Cisco Internetwork Solutions (DESGN)*, Second Edition, the design methodology recommended for use with the PPDIOO model includes three basic steps:

Step 1. Identify customer requirements: In this step, which is typically completed during the PPDIOO Prepare phase, key decision makers identify the initial business and technical requirements. Based on these requirements, a high-level conceptual architecture is proposed.

Step 2. Characterize the existing network and sites: The Plan phase involves characterizing sites and assessing any existing networks, and performing a gap analysis to determine whether the existing system infrastructure, sites, and operational environment can support the proposed system. Characterization of the existing network and sites includes site and network audit and network analysis. During the network audit, the existing network is thoroughly checked for integrity and quality. During the network analysis, network behavior (traffic, congestion, and so forth) is analyzed.

Step 3. Design the network topology and solutions: In this step, the detailed design of the network is created. Decisions are made about networked infrastructure, infrastructure services, and applications. The data for making these decisions is gathered during the first two steps.

A pilot or prototype network might be constructed to verify the correctness of the design and to identify and correct any problems as a proof of concept before implementing the entire network.

A detailed design document is also written during this step; it includes information that has been documented in the previous steps.

When the design is complete, the design implementation process is executed; this process includes the following steps:

Step 1. Plan the implementation: During this step, the implementation plan is prepared in advance to expedite and clarify the actual implementation. Cost assessment is also undertaken at this time. This step is performed during the PPDIOO Design phase.

Step 2. Implement and verify the design: The actual implementation and verification of the design take place during this step by building the network. This step maps directly to the Implement phase of the PPDIOO methodology.

Step 3. Monitor and optionally redesign: The network is put into operation after it is built. During operation, the network is constantly monitored and checked for errors. If troubleshooting problems become too frequent or even impossible to manage, a network redesign might be required; this can

be avoided if all previous steps have been completed properly. This step is a part of the Operate and Optimize phases of the PPDIOO methodology.

Thus, the process of creating the implementation plan is part of the Design phase. Before developing the implementation plan, you must identify all the following information:

- **Network specific information, and the activities and tasks associated with developing the implementation plan**— The network information includes the existing topology, equipment, and software versions; the IP addressing plan; scalability requirements (summarization, stub areas, and so on); the list of advertised networks; the link utilization; and the metric requirements for primary and backup links. Other requirements to consider include site-specific implementation requirements, the tools required, and specific commands (for configuration and verification) that should be used.
- **The dependencies that your implementation plan development has on other service components and the existing network**—Implementation risks should be identified and a plan to manage them established.
- **The recommended resources to accomplish the activities and tasks associated with the implementation plan development**— The implementation schedule and the roles and responsibilities of the resources should also be established.

Based on the information gathered, the network engineer determines the implementation tasks required. For example, topology design, IP addressing, or any other changes might be required to the existing network. After the implementation plan is developed, the solution is implemented and verified, and the documentation is updated.

The following steps are completed during creation and execution of an implementation plan:

- Planning the implementation
- Selecting the tools and resources required
- Coordinating work with specialists
- Verifying the implementation
- Interpreting performance results
- Documenting the baseline, performance, and recommendations

The tasks in a site-specific implementation plan may include the following:

- Identifying applications and devices to be implemented
- Creating installation tasks and checklists
- Defining device configuration and software requirements
- Creating site-specific device configurations, installation tasks, and checklists
- Creating installation verification tests

Implementation Plan Documentation

The implementation plan documentation must be correct and up-to-date, because it is needed during both implementation and verification. After implementation, all verification steps and results should be added to the documentation so that it will be useful for troubleshooting, and for planning future upgrades and changes to the network.

The documentation must also be accessible (for example, to troubleshooting engineers). The documentation should contain all the current information about the equipment and configuration, and should include known issues, the baseline status, and the details and results of the verification tasks.

The documentation can also be used to create reports about the implementation, including the tasks performed, the schedule, the resources involved, and so forth.

The implementation plan documentation should include the following:

- Network information
- Tools required
- Resources required
- Implementation plan tasks
- Verification tasks
- Performance measurement and results
- Screen shots and photographs, as appropriate

The documentation creation process is not finished until the end of the project, when the verification information is added to it.

Use a template for an implementation plan and add information to it during every step of the process. If a standard template does not exist within the organization, create one. At the end of the project, archive the documentation so that it can be used to review, troubleshoot, and update the network in the future.

Implementation Plan Example

This section provides an example of the process of creating an implementation plan.

Example Network Scenario

In this example, an organization has an existing network that it wants to upgrade. The company wants to implement a scalable solution with a routing protocol that provides fast convergence. For more optimal routing and packet forwarding, hierarchical addressing with the summarization is required. Users require high-speed access to the server farm with redundant connectivity. The company has many remote offices, and a redundant connection to the Internet is required to provide the remote offices with nonstop access to its server farm. For remote offices, a secure connection is implemented to prevent unauthorized persons from accessing their data.

Network engineers should review the existing topology and other network information needed to implement the new solution. All requirements must be taken into account, and a complete implementation plan must be created and documented; the results of the verification tests must be added to the documentation when they are complete.

Example Network Requirements

Before the implementation plan can be created, the requirements must be defined, and the existing network characterized.

The existing topology in this example provides redundant connectivity among all the network devices. Internet connectivity is dual-homed, providing redundant access to the remote sites and World Wide Web resources. The equipment used is able to provide all the functionalities required, but the software version of the operation system must be upgraded.

The networking equipment has existing IP addressing that needs to be changed to ensure more optimal routing and forwarding of packets, and to allow summarization. The existing QoS configuration is not affected by the new requirements. The server farm hosts the company's critical applications, which along with Voice over IP (VoIP), require high priority. OSPF is currently configured in the network; the configuration must be changed to EIGRP because faster convergence time is required.

The existing configuration is sufficient to provide secure access to internal resources and remote office connectivity.

After gathering this information, all details and requirements are documented, including the following:

- A list of existing and required equipment
- The current and required software versions on the equipment
- The network topology (physical and logical)
- The design documentation
- The current configurations, including IP addressing, summarization, routing, QoS, security, and so forth
- Current link utilization and metrics
- Site-specific requirements, including IP addressing, software required, topology changes, routing protocol requirements, QoS, security, and so forth

Example Network Implementation Plan

The implementation plan can then be created, and includes the following:

- A project contact list and statements of work, to define all the people involved and their commitments to the project
- Site and equipment location information and details of how access to the premises is obtained
- Tools and resources required
- Assumptions made
- Tasks to be performed, including detailed descriptions
- Network staging plan

Table 1-1 is an example template for a project contact list; in this example the project involves Cisco personnel and customer representatives.

Table 1-1. Project Contact List

Cisco Project Team

Project Manager:
Telephone:

E-mail:

Project Engineer:
Telephone:

E-mail:

Design Engineer:
Telephone:

E-mail:

Account Manager:
Telephone:

E-mail:

Systems Engineer:
Telephone:

E-mail:

<Customer> Project Team

Project Manager:
Telephone:

E-mail:

Project Engineer:
Telephone:

E-mail:

Design Engineer:
Telephone:

E-mail:

Account Manager:
Telephone:

E-mail:

Systems Engineer:
Telephone:

E-mail:

Table 1-2 is an example template for an equipment floor plan, to provide equipment location information and access details for the premises.

Table 1-2. Equipment Floor Plan

Location	Details
Floor	
Room	
Suite	
Position	
Rack No.	

Table 1-3 is an example of a list of tools required by the implementation engineer to do the work detailed in the implementation plan.

Table 1-3. Tools Required

Item No.	Item
1.	PC with a VT100 emulator, Ethernet interface, FTP server and TFTP client applications
2.	Console port cable DB9-RJ45/DB25
3.	Ethernet cable

Table 1-4 is an example implementation task list, providing a breakdown of the steps and a detailed description of each. The output of each activity should be indicated on the implementation record.

Table 1-4. Implementation Task List

Step No.	Task
1.	Connect to the router
2.	Verify the current installation and create backup file
3.	Change IOS version (on all routers)
4.	Update IP address configuration (on distribution routers)
5.	Configure EIGRP routing protocol
6.	Verify configuration and record the results

After successful implementation, the documentation must be updated to include all the details, verification steps, and results.

Reviewing IP Routing Principles

After the design and implementation plan are complete, the implementation begins. This typically involves routing changes; routing is, of course, the focus of this book.

Note

Although much of this section will be a review for many readers, we believe it is important to include it in this book, to provide some context for the following chapters.

This section reviews IP routing, including static and dynamic routing characteristics, and on-demand routing (ODR). Routing protocol characteristics are explored, including distance vector, link-state, and advanced distance vector (also called hybrid) routing; classful and classless routing; and manual and automatic route summarization across network boundaries. Characteristics and configuration of RIP are described. A discussion of how Cisco routers populate their routing tables includes administrative distance, routing metrics, and the criteria routers use for inserting routes into the IP routing table. Comparisons of IP routing protocols are shown. The section ends with a discussion of routing protocols within the Enterprise Composite Network Model.

Note

[Appendix B](#), “IPv4 Supplement,” includes job aids and supplementary information related to IPv4 addresses that you should understand before reading the rest of the book. Therefore, before reading the rest of this chapter, you are encouraged to review any of the material in [Appendix B](#) that you are not familiar with.

[Appendix B](#) is available at this book’s companion website <http://www.ciscopress.com/title/9781587058820>.

IP Routing Overview

Routers forward packets toward destination networks. To forward the packets, routers must know about these remote networks and determine the best way to reach them. This section addresses the ways in which routers learn about networks and how routers can incorporate static and dynamic routes.

Routers must be aware of destination networks to be able to forward packets to them. A router knows about the networks directly attached to its interfaces; it calculates the subnet or network number of an interface by using the address and subnet mask configured on that interface. For networks not directly connected to one of its interfaces, however, the router must rely on outside information. A router can be made aware of remote networks in two ways:

- **Static routing**— An administrator can manually configure the information.
- **Dynamic routing**— A router can learn from other routers.

A routing table can contain both static and dynamically recognized routes. Network administrators can use static routing, dynamic routing, or a combination of both.

Principles of Static Routing

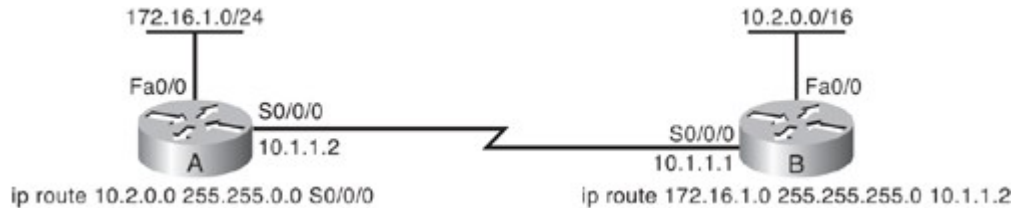
This section explains the situations in which static routes are the most appropriate to use.

A static route can be used in the following circumstances:

- When it is undesirable to have dynamic routing updates forwarded across slow bandwidth links, such as a dialup link.
- When the administrator needs total control over the routes used by the router.
- When a backup to a dynamically recognized route is necessary.

- When it is necessary to reach a network accessible by only one path (a stub network). For example, in [Figure 1-9](#), there is only one way for Router A to reach the 10.2.0.0/16 network on Router B. The administrator can configure a static route on Router A to reach the 10.2.0.0/16 network via its Serial 0/0/0 interface.

Figure 1-9. Configuring Static Routing.



- When a router connects to its ISP and needs to have only a default route pointing toward the ISP router, rather than learning many routes from the ISP.
- When a router is underpowered and does not have the CPU or memory resources necessary to handle a dynamic routing protocol.

A perfect use for static routing is a hub-and-spoke design, with all remote sites defaulting back to the central site (the hub) and the one or two routers at the central site having a static route for all subnets at each remote site. However, without proper design, as the network grows into hundreds of routers, with each router having numerous subnets, the number of static routes on each router also increases. Each time a new subnet or router is added, an administrator must add a static route to the new networks on several routers. The administrative burden to maintain this network can become excessive, making dynamic routing a better choice.

Another drawback of static routing is that when a topology change occurs on the internetwork, an administrator might have to reroute traffic by configuring new static routes around the problem area. In contrast, with dynamic routing, the routers must learn the new topology. The routers share information with each other and their routing processes automatically discover whether any alternative routes exist and reroute without administrator intervention. Because the routers mutually develop an independent agreement of what the new topology is, they are said to *converge* on what the new routes should be. A network is converged when routing tables on all routers in the network are synchronized and contain a route to all destination networks. Convergence time is the time it takes for all routers in a network to agree on the new topology. Dynamic routing provides faster convergence.

Configuring a Static Route

Use the **ip route** *prefix mask* {*address* | *interface* [*address*]} [**dhcp**] [*distance*] [**name** *next-hop-name*] [**permanent**] **track** *number*] [**tag** *tag*] global configuration command to create static routes. The parameters of this command are explained in [Table 1-5](#).

Table 1-5. ip route Command

ip route Command	Description
<i>prefix mask</i>	The IP network and subnet mask for the remote network to be entered into the IP routing table.
address	The IP address of the next hop that can be used to reach the destination network.
<i>interface</i>	The local router outbound interface to be used to reach the destination network.
dhcp	(Optional) Enables a Dynamic Host Configuration Protocol (DHCP) server to assign a static route to a default gateway (option 3).
<i>distance</i>	(Optional) The administrative distance to be assigned to

ip route Command	Description
	this route.
name <i>next-hop-name</i>	(Optional) Applies a name to the specified route.
permanent	(Optional) Specifies that the route will not be removed from the routing table even if the interface associated with the route goes down.
track <i>number</i>	(Optional) Associates a track object with this route. Valid values for the number argument range from 1 to 500.
tag <i>tag</i>	(Optional) A value that can be used as a match value in route maps.

Note

Use static routes pointing to an interface on point-to-point interfaces only, because on multiaccess interfaces the router will not know the specific address to which to send the information. (In some cases these static routes may work anyway, because of proxy Address Resolution Protocol [ARP], but the ARP overhead may result in excessive memory and CPU consumption.) On point-to-point interfaces, the information is sent to the only other device on the network.

If no dynamic routing protocol is used on a link connecting two routers, such as in [Figure 1-9](#), a static route must be configured on the routers on both sides of the link. Otherwise, the remote router will not know how to return the packet to its originator located on the other network; there will be only one-way communication.

While configuring a static route, you must specify either a next-hop IP address or an exit interface to notify the router which direction to send traffic. [Figure 1-9](#) shows both configurations. Router A recognizes the directly connected networks 172.16.1.0 and 10.1.1.0. It needs a route to the remote network 10.2.0.0. Router B knows about the directly connected networks 10.2.0.0 and 10.1.1.0; it needs a route to the remote network 172.16.1.0. Notice that on Router B, the next-hop IP address of the Router A serial interface has been used. On Router A, however, the **ip route** command specifies its own Serial 0/0/0 interface as the exit interface. If a next-hop IP address is used, it should be the IP address of the interface of the router on the other end of the link. If an exit interface is used, the local router sends data out of the specified interface to the router on the other end of its attached link. When an exit interface is specified, the router considers this to be similar to a directly connected route (as detailed in the Note following [Table 1-6](#) later in the “[Administrative Distance](#)” section).

Table 1-6. Administrative Distance of Routing Protocols

Route Source	Default Administrative Distance
Connected interface	0
Static route out an interface ^[1]	1
Static route to a next-hop address	1
EIGRP summary route	5
External BGP	20
Internal EIGRP	90
IGRP ^[2]	100
OSPF	110
IS-IS	115
RIPv1, RIPv2	120

Route Source	Default Administrative Distance
Exterior Gateway Protocol (EGP) ^[3]	140
ODR	160
External EIGRP	170
Internal BGP	200
Unreachable	255

^[1] See Note following this table for an explanation of the administrative distances of static routes.

^[2] IGRP is no longer supported, as of Cisco IOS Release 12.3. It is included in this table for completeness.

^[3] EGP is no longer supported; it is included in this table for completeness.

Configuring a Static Default Route

In some circumstances, a router does not need to recognize the details of remote networks. The router is configured to send all traffic, or all traffic for which there is not a more specific entry in the routing table, in a particular direction; this is known as a default route. Default routes are either dynamically advertised using routing protocols or statically configured.

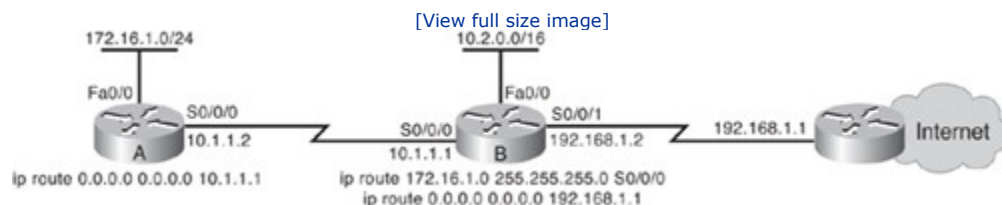
To create a static default route, use the normal **ip route** command, but with the destination network (the *prefix* in the command syntax) and its subnet mask (the *mask* in the command syntax) both set to 0.0.0.0. This address is a type of wildcard designation; any destination network will match. Because the router tries to match the longest common bit pattern, a network listed in the routing table is used before the default route. If the destination network is not listed in the routing table, the default route is used.

Note

See the “[The ip classless Command](#)” section, later in this chapter, for a discussion of scenarios where the default route might not be followed.

In [Figure 1-10](#), on Router A, the static route to the 10.2.0.0 network has been replaced with a static default route pointing to Router B. On Router B, a static default route has been added, pointing to its ISP. Traffic from a device on the Router A 172.16.1.0 network bound for a network on the Internet is sent to Router B. Router B recognizes that the destination network does not match any specific entries in its routing table and sends that traffic to the ISP. It is then the ISP’s responsibility to route that traffic to its destination.

Figure 1-10. Configuring the Static Default Route.



In [Figure 1-10](#), to reach the 172.16.1.0/24 network, Router B still needs a static route pointing out its S0/0/0 interface.

Entering the **show ip route** command on Router A in [Figure 1-10](#) returns the information shown in [Example 1-1](#).

Example 1-1. *show ip route* Command

```
RouterA#show ip route
<output omitted>
Gateway of last resort is not set
C    172.16.1.0 is directly connected, FastEthernet0/0
C    10.1.1.0 is directly connected, Serial0/0/0
S*   0.0.0.0/0 [1/0] via 10.1.1.1
```

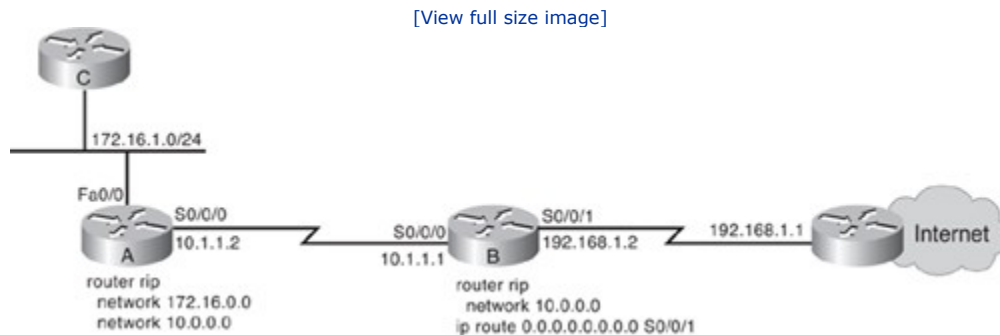
Principles of Dynamic Routing

Dynamic routing allows the network to adjust to changes in the topology automatically, without administrator involvement. This section describes dynamic routing principles.

A static route cannot respond dynamically to changes in the network. If a link fails, the static route is no longer valid if it is configured to use that failed link, so a new static route must be configured. If a new router or new link is added, that information must also be configured on every router in the network. In an unstable network, or one that has more than a few routes, these changes can lead to considerable work for network administrators. It can also take a long time for every router in the network to receive the correct information. In situations such as these, it might be better to have the routers receive information about networks and links from each other using a dynamic routing protocol.

When using a dynamic routing protocol, the administrator configures the routing protocol on each router, as shown in [Figure 1-11](#). The routers then exchange information about the reachable networks and the state of each network. Routers exchange information only with other routers running the same routing protocol. When the network topology changes, the new information is dynamically propagated throughout the network, and each router updates its routing table to reflect the changes. The following are some examples of dynamic routing protocols:

Figure 1-11. Routers Running a Dynamic Routing Protocol Exchange Routing Information.



- RIP (versions 1 and 2)
- EIGRP
- IS-IS
- OSPF
- BGP

The information exchanged by routers includes the metric to each destination (this value is sometimes called the distance or cost). A *metric* is a value that routing protocols use to measure paths to a destination.

Different routing protocols base their metric on different measurements, including hop count, interface speed, or more-complex metrics. Most routing protocols maintain databases containing all the networks that the routing protocol recognizes, all the paths to each network, and the metric of each of these paths. If a routing protocol recognizes more than one way to reach a network, it compares the metric for each different

path and chooses the path with the lowest metric. If multiple paths have the same metric, a maximum of 16 can be installed in the routing table, and the router can perform load balancing between them. EIGRP can also perform load balancing between unequal-cost paths.

Note

Before Cisco IOS Release 12.3(2)T, the maximum number of parallel routes (equal-cost paths) supported by IP routing protocols was 6; in Cisco IOS Release 12.3(2)T that maximum was changed to 16.

To configure an IP dynamic routing protocol, use the **router protocol** global configuration command. Protocols other than RIP also require specification of either an autonomous system or a process number. You also need the **network** command under the router configuration mode of all routing protocols except IS-IS and BGP.

For RIP, EIGRP, and OSPF, the **network** command tells the router which interfaces are participating in that routing protocol. Any interface that has an IP address that falls within the range specified in the **network** statement is considered active for that protocol. In other words, the router sends updates from the specified interfaces and expects to receive updates from the same interfaces. Some protocols look for neighbors by sending hello packets out those interfaces. Thus, because a **network** statement identifies interfaces on the local router, it is configured only for directly connected networks. A router also originates advertisements for the networks connected to the specified interfaces.

RIP allows only major network numbers (Class A, B, or C network numbers) to be specified in the **network** command. EIGRP and OSPF permit exact specification of interfaces with a combination of a subnet or interface address and a wildcard mask.

The **network** statement functions differently in BGP. BGP requires its neighbors to be statically configured. The **network** statement in BGP tells the router to originate an advertisement for that network. Without a **network** statement, BGP passes along advertisements it receives from other routers, but it does not originate any network advertisements itself. In BGP, the network listed in the **network** statement does not have to be directly connected, because it does not identify interfaces on the router as it does in other protocols. (This process is explained in detail in [Chapter 6](#), "Implementing a Border Gateway Protocol Solution for ISP Connectivity.")

Integrated IS-IS does not use the **network** statement. Instead, interfaces participating in the IS-IS routing process are identified under interface configuration mode. (OSPF also permits the interfaces to be specified this way, as an alternative to using the **network** command.)

[Example 1-2](#) shows the configuration of the routers in [Figure 1-11](#). Both Routers A and B are configured with RIP. Router A has two directly attached networks and RIP is used to advertise to neighbors on both of those interfaces. Therefore, **network** statements are configured for both the 172.16.0.0 network and the 10.0.0.0 network. Router A sends RIP packets out interfaces Fa0/0 and S0/0/0, advertising the networks that are attached to those interfaces.

Example 1-2. Configuring RIP

```
routerA(config)#router rip
routerA(config-router)#network 172.16.0.0
routerA(config-router)#network 10.0.0.0

routerB(config)#ip route 0.0.0.0 0.0.0.0 Serial0/0/1
routerB(config)#router rip
routerB(config-router)#network 10.0.0.0
```

Router B also has two directly attached networks. However, Router B wants only the network it shares with Router A to participate in RIP. Therefore, a **network** statement is configured only for the 10.0.0.0 network. As explained earlier, with RIP, only the major network number is actually used in the **network** command. Router B also has a static default route pointing toward its ISP to reach other networks. Router B sends RIP

packets out its interface S0/0/0, but not out its interface S0/0/1. It does not advertise the 192.168.1.0 network attached to S0/0/1 or the static default route unless specifically configured to do so.

Principles of On-Demand Routing

A drawback of static routes is that they must be manually configured and updated when the network topology changes. A drawback of dynamic routing protocols is that they use network bandwidth and router resources. In a hub-and-spoke network with hundreds of spokes, both the configuration needed for static routes and the resource usage of dynamic routing can be considerable.

There is a third option: ODR. ODR uses the Cisco Discovery Protocol (CDP) to carry network information between spoke (stub) routers and the hub router. ODR provides IP routing information with minimal overhead compared to a dynamic routing protocol and requires less manual configuration than static routes.

ODR is applicable in a hub-and-spoke topology only. In this type of topology, each spoke router is adjacent only to the hub. Another name for a spoke router is stub router. The stub router may have some LAN networks connected to it and typically has a WAN connection to the hub router. The hub router needs to recognize the networks connected to each spoke, but the spoke routers need only a default route pointing to the hub router.

When ODR is configured, the stub routers use CDP to send IP prefix information to the hub router. Stub routers send prefix information for all their directly connected networks. ODR reports the subnet mask, so it allows different subnets within the same major network to have different subnet masks. This is known as variable-length subnet masking (VLSM) and is described in detail in [Appendix B](#).

The hub router, in turn, sends a default route to the spokes that points back to itself. It installs the stub networks reported by ODR in its routing table and can be configured to redistribute these routes into a dynamic routing protocol. For a next-hop address, the hub router uses the IP address of the spoke router as reported to it by CDP.

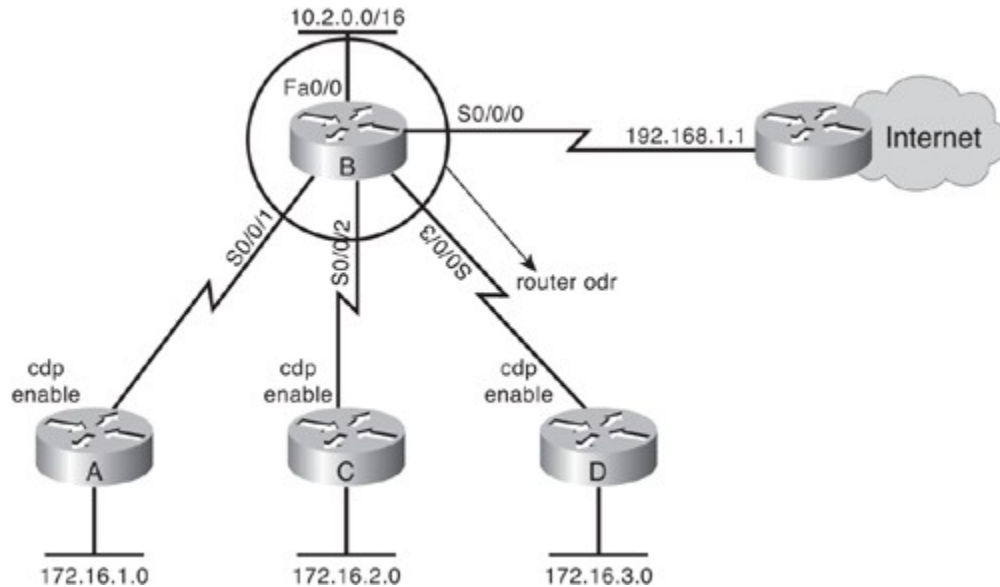
ODR is not a true routing protocol because the information exchanged is limited to IP prefixes and a default route. ODR reports no metric information; the hub router uses a hop count of 1 as the metric for all routes reported via ODR. However, by using ODR, routing information for stub networks can be obtained dynamically without the overhead of a dynamic routing protocol, and default routes can be provided to the stub routers without manual configuration.

Configuring ODR

ODR is configured on the hub router using the **router odr** global configuration command.

On the stub router, there must be no IP routing protocol configured. In fact, from the standpoint of ODR, a router is automatically considered a stub when no IP routing protocols have been configured. [Figure 1-12](#) shows a hub-and-spoke topology.

Figure 1-12. Hub-and-Spoke Topology: Configuring ODR.



ODR can also be tuned with optional commands, including using a distribute list to control the network information that is recognized through ODR, and adjusting the ODR timers with the **timers basic** router configuration command.

ODR relies on CDP to carry the information between the hub router and the spoke routers. Therefore, CDP must be enabled on the links between the hub router and the spoke routers. Cisco routers by default have CDP enabled both globally and per interface on most interfaces. However, on some WAN links, such as ATM, CDP must be explicitly enabled.

CDP updates are sent as multicasts. CDP uses Subnetwork Access Protocol (SNAP) frames, so it runs on all media that support SNAP.

CDP updates are sent every 60 seconds by default. This setting might be too infrequent in rapidly changing networks or too often in stable ones. You can adjust the timers with the **cdp timer** global configuration command. You can verify CDP settings by using the **show cdp interface** command.

As soon as ODR is configured and running, routes from the stub routers are identified in the hub router's routing table with an o character, as shown in [Example 1-3](#). Notice in the example that the metric is 1 (hop), and the administrative distance for ODR is 160. (Administrative distance is described in the "[Administrative Distance](#)" section, later in this chapter.) Also, do not confuse the o character of ODR routes with the O character of OSPF routes.

Example 1-3. Routing Table with ODR Routes

```
routerB#show ip route
<output omitted>
172.16.0.0/16 is subnetted, 4 subnets
o 172.16.1.0/24 [160/1] via 10.1.1.2, 00:00:23, Serial0/0/1
o 172.16.2.0/24 [160/1] via 10.2.2.2, 00:00:03, Serial0/0/2
o 172.16.3.0/24 [160/1] via 10.3.3.2, 00:00:16, Serial0/0/3
<output omitted>
```

The routing table for each spoke router contains only its connected networks and a static default route injected by ODR from the hub router.

Characteristics of Routing Protocols

Routing protocols can be classified into different categories such as distance vector, link-state, or advanced distance vector. IP routing protocols can also be classified as either classful or classless. These characteristics are explored in this section.

Distance Vector, Link-State, and Advanced Distance Vector Routing Protocols

When a network is using a distance vector routing protocol, all the routers periodically send their routing tables (or a portion of their tables) to only their neighboring routers. The routers then use the received information to determine whether any changes need to be made to their own routing table (for example, if a better way to a specific network is now available). This process repeats periodically.

In contrast, when a network is using a link-state routing protocol, each of the routers sends the state of its own interfaces (its links) to all other routers (or to all routers in a part of the network, known as an area) only when there is a change. Each router uses the received information to recalculate the best path to each network and then saves this information in its routing table.

As its name suggests, a hybrid or advanced distance vector protocol has characteristics of both distance vector and link-state protocols. These protocols send only changed information when there is a change (similar to link-state protocols) but only to neighboring routers (similar to distance vector protocols).

Classful Routing Protocol Concepts

IP routing protocols can be categorized as *classful* or *classless*:

- Routing updates sent by a classful routing protocol do not include the subnet mask. RIP Version 1 (RIPv1) is a classful routing protocol.
- Routing updates sent by a classless routing protocol include the subnet mask. RIP Version 2 (RIPv2), EIGRP, OSPF, IS-IS, and BGP are classless routing protocols. Most modern networks use classless protocols.

Classful Routing Protocol Behavior

When classful protocols were originally developed, networks were very different from those used now. The best modem speed was 300 bps, the largest WAN line was 56 kbps, router memory was less than 640 KB, and processors were running in the KHz range. Routing updates had to be small enough not to monopolize the WAN link bandwidth. In addition, routers did not have the resources to maintain current information about every subnet.

A classful routing protocol does not include subnet mask information in its routing updates. Because no subnet mask information is known, when a classful router receives routing updates, the router makes assumptions about the subnet mask being used by the networks listed in the update, based on IP address class.

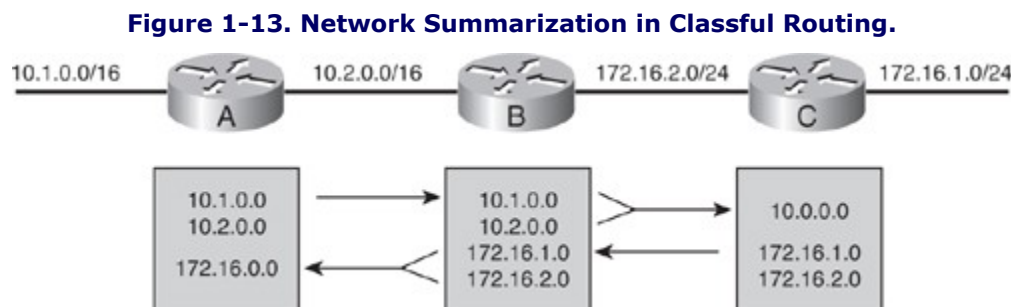
Routers send update packets from their interfaces to other connected routers. A router sends the entire subnet address in the update when an update packet involves a subnet of the same classful network as the IP address of the transmitting interface. The receiving router then assumes that the mask of the subnet in the update (from the sending router) is the same as the mask on the receiving interface. For example, if Router A sends an update about 10.1.0.0 to Router B, and Router A and B are connected by the 10.2.0.0/16 subnet, Router B assumes that the mask for the 10.1.0.0 subnet is /16, the same mask that is on the interface that receives the update. If the subnet in the update actually has a different subnet mask, the receiving router will have incorrect information in its routing table. Therefore, when using a classful routing protocol, it is important to use the same subnet mask on all subnets belonging to the same classful network; in other words, classful routing protocols do not support VLSM.

When a router that is using a classful routing protocol needs to send an update about a subnet of a network across an interface belonging to a different network, the router assumes that the remote router will use the default subnet mask for that class of IP address. Therefore, when the router sends the update, it does not include the subnet information; the update packet contains only the major (classful) network information. This process is called *autosummarization across the network boundary*; the router sends a summary of all the subnets in that network by sending only the major network information. Classful routing protocols

automatically create a classful summary route at major network boundaries. Classful routing protocols do not allow summarization at other points within the major network address space.

The router that receives the update behaves in a similar fashion. When an update contains information about a different classful network than the one in use on its interface, the router applies the default classful mask to that update. The router must assume what the subnet mask is because the update does not contain subnet mask information.

In [Figure 1-13](#), Router A advertises the 10.1.0.0 subnet to Router B because the interface connecting them belongs to the same major classful 10.0.0.0 network. When Router B receives the update packet, it assumes that the 10.1.0.0 subnet uses the same 16-bit mask as the one used on its 10.2.0.0 subnet.



Router C advertises the 172.16.1.0 subnet to Router B because the interface connecting them belongs to the same major classful 172.16.0.0 network. Therefore, Router B's routing table has information about all the subnets that are in use in the network.

However, Router B summarizes the 172.16.1.0 and 172.16.2.0 subnets to 172.16.0.0 before sending them to Router A. Therefore, Router A's routing table contains only summary information about the 172.16.0.0 network.

Similarly, Router B summarizes the 10.1.0.0 and 10.2.0.0 subnets to 10.0.0.0 before sending the routing information to Router C. This summarization occurs because the update crosses a major network boundary. The update goes from a subnet of network 10.0.0.0, subnet 10.2.0.0, to a subnet of another major network, network 172.16.0.0. Router C's routing table contains only summary information about the 10.0.0.0 network.

Summarizing Routes in a Network with Discontiguous Subnets

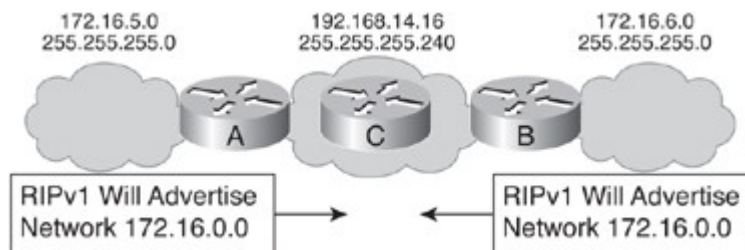
Discontiguous subnets are subnets of the same major network that are separated by a different major network.

Recall that classful protocols automatically summarize at network boundaries, which means that the following:

- Subnets are not advertised to a different major network.
- Discontiguous subnets are not visible to each other.

In [Figure 1-14](#), Routers A and B do not advertise the 172.16.5.0 255.255.255.0 and 172.16.6.0 255.255.255.0 subnets to Router C because RIPv1 cannot advertise subnets across a different major network; both Router A and Router B advertise 172.16.0.0 to Router C. This leads to confusion when routing across network 192.168.14.16/28. Router C, for example, receives routes about 172.16.0.0 from two different directions; it therefore might make an incorrect routing decision.

Figure 1-14. Classful Routing Protocols Do Not Support Discontiguous Subnets.



Although they are classless protocols, RIPv2 and EIGRP also automatically summarize at network boundaries by default. However, this feature can be turned off in RIPv2 and EIGRP. It cannot be turned off for RIPv1.

You can resolve this situation by using RIPv2, OSPF, IS-IS, or EIGRP and not using summarization, so that the subnet routes will be advertised with their actual subnet masks.

Note

For EIGRP the Cisco IOS documentation says that automatic summarization is now disabled by default. However, testing has confirmed it is still on, at least in some versions of the IOS. Therefore, it would be prudent to confirm the `autosummary` configuration or to configure it explicitly.

The `ip classless` Command

The behavior of a classful routing protocol changes when the **`ip classless`** global configuration command is used.

Note

The **`ip classless`** command is enabled by default in Release 12.0 and later of the Cisco IOS Software. In earlier releases, it is disabled by default.

When you are running a classful protocol (RIPv1), **`ip classless`** must be enabled if you want the router to use the default route when it receives a packet destined to an unknown subnet of a network for which it knows some subnets. For example, consider a router's routing table that has entries for subnets 10.5.0.0/16 and 10.6.0.0/16 and a default route of 0.0.0.0. If a packet arrives for a destination on the 10.7.0.0/16 subnet and **`ip classless`** is *not* enabled, the packet is dropped. Classful protocols assume that if they know some of the subnets of network 10.0.0.0, they must know all that network's existing subnets. Enabling **`ip classless`** tells the router that it should follow the best supernet route or the default route for unknown subnets of known networks, and for unknown networks. In this example, the router would use the default route to forward the packet for the 10.7.0.0/16 subnet.

The Routing Table Acts Classfully

The routing table itself acts classfully by default without the **`ip classless`** command and will do so even if no routing protocols are running. For example, if you have only static routes and no routing protocols, you still would not be able to reach a subnet of a known major network using a default route unless the **`ip classless`** command is enabled.

A CCIE technical reviewer of an earlier edition of this book performed the following test (using two Cisco 2520 routers running Cisco IOS c2500-i-l.122-8.T5.bin). The two routers, R1 and R2, were connected via interface E0, and no routing protocols were enabled on either router.

Router R1 configuration:

!

```

interface Loopback 0
 ip address 10.1.0.1 255.255.0.0
interface Loopback 1
 ip address 10.2.0.1 255.255.0.0
interface Ethernet 0
 ip address 10.3.0.1 255.255.0.0
!
ip route 0.0.0.0 0.0.0.0 10.3.0.2
!
no ip classless

```

Router R2 configuration:

```

!
interface Loopback 0
 ip address 10.4.0.1 255.255.0.0
interface Ethernet 0
 ip address 10.3.0.2 255.255.0.0
!

```

Test 1:

R1 has a default route pointing to R2 and has the **no ip classless** command configured. A ping from R1 to R2's loopback0 fails. When the **ip classless** command is entered on R1, the ping from R1 to R2's loopback0, via the default route, succeeds. This test proves that even though no routing protocols are used, the routing table acts classfully.

Test 2:

The second step is to test the classful nature of the routing table using a classless routing protocol, OSPF. OSPF is turned on for all interfaces on R1 but is activated only on R2's Ethernet link.

R2's OSPF is configured to inject a default route into R1 using the **default-information originate always** command (which is covered in [Chapter 3](#), "Configuring the Open Shortest Path First Protocol"). R1 therefore has a default route pointing to R2 that is introduced via OSPF. The pings from R1 to R2's loopback0 succeed regardless of the **ip classless** command. Therefore, turning on OSPF, a classless protocol, overrides the routing table's classful nature.

Classless Routing Protocol Concepts

Classless routing protocols can be considered second-generation protocols because they are designed to address some of the limitations of the earlier classful routing protocols. One of the most serious limitations in a classful network environment is that the subnet mask is not exchanged during the routing update process, and therefore, the same subnet mask must be used on all subnetworks within the same major network.

With classless routing protocols, different subnets within the same major network can have different subnet masks; in other words, they support VLSM. If more than one entry in the routing table matches a particular destination, the longest prefix match in the routing table is used. For example, if a routing table has different paths to 172.16.0.0/16 and to 172.16.5.0/24, packets addressed to 172.16.5.99 are routed through the 172.16.5.0/24 path, because that address has the longest match with the destination network.

Another limitation of the classful approach is the need to automatically summarize to the classful network boundary at major network boundaries. In a classless environment, the route summarization process can be controlled manually and can usually be invoked at any bit position within the address. Because subnet routes

might be propagated throughout the routing domain, manual route summarization might be required to keep the size of the routing tables manageable.

RIPv2 and EIGRP Automatic Network-Boundary Summarization

As mentioned earlier, by default RIPv2 and EIGRP perform automatic network summarization at classful boundaries, just like a classful protocol does. Automatic summarization lets RIPv2 and EIGRP be backward compatible with their predecessors, RIPv1 and Interior Gateway Routing Protocol (IGRP).

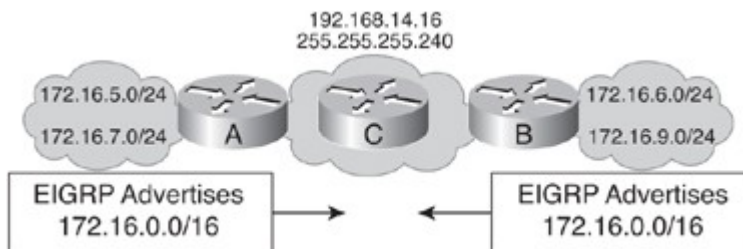
Note

IGRP is no longer supported, as of Cisco IOS Release 12.3.

The difference between these protocols and their predecessors is that you can manually turn off automatic summarization, using the **no auto-summary** router configuration command. You do not need this command when you are using OSPF or IS-IS because neither protocol performs automatic network summarization by default.

The autosummarization behavior can cause problems in a network that has discontinuous subnets or if some of the summarized subnets cannot be reached via the advertising router. If a summarized route indicates that certain subnets can be reached via a router, when in fact those subnets are discontinuous or unreachable via that router, the network might have problems similar to those caused by a classful protocol. For example, in Figure 1-15, both Router A and Router B are advertising a summarized route to 172.16.0.0/16. Router C therefore receives two routes to 172.16.0.0/16 and cannot identify which subnets are attached to which router.

Figure 1-15. Automatic Network-Boundary Summarization.

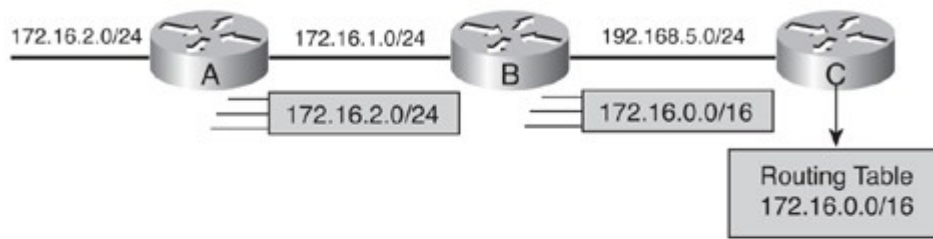


You can resolve this problem by disabling automatic summarization when running RIPv2 or EIGRP. Classless routers use the longest prefix match when selecting a route from the routing table. Therefore, if one of the routers advertises without summarizing, the other routers see subnet routes and the summary route. The other routers can then select the longest prefix match and follow the correct path. For example, in Figure 1-15, if Router A continues to summarize to 172.16.0.0/16 and Router B is configured not to summarize, Router C receives explicit routes for 172.16.6.0/24 and 172.16.9.0/24, along with the summarized route to 172.16.0.0/16. All traffic for Router B subnets is sent to Router B, and all other traffic for the 172.16.0.0 network is sent to Router A.

Another example is shown in Figure 1-16 and Figure 1-17. In the RIPv2 network illustrated in Figure 1-16, notice what routing information Router C, which is attached to Router B via the 192.168.5.0/24 network, has about network 172.16.0.0. Router B automatically summarizes the 172.16.1.0/24 and 172.16.2.0/24 subnets to 172.16.0.0/16 before sending the route to Router C, because it is sent over an interface in a different network. Instead of using the subnet mask known to Router B (/24), Router C uses this default classful mask for a Class B address (/16) when it stores the 172.16.0.0 information in its routing table.

Figure 1-16. RIPv2 Summarizes By Default; OSPF Does Not.

RIPv2 Network with Default Behavior



OSPF Network

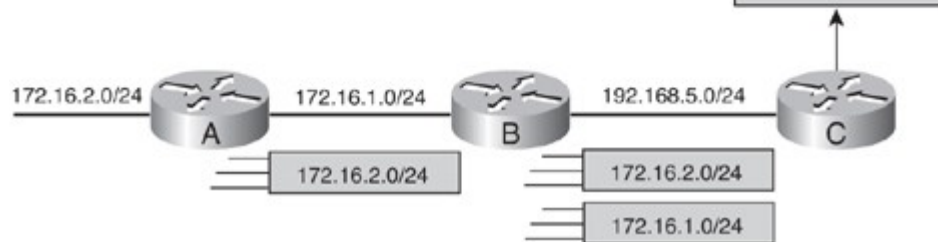
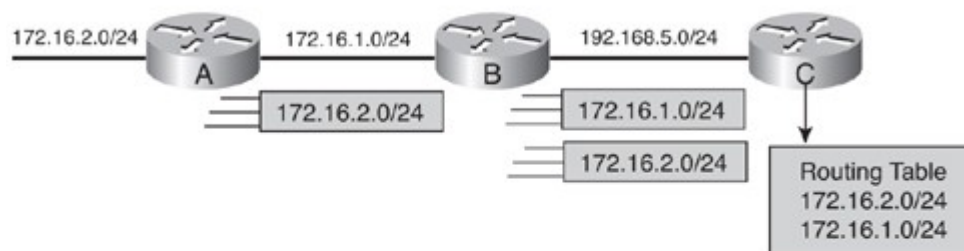
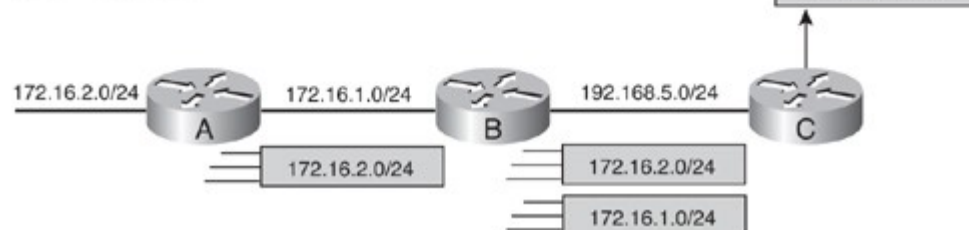


Figure 1-17. Effect of the *no auto-summary* Command for RIPv2.

RIPv2 Network with "no auto-summary"



OSPF Network



In the OSPF network shown in [Figure 1-16](#), Router B passes the subnet and subnet mask information to Router C, and Router C puts the subnet details in its routing table. Router C does not need to use default classful masks for the received routing information because the subnet mask is included in the routing update, and OSPF does not automatically summarize networks.

You can disable automatic summarization for RIPv2 and EIGRP with the **no auto-summary** router configuration command. When automatic summarization is disabled, RIPv2 and EIGRP forward subnet information, even over interfaces belonging to different major networks. In the RIPv2 network in [Figure 1-17](#), automatic summarization has been disabled. Notice that now the routing table is the same for both the RIPv2 and the OSPF routers.

Note

The BGP **auto-summary** router configuration command determines how BGP handles redistributed routes. [Chapter 5](#), “Implementing Path Control,” describes this command in detail.

RIP

This section describes the two versions of RIP—RIPv1 and RIPv2—and how to configure them. Later chapters in this book detail other routing protocols.

Characteristics of RIPv1

RIPv1 is described in RFC 1058, *Routing Information Protocol*. Its key characteristics include the following:

- Hop count is used as the metric for path selection.
- The maximum allowable hop count is 15.
- Routing updates are broadcast every 30 seconds by default. Because it is a distance vector routing protocol, updates are sent even if no change has occurred.
- RIP can load balance over as many as 16 equal-cost paths (4 paths by default).
- It has no authentication support.

Note

RFCs are available at <http://www.rfc-editor.org/rfcsearch.html>.

RIPv1 is a classful distance vector routing protocol that does not send the subnet mask in its updates. Therefore, RIPv1 does not support VLSM or discontinuous subnets. RIPv1 automatically summarizes at the network boundary and cannot be configured not to.

Characteristics of RIPv2

RIPv2 is a classless distance vector routing protocol defined in RFC 1721, *RIP Version 2 Protocol Analysis*; RFC 1722, *RIP Version 2 Protocol Applicability Statement*; and RFC 2453, *RIP Version 2*. The most significant addition to RIPv2 is the inclusion of the mask in the RIPv2 routing update packet, allowing RIPv2 to support VLSM and discontinuous subnets. RIPv2 automatically summarizes routes on classful network boundaries. As described earlier, however, you can disable this behavior.

In addition, RIPv2 uses multicast addressing for more-efficient periodic updating on each interface. RIPv2 uses the 224.0.0.9 multicast address to advertise to other RIPv2 routers. This approach is more efficient than RIPv1’s approach. RIPv1 uses a 255.255.255.255 broadcast address, so all devices, including PCs and servers, must process the update packet. They perform the checksum on the Layer 2 packet and pass it up their IP stack. IP sends the packet to the User Datagram Protocol (UDP) process, and UDP checks to see whether RIP port 520 is available. Most PCs and servers do not have any process running on this port and discard the packet.

RIP can fit up to 25 networks and subnets in each update, and updates are dispatched every 30 seconds. For example, if the routing table has 1000 subnets, 40 packets are dispatched every 30 seconds (80 packets a minute). With each packet being a broadcast for RIPv1, all devices must look at it; most of the devices discard the packet.

The IP multicast address for RIPv2 has its own multicast MAC address. Devices that can distinguish between a multicast and a broadcast at Layer 2 read the start of the frame and determine whether the destination MAC address is for them. Nonrouting devices can then discard all these packets at the interface level and not use CPU resources or buffer memory for these unwanted packets. Even on devices that cannot distinguish between broadcast and multicast at Layer 2, the worst that will happen is that the RIPv2 updates will be discarded at the IP layer instead of being passed to UDP, because those devices are not using the 224.0.0.9 multicast address.

RIPv2 also supports security between RIP routers using message-digest or clear-text authentication. (RIPv2 security features are not covered in this book.)

RIP Configuration Commands

To activate the RIP process (version 1 by default), use the **router rip** global configuration command.

By default, the Cisco IOS software processes both RIPv1 and RIPv2 packets. However, it sends only version 1 packets. To configure the software to send and receive packets from only one version, use the **version {1 | 2}** router configuration command.

To select participating attached networks, use the **network network-number** router configuration command, specifying the major classful network number. Regardless of the RIP version, at least one **network** command, using a classful network number, is required under the RIP routing process.

Although the RIP **version** command controls RIP's overall default behavior, you might need to control the version of RIP on a per-interface basis, for example when you are connecting legacy RIP networks to newer networks. To control the version of RIP on each interface, use the **ip rip {send | receive} version {1 | 2 | 1 2}** interface configuration command.

By default, automatic summarization across network boundaries is activated for all networks in both versions of RIP. Manually summarizing routes in RIPv2 improves scalability and efficiency in large networks because the more-specific routes are not advertised. Only the summary routes are advertised, thus reducing the size of the IP routing table and allowing the router to handle more routes.

Manual summarization is done at the interface. One limitation of RIPv2 is that routes can be summarized only up to the classful network boundary; RIPv2 does not support classless interdomain routing (CIDR)-type summarization to the left of the classful boundary.

Note

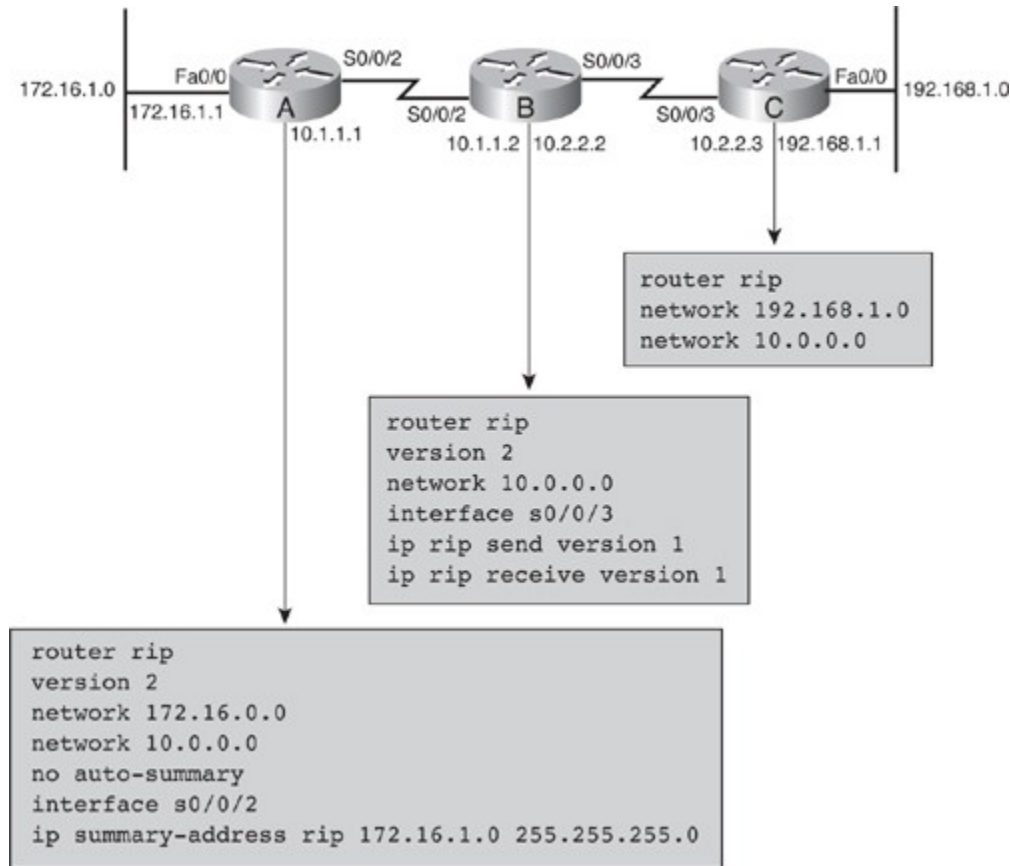
CIDR is described in [Appendix B](#).

To summarize RIP routes on nonclassful boundaries, do the following:

- Turn off automatic summarization using the **no auto-summary** router configuration command.
- Use the **ip summary-address rip network-number mask** interface configuration command to define a network number and mask that meet the particular summarization requirement.

[Figure 1-18](#) illustrates how RIPv1 and RIPv2 may coexist in the same network. Router A is running RIPv2, and Router C is running RIPv1. Router B runs both versions of RIP. Notice that the **ip rip send version 1** and **ip rip receive version 1** commands are required only on interface Serial 0/0/3 of Router B, because RIPv2 is configured as the primary version for all interfaces. The Serial 0/0/3 interface has to be manually configured to support RIPv1 so that it can connect correctly with Router C.

Figure 1-18. RIPv2 Configuration Example.



An **ip summary-address rip** command is configured on Router A along with the **no auto-summary** command. The combination of these two commands allows Router A to send the 172.16.1.0 subnet detail to Router B. Because the interface between Router A and Router B is in a different network (10.0.0.0), the default behavior for Router A is to send only the classful summarization (172.16.0.0) to Router B.

Note

In Figure 1-18, the **ip summary-address rip 172.16.1.0 255.255.255.0** command is actually unnecessary because the **no auto-summary** command is also applied. The moment that the **no auto-summary** command is used, the subnet 172.16.1.0 is advertised as such because it uses a nondefault mask (in this case, a 24-bit mask).

Commands used to verify RIP include the **show ip route** command to examine the IP routing table, and the **show ip rip database** command to display summary address entries in the RIP routing database entries if relevant child routes are being summarized.

Populating the Routing Table

This section describes how Cisco routers populate their routing tables. Administrative distance, routing metrics, and floating static routes are discussed. The criteria routers use for inserting routes into the IP routing table are described.

Administrative Distance

Most routing protocols have metric structures and algorithms that are incompatible with other protocols. It is critical that a network using multiple routing protocols be able to seamlessly exchange route information and be able to select the best path across multiple protocols. Cisco routers use a value called administrative distance to select the best path when they learn of two or more routes to the same destination with the same prefix from different routing protocols.

Administrative distance rates a routing protocol's *believability* or *trustworthiness*. Cisco has assigned a default administrative distance value to each routing protocol supported on its routers. Each routing protocol is prioritized in the order of most to least believable.

The administrative distance is a value between 0 and 255. The lower the administrative distance value, the higher the protocol's believability or trustworthiness. [Table 1-6](#) lists the default administrative distance of the protocols supported by Cisco routers.

Note

Static routes are configured with the **ip route** *prefix mask {address | interface [address]} [dhcp] [distance] [name next-hop-name] [permanent | track number] [tag tag]* global configuration command, described in the "[Principles of Static Routing](#)" section earlier in this chapter. If the *address* parameter is used in this command, specifying the address of the next-hop router to use to reach the destination network, the default administrative distance is 1. If the *interface* parameter is used instead, specifying the local router outbound interface to use to reach the destination network, the router considers this a directly connected route; however, the default administrative distance in this case appears to be somewhere between 0 and 1.

To confirm this, we performed a test. We configured two static routes to the same network, one via an address and one via an interface:

```
ip route 192.168.22.0 255.255.255.0 s1/0
ip route 192.168.22.0 255.255.255.0 192.158.2.101
```

As expected, only the one via the interface appeared in the routing table, as a "directly connected" static route:

```
S    192.168.22.0/24 is directly connected, Serial1/0
```

Therefore, its administrative distance must be less than the administrative distance of the route via the address.

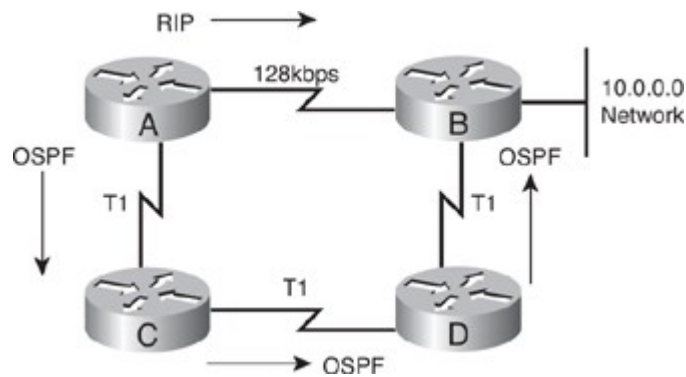
However, when we connected the same network to an interface on the router, the real connected route to the network appeared in the routing table:

```
C    192.168.22.0/24 is directly connected, FastEthernet0/0
```

Therefore, its administrative distance must be less than the administrative distance of the route via the interface. Therefore, it appears that the administrative distance of a static route via an interface has an administrative distance of something between 0 and 1.

For example, in [Figure 1-19](#), if Router A receives a route to network 10.0.0.0 from RIP and also receives a route to the same network from OSPF, the router compares RIP's administrative distance, 120, with OSPF's administrative distance, 110, and determines that OSPF is more believable. The router therefore adds the OSPF route to network 10.0.0.0 to the routing table.

Figure 1-19. Route Selection and Administrative Distance.



Routing Protocol Metrics

RIPv1 and RIPv2 use only the hop count to determine the best path (the path with the smallest hop count is preferred). Because they do not consider bandwidth, RIPv1 and RIPv2 are not suitable for networks that have significantly different transmission speeds on redundant paths. For networks that use diverse media on redundant paths, routing protocols must account for bandwidth and possibly the delay of the links.

By default EIGRP uses the minimum bandwidth and accumulated delay of the path toward the destination network in its metric calculation. Other parameters (reliability and load) can also be used, but should be configured only if the consequences are fully understood because, if misconfigured, they might affect convergence and cause routing loops. The EIGRP minimum bandwidth is the minimum (slowest) bandwidth along the path. An interface's bandwidth is either the default value of the interface or as specified by the **bandwidth** command—this command is usually used on serial interfaces.

Note

On Cisco routers, the bandwidth and delay metrics can be manually configured and do not necessarily reflect the link's true speed.

These bandwidth and delay metrics should be changed only if the consequences are well understood. For example, a bandwidth change might affect the QoS provided to data. As another example, EIGRP limits the amount of routing protocol traffic it sends to a percentage of the bandwidth value; changing the value could result in either too much bandwidth being used for routing protocol updates or updates not being sent in a timely manner.

Note

In earlier Cisco IOS releases, the default bandwidth on all serial ports was T1, or 1.544 Mbps. In the latest Cisco IOS releases, the default bandwidth varies with interface type.

In the case of link-state protocols (OSPF and IS-IS), a cumulative cost or metric is used (the lowest cost or metric path is selected). OSPF uses cost for path calculation, usually reflecting the link's bandwidth (the OSPF RFC does not specify what the cost should be, but on Cisco routers it defaults to being inversely proportional to the link's bandwidth). As a result, the highest bandwidth (lowest cost) is used to select the best path. The IS-IS interface metric defaults to 10 on Cisco routers; this value can be changed, to reflect different bandwidths, for example.

Note

The IS-IS metric is known as the *metric*; the IS-IS specification defines four different types of metrics. All routers support cost, the default metric. Delay, expense, and error are optional metrics. The default Cisco implementation of IS-IS uses cost only, but the Cisco IOS does allow all four metrics to be set with optional parameters in the **isis metric** command.

BGP uses many attributes to select the best path. One of these is the AS-path attribute; the length of this attribute is the number of autonomous systems that must be traversed to reach a destination, and is usually a factor that influences the path selection. Another attribute is the multiexit discriminator (MED). The MED attribute is called the metric in the Cisco IOS. In the output of the **show ip bgp** command for example, the MED is displayed in the *metric* column. BGP incorporates additional path attributes that can influence routing decisions; these can be manually configured.

Criteria for Inserting Routes into the IP Routing Table

A Cisco router chooses the best route for a specific destination among those presented by routing protocols, manual configuration, and various other means by considering the following four criteria:

- **Valid next-hop IP address**— As each routing process receives updates and other information, the router first verifies that the route has a valid next-hop IP address.
- **Metric**— If the next hop is valid, the routing protocol chooses the best path to any given destination based on the lowest metric. The routing protocol offers this path to the routing table. For example, if EIGRP learns of a path to 10.1.1.0/24 and decides that this particular path is the best EIGRP path to this destination, the routing protocol offers the learned path to the routing table.
- **Administrative distance**— The next consideration is administrative distance. If more than one route exists for the same network, and with the same prefix, from different routing sources, the router decides which route to install based on the administrative distance of the route's source. The route with the lowest administrative distance is installed in the routing table. Routes with higher administrative distances are rejected. For example, if both EIGRP and OSPF offered the 10.1.1.0/24 route, the EIGRP route would be installed in the routing table because EIGRP has a lower administrative distance (by default).
- **Prefix**— The router looks at the prefix being advertised. Routes to the same network but with different prefixes can coexist in the routing table. For example, suppose the router has three routing processes running on it, and the routing protocols have received and installed the following routes:
 - RIPv2: 192.168.32.0/26
 - OSPF: 192.168.32.0/24
 - EIGRP: 192.168.32.0/19

Because each route has a different prefix length (different subnet mask), the routes are considered different destinations and are all installed in the routing table. As discussed in the "[Classless Routing Protocol Concepts](#)" section, earlier in this chapter, if more than one entry in the routing table matches a particular destination, the longest prefix match in the routing table is used. Therefore, in this example, if a packet arrives for the address 192.168.32.5, the router will use the 192.168.32.0/26 subnet, advertised by RIPv2, because it is the longest match for this address.

Floating Static Routes

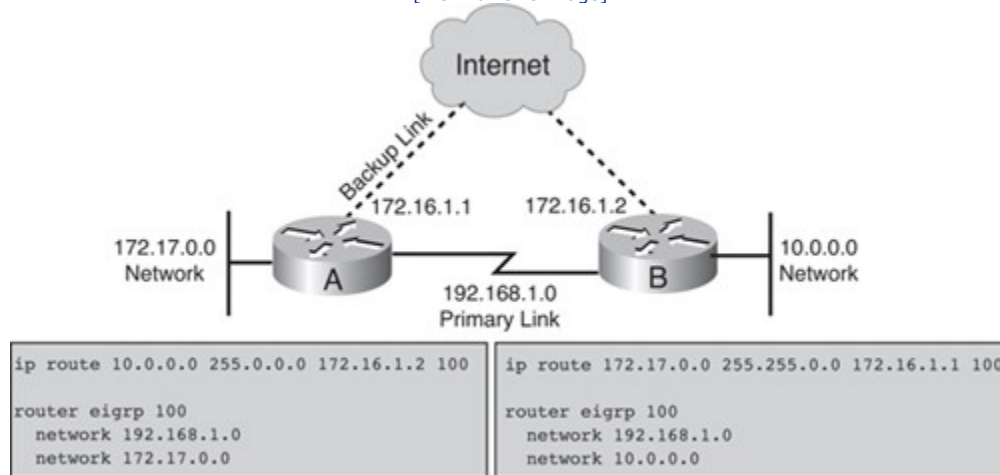
Based on the default administrative distances, routers believe static routes over any dynamically learned route. Sometimes, however, this default behavior might not be the desired behavior. For example, when you configure a static route as a backup to a dynamically learned route, you do not want the static route to be used as long as the dynamic route is available. In this case, you can manipulate the optional *distance* parameter in the **ip route** command to make the static route appear less desirable than another static or dynamic route.

A static route that appears in the routing table only when the primary route goes away is called a *floating static route*. The administrative distance of the static route is configured to be higher than the administrative distance of the primary route and it "floats" above the primary route, until the primary route is no longer available.

In [Figure 1-20](#), Routers A and B have two connections: a point-to-point serial connection that is the primary link, and a backup connection to be used if the other line goes down. Both routers use EIGRP, but do not use a routing protocol on the backup 172.16.1.0 network link.

Figure 1-20. Floating Static Routes.

[\[View full size image\]](#)



A static route that points to the backup interface of the other router has been created on each router. Because EIGRP has an administrative distance of 90, the static route has been given an administrative distance of 100. As long as Router A has an EIGRP route to the 10.0.0.0 network, it appears more believable than the static route, and the EIGRP route is used. If the serial link goes down, deleting the EIGRP route, Router A will insert the static route into the routing table. A similar process happens on Router B with its route to the 172.17.0.0 network.

IP Routing Protocol Comparisons

This section provides comparative summaries of routing protocols.

IGRP, EIGRP, and OSPF are transport layer protocols, because, like UDP and TCP, they run directly over IP. In contrast, RIP and BGP both reside at the application layer. RIP uses UDP as its transport protocol; its updates are sent unreliably with best-effort delivery. BGP uses TCP as its transport protocol; it takes advantage of TCP's reliability mechanisms and windowing. [Table 1-7](#) lists the protocol numbers, port numbers, and how reliability is handled for the various routing protocols.

Table 1-7. Protocols, Ports, and Reliability of Routing Protocols

Routing Protocol	Protocol Number	Port Number	Update Reliability
IGRP ^[1]	9	—	Best-effort delivery
EIGRP	88	—	1-to-1 window
OSPF	89	—	1-to-1 window
RIP	—	UDP 520	Best-effort delivery
BGP	—	TCP 179	Uses TCP windowing

^[1] IGRP is no longer supported, as of Cisco IOS Release 12.3. It is mentioned in this table for completeness.

Note

IS-IS is a network layer protocol and does not use the services of IP to carry its routing information. IS-IS packets are encapsulated directly into a data link layer frame and require knowledge of OSI protocol suite configuration.

Table 1-8 compares some of the characteristics of the different routing protocols.

Table 1-8. Routing Protocol Comparison

Characteristic	RIPv2	EIGRP ^[1]	IS-IS	OSPF	BGP ^[2]
Distance vector	✓	✓			✓
Link state			✓	✓	
Hierarchical topology required			✓	✓	
Automatic route summarization	✓	✓			✓
Manual route summarization	✓	✓	✓	✓	✓
VLSM support	✓	✓	✓	✓	✓
Classless	✓	✓	✓	✓	✓
Metric	Hops	Composite metric	Metric	Cost	Path attributes
Convergence time	Slow	Very fast	Fast	Fast	Slow

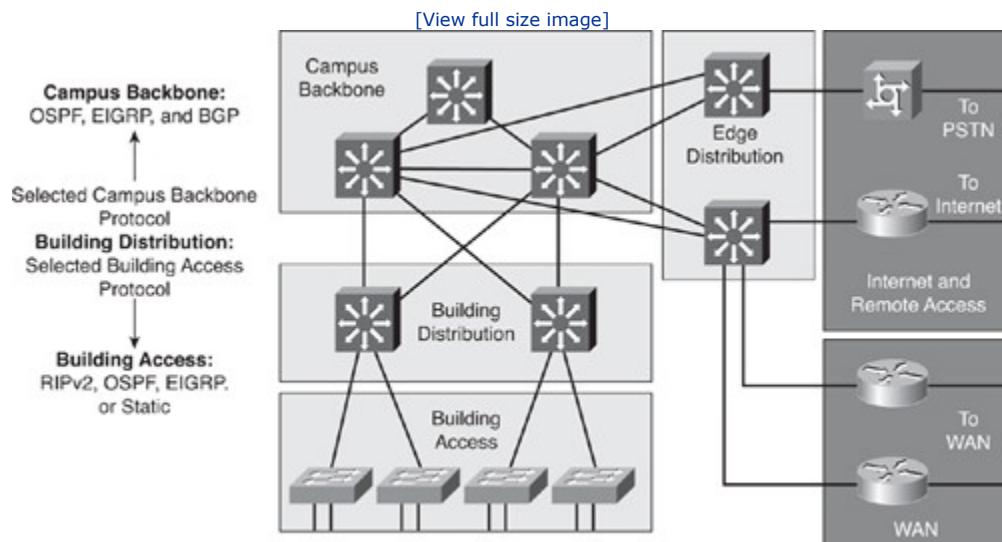
^[1] EIGRP is an advanced distance vector protocol with some characteristics also found in link-state protocols.

^[2] BGP is a path vector policy-based protocol.

Routing and Routing Protocols Within the Enterprise Composite Network Model

Routing protocols are an integral part of any network. When designing a network using the architectures and models introduced in this chapter, routing protocol selection and planning are among the design decisions to be made. Although the best practice is to use one IP routing protocol throughout the enterprise if possible, in many cases multiple routing protocols might be required, as illustrated in Figure 1-21. For example, BGP might be used in the Corporate Internet module, whereas static routes are often used for remote-access and VPN users. Therefore, enterprises might have to deal with multiple routing protocols.

Figure 1-21. Multiple Routing Protocols May Be Used Within a Network.



The Enterprise Composite Network Model can assist in determining where each routing protocol is implemented, where the boundaries between protocols are, and how traffic flows between them will be managed.

Each routing protocol has its own unique characteristics. You can use a table, like [Table 1-9](#), to identify the characteristics of the routing protocols that are being considered for a network, so that they can be compared and a decision on which to use can be made.

Table 1-9. Example Routing Protocol Comparison

Parameters	EIGRP	OSPF	BGP
Size of network (small-medium-large-very large)	Large	Large	Very large
Speed of convergence (very high-high-medium-low)	Very high	High	Slow
Use of VLSM (yes-no)	Yes	Yes	Yes
Support for mixed-vendor devices (yes-no)	No	Yes	Yes
Network support staff knowledge (good-poor)	Good	Good	Fair

Although static routes may be used (for example, for Internet connectivity) and RIPv2 is a plausible choice for smaller networks, EIGRP and OSPF are the recommended protocols within the Enterprise. BGP is required for inter-autonomous system connectivity on the Internet.

Subsequent chapters in this book cover EIGRP, OSPF, and BGP characteristics, operation, and configuration, and manipulating routing updates and traffic.

Summary

In this chapter, you learned about network models, requirements, and implementation plans, and reviewed IP routing principles. The chapter focused on the following topics:

- Traffic in converged networks, including voice and video, voice applications, mission-critical, transactional, routing protocol, and network management.
- The three phases of the Cisco IIN: integrated transport, integrated services, and integrated applications.
- The three layers of the Cisco SONA architectural framework: networked infrastructure, interactive services, application.
- The components of the Cisco Enterprise Architecture for integration of the entire network: campus, data center, branches, teleworkers, and WAN.
- The traditional hierarchical network model with its three layers: core, distribution, and access.
- The Cisco Enterprise Composite Network Model with its three functional areas and their associated modules:
 - Enterprise Campus: Building, Building Distribution, Core, Edge Distribution, Server Farm, Management
 - Enterprise Edge: E-commerce, Corporate Internet, VPN and Remote Access, WAN
 - Service Provider Edge: ISP, PSTN, Frame Relay/ATM.
- The two approaches to implementing changes to a network: using an ad hoc approach or using a structured approach.

- Four models used in IT services lifecycles: Cisco Lifecycle Services (PPDIOO), ITIL, FCAPS, and TMN.
- Creating an implementation plan, as part of the network design phase, including
 - Network information
 - Tools required
 - Resources required
 - Implementation plan tasks
 - Verification tasks
 - Performance measurement and results
- Network *convergence*, when routing tables on all routers in the network are synchronized and contain a route to all destination networks. Convergence time is the time it takes for all routers in a network to agree on the new topology.
- Static routing characteristics and configuration (using the **ip route** global configuration command).
- Characteristics and configuration (with the **router odr** global configuration command) of ODR, which uses CDP to carry network information between spoke (stub) routers and the hub router.
- Dynamic routing protocol characteristics, including
 - The metric, a value that routing protocols use to measure paths to a destination.
 - Configuration, using the **router protocol** global configuration command.
 - Distance vector routing, in which all the routers periodically send their routing tables (or a portion of their tables) to only their neighboring routers.
 - Link-state routing, in which each of the routers sends the state of its own interfaces (its links) to all other routers (or to all routers in a part of the network, known as an area) only when there is a change.
 - Advanced distance vector routing, in which routers send only changed information when there is a change (similar to link-state protocols) but only to neighboring routers (similar to distance vector protocols).
 - Classful routing protocol updates, which do not include the subnet mask. Classful protocols (such as RIPv1) do not support VLSM or discontinuous subnets and must automatically summarize across the network boundary to the classful address.
 - Classless routing protocol updates, which do include the subnet mask. Classless protocols (such as RIPv2, EIGRP, and OSPF) do support VLSM and discontinuous subnets, and do not have to summarize automatically across network boundaries.
- RIPv1 and RIPv2 characteristics and configuration, including
 - The **router rip** global configuration command to enable RIP
 - The **version {1 | 2}** router configuration command to send and receive packets from only one version
 - The **network network-number** router configuration command to configure RIP to start up the protocol on interfaces in that network, and send advertisements for the networks connected to the specified interfaces.
 - The **ip rip {send | receive} version {1 | 2 | 1 2}** interface configuration command to control the version of RIP on each interface
 - The **no auto-summary** router configuration command to turn off automatic summarization

- The **ip summary-address rip** *network-number mask* interface configuration command to define a network number and mask that meet the particular summarization requirement
- The process that Cisco routers use to populate their routing tables:
 - **Valid next-hop IP address**— The router first verifies that the route has a valid next-hop IP address.
 - **Metric**— If the next hop is valid, the routing protocol chooses the best path to any given destination based on the lowest metric and offers this path to the routing table.
 - **Administrative distance**— If more than one route exists for the same network, and with the same prefix from different routing sources, the router decides which route to install based on the administrative distance of the route's source. The administrative distance is a value between 0 and 255. The lower the administrative distance value, the higher the protocol's believability or trustworthiness. The route with the lowest administrative distance is installed in the routing table. Routes with higher administrative distances are rejected.
 - **Prefix**— The router looks at the prefix being advertised; routes to the same network but with different prefixes can coexist in the routing table.
- A comparison of the various IP routing protocols. Multiple protocols may be used within the modules of the Enterprise Composite Network Model. EIGRP and OSPF are the recommended protocols within the Enterprise; BGP is required for inter-autonomous system connectivity on the Internet.

Review Questions

Answer the following questions, and then see [Appendix A](#), "Answers to Review Questions," for the answers.

1. What is a converged network?
2. What are the three phases of the IIN?
3. Which are layers within the SONA framework?
 - a. Access
 - b. Networked Infrastructure
 - c. Interactive Services
 - d. Enterprise Edge
 - e. Application
 - f. Edge Distribution
4. What are the components of the Cisco Enterprise Architecture?
5. Which are the layers within the hierarchical network model?
 - a. Access
 - b. Network Infrastructure
 - c. Core
 - d. Distribution
 - e. Application
 - f. Edge Distribution

g. Network Management

6. Describe each of the functional areas of the Enterprise Composite Network Model.

7. Which modules are within the Enterprise Campus functional area?

8. Which of the following are steps in the structured approach to implementing network changes?

- a. Implementing the solution
- b. Documenting the implementation plan and the implementation results
- c. Troubleshooting operational issues
- d. Creating an implementation plan

9. What are the six phases in the Cisco Lifecycle Services?

10. In which phase of the Cisco Lifecycle Services is the implementation plan created?

11. Which of the following information must be identified before developing the implementation plan?

- a. Verification test results
- b. Network specific information
- c. Dependencies on the existing network
- d. Implementation reports
- e. Recommended resources

12. You are planning a routing protocol change in your network and submit your implementation plan for approval before making any changes. Which of the following should be included in this plan?

- a. Project contact list
- b. Verification steps
- c. Tools used
- d. Implementation tasks
- e. Equipment location information

13. Which of the following is not a scenario in which static routes would be used?

- a. When the administrator needs total control over the routes used by the router
- b. When a backup to a dynamically recognized route is necessary
- c. When rapid convergence is needed

14. What are two drawbacks of static routes?

- a. Reconfiguring to reflect topology changes
- b. Complex metrics
- c. Involved convergence
- d. Absence of dynamic route discovery

15. What is used by traffic for which the destination network is not specifically listed in the routing table?

- a. Dynamic area
- b. Default route
- c. Border gateway
- d. Black hole

16.The **show ip route** command usually provides information on which of the following two items?

- a. Next hop
- b. Metric
- c. CDP
- d. Hostname

17.When using dynamic routing protocols, on what does the administrator configure the routing protocol?

- a. Each area
- b. Each intermediate system
- c. Each router
- d. Each gateway of last resort

18.Which of the following is not a dynamic routing protocol?

- a. RIPv1
- b. CDP
- c. EIGRP
- d. BGP
- e. RIPv2

19.What is a metric?

- a. A standard of measurement used by routing algorithms
- b. The set of techniques used to manage network resources
- c. Interdomain routing in TCP/IP networks
- d. Services limiting the input or output transmission rate

20.Which routing protocol uses only major classful networks to determine the interfaces participating in the protocol?

- a. EIGRP
- b. RIPv1
- c. BGP
- d. OSPF

21.ODR uses what to carry network information between spoke (stub) routers and the hub?

- a. Metric
- b. BGP
- c. Convergence
- d. CDP
- e. TCP
- f. UDP

22.Which of the following is not a classification of routing protocols?

- a. Link state
- b. Default
- c. Advanced distance vector
- d. Distance vector

23.What is autosummarization?

24.True or false: Discontiguous subnets are subnets of the same major network that are separated by a different major network.

25.Classless routing protocols allow _____.

- a. QoS
- b. VLSM
- c. VPN
- d. RIP

26.What is the command to turn off autosummarization?

- a. no auto-summarization**
- b. enable classless**
- c. ip route**
- d. no auto-summary**

27.What is the OSPF default administrative distance value?

- a. 90
- b. 100
- c. 110
- d. 120

28.When a static route's administrative distance is manually configured to be higher than the default administrative distance of dynamic routing protocols, that static route is called what?

- a. Semistatic route

- b. Floating static route
- c. Semidynamic route
- d. Manual route

29. Which variables can be used to calculate metrics?

- a. Hops
- b. Convergence time
- c. Administrative distance
- d. Path attributes
- e. Cost

30. Why might a network need to have more than one routing protocol running?

Chapter 2. Configuring the Enhanced Interior Gateway Routing Protocol

This chapter introduces you to the Enhanced Interior Gateway Routing Protocol (EIGRP). This chapter covers the following topics:

Understanding EIGRP Terminology and Operation
Planning EIGRP Routing Implementations
Configuring and Verifying EIGRP
Configuring and Verifying EIGRP in an Enterprise WAN
Configuring and Verifying EIGRP Authentication
Optimizing EIGRP Implementations

In present-day and future routing environments, EIGRP offers benefits and features over historic distance vector routing protocols, such as Routing Information Protocol Version 1 (RIPv1) and Interior Gateway Routing Protocol (IGRP). These benefits include rapid convergence, lower bandwidth utilization, and multiple-routed protocol support.

Note

As of Cisco IOS Software Release 12.3, IGRP is no longer supported. It is mentioned here to provide historical context for EIGRP.

This chapter introduces EIGRP terminology and operations, and explains how to plan for, configure, and verify EIGRP. The chapter explores considerations for deploying EIGRP in enterprise WANs, and how to configure and verify EIGRP authentication. The chapter concludes with a discussion of optimizing EIGRP implementations.

Understanding EIGRP Terminology and Operation

This section introduces EIGRP, describes its capabilities, and explains its terminology. EIGRP operation, including how EIGRP's tables are created, is examined. This section also describes the Diffusing Update Algorithm (DUAL) and provides a detailed example. The EIGRP metric calculation is also described.

EIGRP Capabilities and Attributes

EIGRP is a Cisco-proprietary protocol that combines the advantages of link-state and distance vector routing protocols. EIGRP has its roots as a distance vector routing protocol and is predictable in its behavior. Like its predecessor IGRP, EIGRP is easy to configure and is adaptable to a wide variety of network topologies. What makes EIGRP an *advanced* distance vector protocol is the addition of several link-state features, such as dynamic neighbor discovery. EIGRP is an *enhanced* IGRP because of its rapid convergence and the guarantee of a loop-free topology at all times. Features of this protocol include the following:

- **Fast convergence**— EIGRP uses DUAL to achieve rapid convergence. A router running EIGRP stores its neighbors' routing tables so that it can quickly adapt to changes in the network. If no appropriate route exists in the local routing table and no appropriate backup route exists in the topology table, EIGRP queries its neighbors to discover an alternative route. These queries are propagated until an alternative route is found or until it is determined that no alternative route exists.
- **Partial updates**— EIGRP sends partial triggered updates rather than periodic updates. These updates are sent only when the path or the metric for a route changes. They contain information about only that changed link rather than the entire routing table. Propagation of these partial updates is automatically bounded so that only those routers that require the information are updated. As a result, EIGRP consumes significantly less bandwidth than IGRP. This behavior is also different from link-state protocol operation, which sends a change update to all routers within an area.
- **Multiple network layer support**— EIGRP supports IP Version 4 (IPv4), IP Version 6 (IPv6), AppleTalk, and Novell NetWare Internetwork Packet Exchange (IPX) using protocol-dependent modules that are responsible for protocol requirements specific to the network layer. EIGRP's rapid convergence and sophisticated metric offer superior performance and stability when implemented in IP, IPv6, IPX, and AppleTalk networks.

Note

In this chapter, IP refers to IPv4.

Note

Only the IP implementation of EIGRP is thoroughly covered in this chapter. EIGRP for IPv6 is covered in Chapter 8, "Implementing IPv6 in the Enterprise Network."

AppleTalk and IPX are legacy protocols. See the Cisco IOS technical documentation at Cisco.com for information about how EIGRP operates, and how to configure it, for AppleTalk and IPX.

- **Use of multicast and unicast**— For communication between routers, EIGRP uses multicast and unicast rather than broadcast. As a result, end stations are unaffected by routing updates or queries. The multicast address used for EIGRP is 224.0.0.10.

Note

EIGRP previously was called a hybrid protocol; currently the term *advanced distance vector* is typically used to describe EIGRP.

Other EIGRP features include the following:

- **Variable-length subnet masking (VLSM) support**— EIGRP is a classless routing protocol, which means that it advertises a subnet mask for each destination network. This enables EIGRP to support discontinuous subnetworks and VLSM.
- **Seamless connectivity across all data link layer protocols and topologies**— EIGRP does not require special configuration to work across any Layer 2 protocols. Other routing protocols, such as Open Shortest Path First (OSPF), require different configurations for different Layer 2 protocols, such as Ethernet and Frame Relay (as you will see in Chapter 3, "Configuring the Open Shortest Path First Protocol"). EIGRP was designed to operate effectively in both LAN and WAN environments. In multiaccess topologies, such as Ethernet, neighbor relationships (also known as neighborships) are formed and maintained using reliable multicasting. EIGRP supports all WAN topologies: dedicated links, point-to-point links, and nonbroadcast multiaccess (NBMA) topologies. EIGRP accommodates differences in media types and speeds when neighbor adjacencies form across WAN links. The amount of bandwidth that EIGRP uses on WAN links can be limited.
- **Sophisticated metric**— EIGRP uses the same algorithm for metric calculation as IGRP, but represents values in a 32-bit format, rather than IGRP's 24-bit format, to give additional granularity (thus, the EIGRP metric is the IGRP metric multiplied by 256). A significant advantage of EIGRP (and IGRP) over other protocols is its support for unequal metric load balancing that allows administrators to better distribute traffic flow in their networks.

Like most IP routing protocols, EIGRP relies on IP packets to deliver routing information.

The EIGRP routing process is a transport layer function. IP packets carrying EIGRP information have protocol number 88 in their IP header, as illustrated in Figure 2-1 (similar to how the Transmission Control Protocol [TCP] is protocol number 6 and the User Datagram Protocol [UDP] is protocol number 17).

Figure 2-1. EIGRP Is a Transport Layer Function.

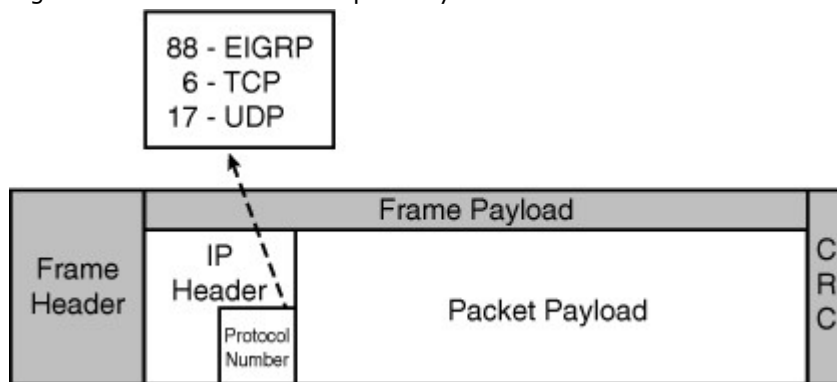
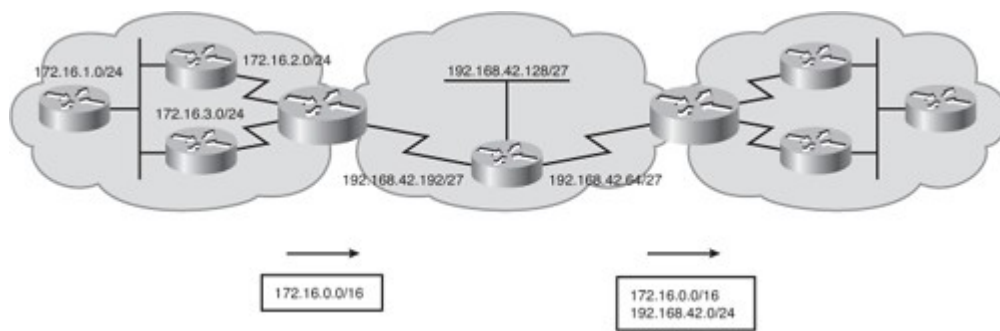


Figure 2-2 illustrates how EIGRP performs automatic route summarization at major network boundaries. You can disable this feature. Administrators can also configure manual summarization on arbitrary bit boundaries on any router interface (as long as a more-specific route exists in the routing table) to shrink the size of the routing table. EIGRP also supports the creation of supernets or aggregated blocks of addresses (networks).

Figure 2-2. EIGRP Performs Route Summarization by Default.

[View full size image]



Note

As mentioned in Chapter 1, “Routing Services,” the Cisco IOS documentation for EIGRP says that automatic summarization is now disabled by default. However, testing has confirmed it is still on, at least in some versions of the IOS. Therefore, it is prudent to confirm the automatic summarization configuration or to configure it explicitly.

EIGRP supports hierarchical addressing to enable EIGRP summarization and also supports nonhierarchical IP addressing.

EIGRP uses the following four key technologies that combine to differentiate it from other routing technologies:

- **Neighbor discovery/recovery mechanism**— EIGRP’s neighbor discovery mechanism enables routers to dynamically learn about other routers on their directly attached networks. Routers also must discover when their neighbors become unreachable or inoperative. This process is achieved with low overhead by periodically sending small hello packets. As long as a router receives hello packets from a neighboring router, it assumes that the neighbor is functioning, and the two can exchange routing information.
- **Reliable Transport Protocol (RTP)**— RTP is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. RTP supports intermixed transmission of multicast or unicast packets. For efficiency, only certain EIGRP packets are transmitted reliably.
For example, it is not necessary to send hello packets reliably to all neighbors individually, so EIGRP sends a single multicast hello packet containing an indicator that informs the receivers that the packet need not be acknowledged. Other types of packets, such as updates, indicate in the packet that acknowledgment is required. RTP contains a provision for sending multicast packets quickly even when unacknowledged packets are pending, which helps ensure that convergence time remains low in the presence of varying speed links.
- **DUAL finite-state machine**— DUAL embodies the decision process for all route computations. DUAL tracks all routes advertised by all neighbors and uses *distance* information, known as the composite metric or cost, to select efficient, loop-free paths to all destinations.
- **Protocol-dependent modules**— EIGRP’s protocol-dependent modules are responsible for network layer protocol-specific requirements. As mentioned earlier, EIGRP supports IP, IPv6, and the legacy protocols AppleTalk, and IPX. Each protocol has its own EIGRP module and operates independently from any of the others that might be running. The IP-EIGRP module, for example, is responsible for sending and receiving EIGRP packets that are encapsulated in IP. Likewise, IP-EIGRP is also responsible for parsing EIGRP packets and informing DUAL of the new information that has been received. IP-EIGRP asks DUAL to make routing decisions, the results of which are stored in the IP routing table. IP-EIGRP is also responsible for redistributing routes learned by other IP routing protocols.

EIGRP Terminology

The following terms are related to EIGRP and are used throughout the rest of this chapter:

- **Neighbor table**— EIGRP routers use hello packets to discover neighbors. When a router discovers and forms an adjacency with a new neighbor, it includes the neighbor’s address and the interface through which it can be reached in an entry in the neighbor table. This table is comparable to the neighborhood (adjacency) database used by link-state routing protocols (as described in Chapter 3). It serves the same purpose—ensuring bidirectional communication between each of the directly connected neighbors. EIGRP keeps a neighbor table for each network protocol supported. In other

words, the following tables could exist: an IP neighbor table, an IPv6 neighbor table, an IPX neighbor table, and an AppleTalk neighbor table.

- **Topology table**— When the router dynamically discovers a new neighbor, it sends an update about the routes it knows to its new neighbor and receives the same from the new neighbor. These updates populate the topology table. The topology table contains all destinations advertised by neighboring routers. In other words, each router stores its neighbors' routing tables in its EIGRP topology table. If a neighbor is advertising a destination, it must be using that route to forward packets. This rule must be strictly followed by all distance vector protocols. An EIGRP router maintains a topology table for each network protocol configured (IP, IPv6, IPX, and AppleTalk).
- **Advertised distance and feasible distance**— DUAL uses distance information, known as a metric or cost, to select efficient, loop-free paths. The lowest-cost route is calculated by adding the cost between the next-hop router and the destination—referred to as the advertised distance (AD)—to the cost between the local router and the next-hop router. The sum of these costs is referred to as the feasible distance (FD).
- **Successor**— A successor, also called a current successor, is a neighboring router that has a least-cost path to a destination (the lowest FD) that is guaranteed not to be part of a routing loop. Successor routes are offered to the routing table to be used for forwarding packets. Multiple successors can exist if they have the same FD.
- **Routing table**— The routing table holds the best routes to each destination and is used for forwarding packets. EIGRP successor routes are offered to the routing table. (As discussed in Chapter 1, if a router learns more than one route to exactly the same destination from different routing sources, it uses the administrative distance to determine which offered route to keep in the routing table.) By default, each protocol can add up to four routes to the same destination with the same metric to the routing table (recall that the router can be configured to accept up to 16 per destination). The router maintains one routing table for each network protocol configured.
- **Feasible successor**— Along with keeping least-cost paths, DUAL keeps backup paths to each destination. The next-hop router for a backup path is called the feasible successor (FS). An FS is a neighbor that is closer to the destination, but it is not in the least-cost path and, therefore, is not used to forward data. To qualify as an FS, a next-hop router must have an AD less than the FD of the current successor route. This ensures a loop-free topology. Feasible successors are selected at the same time as successors but are kept only in the topology table. The topology table can maintain multiple feasible successors for a destination.

EIGRP uses DUAL to calculate the best route to a destination. DUAL selects successor routes and FS routes based on the composite metric and ensures that the selected routes are loop-free.

EIGRP Operation

This section explains the mechanisms for creating the various EIGRP tables and describes the five types of EIGRP packets. This section also explains how EIGRP routers become neighbors, the initial route discovery process, how routes are selected, and how the DUAL algorithm functions.

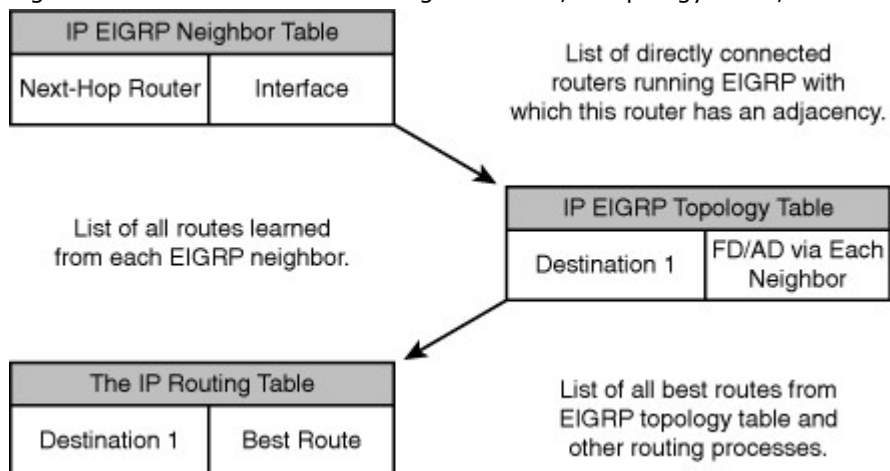
Populating EIGRP Tables

The EIGRP route selection process is perhaps what most distinguishes it from other routing protocols. EIGRP selects primary (successor) and backup (feasible successor) routes. These are marked as such in the topology table. The primary (successor) routes are then moved to the routing table.

EIGRP supports several types of routes: internal, external, and summary. Internal routes originate within the EIGRP autonomous system (AS). External routes are learned from another routing protocol or another EIGRP autonomous system. Summary routes are routes encompassing multiple subnets.

Figure 2-3 illustrates the three tables that EIGRP uses in its operation:

Figure 2-3. EIGRP Maintains a Neighbor Table, a Topology Table, and a Routing Table.



- The neighbor table lists adjacent routers.
- The topology table lists all the learned routes to each destination.
- The routing table contains the best route (the successor route) to each destination.

Neighbor Table

The neighbor table includes the address of each neighbor and the interface through which it can be reached. The neighbor table entry also includes information required by RTP. Sequence numbers are used to match acknowledgments with data packets, and the last sequence number received from the neighbor is recorded, to detect out-of-order packets. A transmission list is used to queue packets for possible retransmission on a per-neighbor basis. Round-trip timers are kept in the neighbor table entry to estimate an optimal retransmission interval.

Topology Table

Each EIGRP router forwards a copy of its IP routing table to all its adjacent EIGRP neighbors, as specified in its EIGRP neighbor table. Each router then stores the routing tables of the adjacent neighbors in its EIGRP topology table (database). The topology table also maintains the metric that each neighbor advertises for each destination (the AD) and the metric that this router would use to reach the destination via that neighbor (the FD). The **show ip eigrp topology all-links** command displays all the IP entries in the topology table, while the **show ip eigrp topology** command displays only the successors and feasible successors for IP routes.

The topology table is updated when a directly connected route or interface changes or when a neighboring router reports a change to a route.

A topology table entry for a destination can exist in one of two states: active or passive. A route is considered *passive* when the router is not performing recomputation on that route. A route is *active* when it is undergoing recomputation (in other words, when it is looking for a new successor). Note that *passive* is the operational, stable state.

If the route via the successor becomes invalid (because of a topology change) or if a neighbor is lost or changes the metric, DUAL checks for feasible successors to the destination. If an FS is found, DUAL uses it, thereby avoiding recomputing the route. This results in fast convergence. If feasible successors are always available, a destination never has to go into the active state, thereby avoiding a recomputation.

A recomputation occurs when the current route to a destination, the successor, goes down and there are no feasible successors for the destination. Although recomputation is not processor intensive, it does affect convergence time, so it is advantageous to avoid unnecessary recomputations. The router starts the recomputation by sending a query packet to each of its neighboring routers. If the neighboring router has a route for the destination, it will send a reply packet. If it does not have a route, it sends a query packet to its neighbors. In this case, the route is also in the active state in the neighboring router; while a destination is in the active state, a router cannot change the routing table information for the destination.

This process continues. Routers that have no other neighbors and routers that know that the destination is unreachable reply to queries immediately, indicating the route is unreachable. When a router receives all replies to its query, it replies to queries from its neighbor, and so on. If there is at least one topology table entry for the destination after a router has received a reply from each neighboring router, the destination returns to the passive state, and the router selects a successor for it. If there are no topology table entries for the destination, it is no longer reachable from that router. (The "Stuck-in-Active Connections in EIGRP")

section, later in this chapter, describes what happens if not all replies are received for a query in a timely manner.)

Routing Table

Each router examines its EIGRP topology table and determines the best route and other feasible routes to every destination network. A router compares all FDs to reach a specific network and then selects the route with the lowest FD and places it in the IP routing table. This is the successor route. The FD for the chosen successor route becomes the EIGRP routing metric to reach that network in the routing table.

EIGRP Packets

EIGRP sends out five different types of packets: hello, update, query, reply, and acknowledge (ACK). These packets are used to establish the initial adjacency between neighbors and to keep the topology and routing tables current. When troubleshooting an EIGRP network, network administrators must understand what the EIGRP packets are used for and how they are exchanged. For example, if routers running EIGRP do not form neighbor relationships, those routers cannot exchange EIGRP updates with each other. Without EIGRP routing updates, users cannot connect to services across the internetwork.

EIGRP uses the following five types of packets:

- **Hello**— Hello packets are used for neighbor discovery. They are sent as multicasts and do not require an acknowledgment. (They carry an acknowledgment number of 0.)
- **Update**— Update packets contain route change information. An update is sent to communicate the routes that a particular router has used to converge. An update is sent only to affected routers. Update packets are sent as multicasts when a new route is discovered, and when convergence is complete (in other words, when a route becomes passive). To synchronize topology tables, update packets are sent as unicasts to neighbors during their EIGRP startup sequence. Update packets are sent reliably.
- **Query**— When a router is performing route computation and does not have an FS, it sends a query packet to its neighbors, asking whether they have a successor to the destination. Queries are normally multicast but can be retransmitted as unicast packets in certain cases. They are sent reliably.
- **Reply**— A reply packet is sent in response to a query packet. Replies are unicast to the originator of the query and are sent reliably. A router must reply to all queries.
- **ACK**— The ACK is used to acknowledge updates, queries, and replies. ACK packets are unicast hello packets and contain a nonzero acknowledgment number. (Note that hello and ACK packets do not require acknowledgment.)

The hello packet is the first type exchanged by EIGRP routers. The following section provides details of the hello protocol and how hello packets are used. The details of how the other packet types are used are provided throughout the rest of the chapter.

EIGRP Hello Packets

Through the hello protocol, an EIGRP router dynamically discovers other EIGRP routers directly connected to it. The router sends hello packets out of interfaces configured for EIGRP using the EIGRP multicast address 224.0.0.10. When an EIGRP router receives a hello packet from a router belonging to the same autonomous system, it establishes a neighbor relationship (adjacency).

Note

The term autonomous system as used by EIGRP (and OSPF) is not the same as a Border Gateway Protocol (BGP) autonomous system (as covered in Chapter 6, "Implementing a Border Gateway Protocol Solution for ISP Connectivity"). For EIGRP, consider the autonomous system to be a group of routers all running the same protocol. You may have more than one EIGRP autonomous system (group) within your network, in which case you might want to redistribute (share) routes between them. Redistribution is detailed in Chapter 4, "Manipulating Routing Updates."

The time interval of hello packets varies depending on the medium. By default, hello packets are sent every 60 seconds on T1 or slower NBMA interfaces and every 5 seconds on other serial interfaces and on LANs.

Note

The default of 60 seconds applies only to low-speed, NBMA media. Low speed is considered to be a rate of T1 or slower, as specified with the **bandwidth** interface configuration command. For the purposes of EIGRP, Frame Relay and Switched Multimegabit Data Service (SMDS) networks may be considered to be NBMA if the interface has not been configured to use physical multicasting. Otherwise, they are considered not to be NBMA.

You can adjust the rate at which hello packets are sent, called the *hello interval*, on a per-interface basis with the **ip hello-interval eigrp as-number seconds** interface configuration command.

Hello packets include the hold time. The hold time is the amount of time a router considers a neighbor up without receiving a hello or some other EIGRP packet from that neighbor. The hold-time interval is set by default to three times the hello interval. Therefore, the default hold-time value is 15 seconds on LAN and fast WAN interfaces and 180 seconds on slower WAN interfaces. You can adjust the hold time with the **ip hold-time eigrp as-number seconds** interface configuration command.

Note

The hold time is not automatically adjusted after a hello interval change. If you change the hello interval, you must manually adjust the hold time to reflect the configured hello interval.

If a packet is not received before the expiration of the hold time, the neighbor adjacency is deleted, and all topology table entries learned from that neighbor are removed, as if the neighbor had sent an update stating that all the routes are unreachable. If the neighbor is a successor for any destination networks, those networks are removed from the routing table, and alternative paths, if available, are computed. This lets the routes quickly reconverge if an alternative feasible route is available.

EIGRP Neighbors

Two routers can become EIGRP neighbors even though the hello and hold time values do not match. This means that the hello interval and hold-time values can be set independently on different routers.

Secondary addresses can be applied to interfaces to solve particular addressing issues, although all routing overhead traffic is generated through the primary interface address. EIGRP will not build peer relationships over secondary addresses because all EIGRP traffic uses the interface's primary address. To form an EIGRP adjacency, all neighbors use their primary address as the source IP address of their EIGRP packets.

Adjacency between EIGRP routers takes place if the primary address of each neighbor is part of the same IP subnet. In addition, peer relationships are not formed if the neighbor resides in a different EIGRP autonomous system or if the metric-calculation mechanism constants (the K values) are misaligned on that link. (K values are discussed in the "EIGRP Metric Calculation" section, later in this chapter.)

Neighbor Table Contents

An EIGRP router multicasts hello packets to discover neighbors. It forms an adjacency with these neighbors so that it can exchange route updates. Only adjacent routers exchange routing information. Each router builds a neighbor table from the hello packets it receives from adjacent EIGRP routers running the same network layer protocol. EIGRP maintains a neighbor table for each configured network-layer protocol. You can display the IP neighbor table with the `show ip eigrp neighbors` command, as shown in Example 2-1.

Example 2-1. Sample Output for the `show ip eigrp neighbors` Command

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H  Address          Interface  Hold  Uptime  SRTT  RTO  Q  Seq
                               (sec)    (ms)    Cnt  Num
0  192.168.1.102     Se0/0/1   10    00:07:22  10    2280  0  5
R1#
```

This neighbor table includes the following key elements:

- **H (handle)**— A number used internally by the Cisco IOS to track a neighbor. This column lists the order in which a peering session was established with the specified neighbor, starting with 0.
- **Interface**— The interface on this router through which it is receiving hello packets for the neighbor, and therefore through which the neighbor can be reached.
- **Hold Time**— The maximum time, in seconds, that the router waits to hear from the neighbor without receiving anything from a neighbor before considering the link unavailable. Originally, the expected packet was a hello packet, but in current Cisco IOS software releases, any EIGRP packets received after the first hello from that neighbor resets the timer.
- **Uptime**— The elapsed time, in hours, minutes, and seconds since the local router first heard from this neighbor.
- **Smooth Round Trip Timer (SRTT)**— The average number of milliseconds it takes for an EIGRP packet to be sent to this neighbor and for the local router to receive an acknowledgment of that packet. This timer is used to determine the retransmit interval, also known as the retransmit timeout (RTO).

- **RTO**— The amount of time, in milliseconds, that the router waits for an acknowledgment before retransmitting a reliable packet from the retransmission queue to a neighbor.
- **Queue (Q) count**— The number of EIGRP packets (update, query, and reply) waiting in the queue to be sent out. If this value is constantly higher than 0, a congestion problem might exist. A 0 indicates that no EIGRP packets are in the queue.
- **Seq Num**— The sequence number of the last update, query, or reply packet that was received from this neighbor.

EIGRP Reliability

EIGRP's reliability mechanism ensures delivery of critical route information to neighboring routers. This information is required to allow EIGRP to maintain a loop-free topology. For efficiency, only certain EIGRP packets are transmitted reliably.

All packets carrying routing information (update, query, and reply) are sent reliably (because they are not sent periodically). A sequence number is assigned to each reliable packet and an explicit acknowledgment is required for that sequence number.

Recall that RTP is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. RTP supports an intermixed transmission of multicast and unicast packets.

RTP ensures that ongoing communication is maintained between neighboring routers. As such, a retransmission list is maintained for each neighbor. This list indicates packets not yet acknowledged by a neighbor within the RTO. It is used to track all the reliable packets that were sent but not acknowledged.

If the RTO expires before an ACK packet is received, the EIGRP process retransmits another copy of the reliable packet, up to a maximum of 16 times or until the hold time expires.

The use of reliable multicast packets is efficient. However, a potential delay exists on multiaccess media where multiple neighbors reside. The next reliable multicast packet cannot be transmitted until all peers have acknowledged the previous multicast. If one or more peers are slow to respond, this adversely affects all peers by delaying the next transmission. RTP is designed to handle such exceptions: Neighbors that are slow to respond to multicasts have the unacknowledged multicast packets retransmitted as unicasts. This allows the reliable multicast operation to proceed without delaying communication with other peers, helping to ensure that convergence time remains low in the presence of variable-speed links.

The multicast flow timer (seen in the **show ip eigrp interfaces** command output) determines how long to wait for an ACK packet before switching from multicast to unicast. The RTO determines how long to wait between the subsequent unicasts. The EIGRP process for each neighbor calculates both the multicast flow timer and RTO, based on the SRTT. The formulas for the SRTT, RTO, and multicast flow timer are Cisco-proprietary.

In a steady-state network where no routes are flapping, EIGRP waits the specified hold-time interval before it determines that an EIGRP neighbor adjacency is down. Therefore, by default, EIGRP waits up to 15 seconds on high-speed links and up to 180 seconds on slower WAN links. When EIGRP determines that a neighbor is down and the router cannot reestablish the adjacency, the routing table removes all networks that could be reached through that neighbor. The router attempts to find alternative routes to those networks so that convergence can occur.

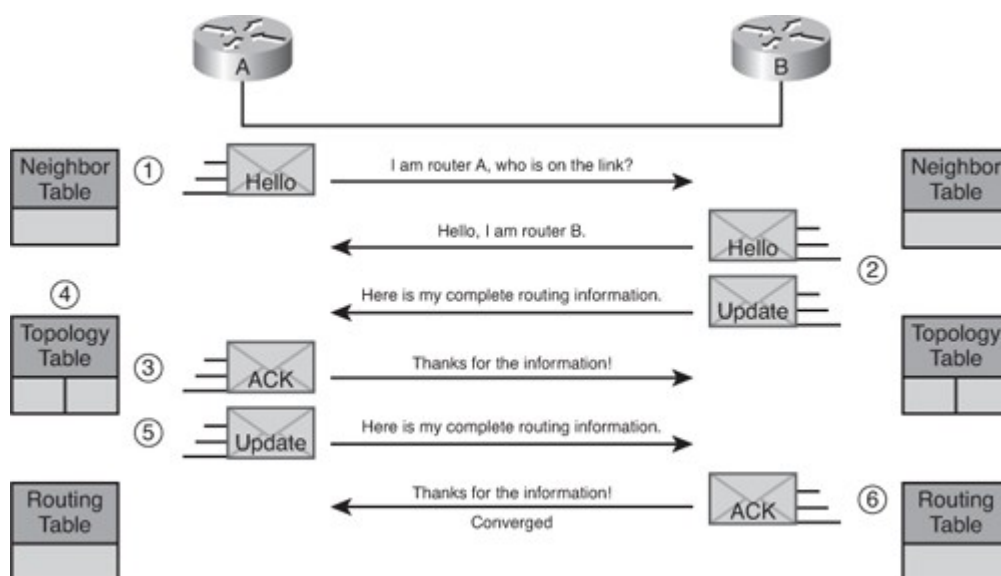
The 180-second hold time on low-speed links can seem excessive, but it accommodates the slowest-speed links, which are generally connected to less-critical remote sites. In some networks with mission-critical or time-sensitive applications (such as IP telephony), even on high-speed links, 15 seconds is too long. The point to remember is that other conditions can override the hold time and allow the network to converge quickly. For example, if the network is unstable and routes are flapping elsewhere because a remote site is timing out on its adjacency, EIGRP hold timers begin counting down from 180 seconds. When the upstream site sends the remote site an update, and the remote site does not acknowledge the update, the upstream site attempts 16 times to retransmit the update. The retransmission occurs each time the RTO expires. After 16 retries, the router resets the neighbor relationship. This causes the network to converge faster than waiting for the hold time to expire.

Initial Route Discovery

EIGRP combines the process of discovering neighbors and learning routes. Figure 2-4 illustrates the initial route discovery process.

Figure 2-4. Initial Route Discovery.

[View full size image]



The following describes the initial route discovery process:

1. A new router (Router A in Figure 2-4) comes up on the link and sends out a hello packet through all of its EIGRP-configured interfaces.
2. Routers receiving the hello packet on an interface (Router B in Figure 2-4) reply with update packets that contain all the routes they have in their routing table, except those learned through that interface (because of the split horizon rule). Router B sends an update packet to Router A, but a neighbor relationship is not established until Router B sends a hello packet to Router A. The update packet from Router B has the initial bit set, indicating that this is the initialization process. The update packet contains information about the routes that the neighbor (Router B) is aware of, including the metric that the neighbor is advertising for each destination.
3. After both routers have exchanged hellos and the neighbor adjacency is established, Router A replies to Router B with an ACK packet, indicating that it received the update information.
4. Router A inserts the update packet information in its topology table. The topology table includes all destinations advertised by neighboring (adjacent) routers. It is organized so that each destination is listed, along with all the neighbors that can get to the destination and their associated metrics.
5. Router A then sends an update packet to Router B.
6. Upon receiving the update packet, Router B sends an ACK packet to Router A.

After Router A and Router B successfully receive the update packets from each other, they are ready to choose the successor (best) and FS (backup) routes in the topology table, and offer the successor routes to the routing table.

Split Horizon

Split horizon controls the sending of IP EIGRP update and query packets. When split horizon is enabled on an interface, no update or query packets for destinations for which this interface is the next hop are sent out of this interface. This reduces the possibility of routing loops. By default, EIGRP split horizon is enabled on most interfaces. (The "EIGRP over Frame Relay and on a Physical Interface" section notes an exception.)

Split horizon blocks information about a destination from being advertised by a router out of any interface that the router uses to route to that destination. This behavior usually optimizes communications among multiple routers, particularly when links are broken.

When a router changes its topology table in such a way that the interface through which the router reaches a network changes, it turns off split horizon and poisons the old route out of all interfaces, indicating that the route is unreachable. This ensures that other routers will not try to use the now-invalid route.

DUAL

Diffusing Update Algorithm (DUAL) is the finite-state machine that selects which information is stored in the topology and routing tables. As such, DUAL embodies the decision process for all EIGRP route computations. It tracks all routes advertised by all neighbors, uses the metric to select an efficient and loop-free path to each destination, and inserts that choice in the routing table.

Advertised Distance and Feasible Distance

The AD is the EIGRP metric for an EIGRP neighbor router to reach a particular network. This is the metric between the next-hop neighbor router and the destination network.

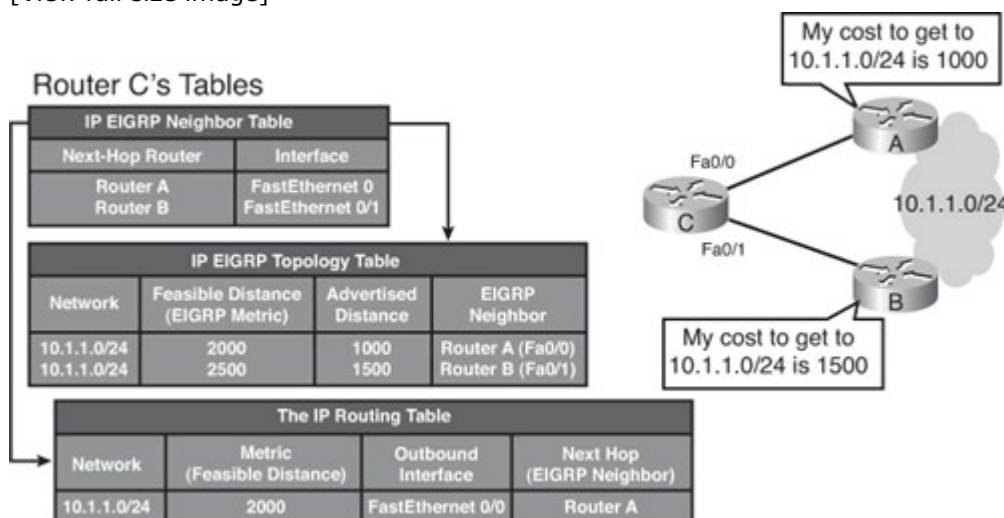
The FD is the EIGRP metric for this router to reach a particular network. This is the sum of the AD for the particular network learned from an EIGRP neighbor, plus the EIGRP metric to reach that neighbor (the metric between this router and the next-hop router).

A router compares all FDs to reach a specific network in its topology table. The route with the lowest FD is offered to its IP routing table. This is the successor route. The FD for the chosen route becomes the EIGRP routing metric to reach that network in the routing table.

For example, in Figure 2-5, Routers A and B send their routing tables to Router C, whose tables are shown in the figure. Both Routers A and B have paths to network 10.1.1.0/24 (among many others that are not shown).

Figure 2-5. EIGRP Chooses the Route with the Lowest Feasible Distance.

[View full size image]



The routing table on Router A has an EIGRP metric of 1000 for 10.1.1.0/24. Therefore, Router A advertises 10.1.1.0/24 to Router C with a metric of 1000. Router C places the 10.1.1.0/24 network from Router A in its EIGRP topology table with an AD of 1000. Router B has network 10.1.1.0/24 with a metric of 1500 in its IP routing table. Therefore, Router B advertises 10.1.1.0/24 to Router C with an AD of 1500. Router C places the 10.1.1.0/24 network from Router B in the EIGRP topology table with an AD of 1500.

Router C in Figure 2-5 has two entries to reach 10.1.1.0/24 in its topology table. The EIGRP metric for Router C to reach either Router A or B is 1000. This cost (1000) is added to the respective AD from each router, and the results represent the FDs that Router C must travel to reach network 10.1.1.0/24. Router C chooses the least-cost FD (in this case 2000, via Router A) and installs it in its IP routing table as the best route to reach 10.1.1.0/24. The EIGRP metric in the routing table is the best FD from the EIGRP topology table. In this case Router C's routing table shows the route to 10.1.1.0/24 is via Router A with a metric of 2000.

Successor and Feasible Successor

A successor, also called a current successor, is a neighboring router used for packet forwarding that has a least-cost path to a destination that is guaranteed not to be part of a routing loop.

The FD, not the AD, affects the selection of the best routes for incorporation in the routing table. The AD is used only to calculate the FD. A router is chosen as a successor because it has the lowest FD of all possible paths to that destination network. The successor is the next router in line to reach that destination—it is the next-hop router in the best path to reach that destination network.

An EIGRP router selects the best path to reach a given network and then installs the destination network, the metric to reach that network, the outbound interface to reach the next-hop router, and the IP address of the next-hop router into the IP routing table. If the EIGRP topology table has many entries that have an equal-cost FD to a given destination network, all successors (up to four by default) for that destination network are installed in the routing table.

All IP routing protocols can install only the next-hop router information in the routing table. Information about the subsequent routers in the path is not put in the routing table. Each router relies on the next-hop router to make a reliable decision to reach a specific destination network. The hop-by-hop path through a network goes from one router to the next. Each router makes a path selection to reach a given network and installs the best next-hop address along the path to reach that destination network. A router trusts a route's successor (the best next-hop router) to send traffic toward that destination address.

The routing table is essentially a subset of the topology table. The topology table contains more detailed information about each route, any backup routes, and information used exclusively by DUAL.

An FS is a router providing a backup route. The route through the FS must be loop free. In other words, it must not loop back to the current successor. FSs are selected at the same time the successors are identified. These FS routes are kept in the topology table. The topology table can retain multiple FS routes for a destination.

An FS must be mathematically proven. To qualify as an FS, a next-hop router must have an AD less than the FD of the current successor route for the particular network. This is known as the feasibility condition. This requirement ensures that the FS cannot use a route through the local router (which would be a routing loop), because the AD through the FS is less than the best route through the local router. For example, as shown in Router C's topology table in Figure 2-6 (for the network shown earlier in Figure 2-5), Router B is an FS, because the AD through Router B (1500) is less than the FD of the current successor, Router A (2000).

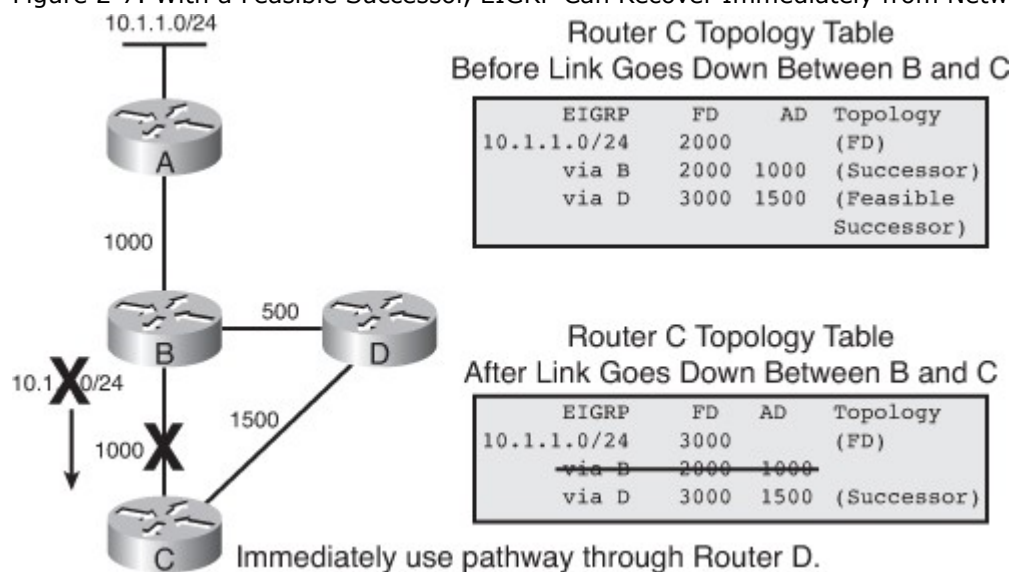
Figure 2-6. Router C's Topology Table: Feasible Successor's AD Must Be Less Than the Successor's FD.

EIGRP Topology Table			
Network	FD (EIGRP Metric)	AD	EIGRP Neighbor
10.1.1.0/24	2000-Successor	1000	Router A (Fa0/0)
10.1.1.0/24	2500	1500FeasibleSuccessor	Router B (Fa0/1)

When a router loses a route, it looks at the topology table for an FS. If one is available, the route does not go into an active state. Instead, the best FS is promoted as the successor and is installed in the routing table. The FS can be used immediately, without any recalculation. If there are no FSs, a route goes into active state, and route computation occurs. Through this process, a new successor is determined (if there is one). The amount of time it takes to recalculate the route affects the convergence time.

Figure 2-7 illustrates another example. Router C's initial topology table is shown at the top of the figure. Router B is the successor for network 10.1.1.0/24, and Router D is the FS.

Figure 2-7. With a Feasible Successor, EIGRP Can Recover Immediately from Network Failures.

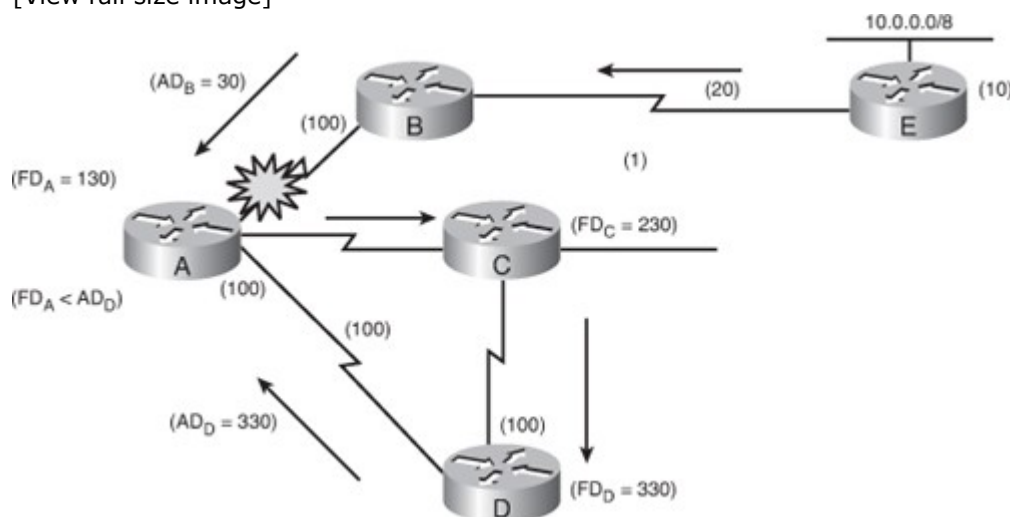


In Figure 2-7, the link between Router B and Router C fails. Router C removes the route 10.1.1.0/24 through Router B from its routing table and searches the EIGRP topology table for an FS; Router D is an FS. Because Router D can still reach the network and does not send an update or query packet to inform Router C of the lost route, Router C immediately uses the path through Router D. Router C chose this path as an FS because the AD through Router D (1500) is less than the FD of the best route, through Router B (2000). This path is guaranteed to be loop free.

Figure 2-8 illustrates another scenario that shows how DUAL ensures a loop-free network. Router B sends the routing update about network 10.0.0.0/8, with an AD of 30. Router A receives the update, calculates the FD value (130), and sends an update to both of its neighbors, routers C and D. Routers A, C, and D are in a loop. The update sent to Router C is sent on to Router D, which then sends it to Router A. The AD of the route that Router D sends to Router A is 330. This AD value is higher than the FD (130) on Router A, calculated from the original update received from Router B. Because the FD of the route on Router A is smaller than the AD of the update coming from Router D, the route via Router D does not become an FS. Thus, DUAL ensures there will be no routing loop in the network.

Figure 2-8. DUAL Ensures a Loop-free Network.

[View full size image]



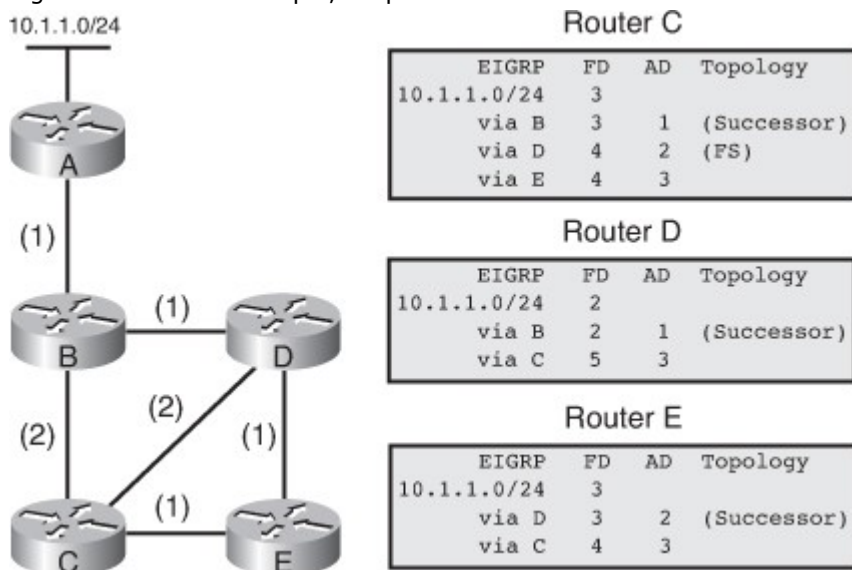
DUAL Example

The mathematical formula to ensure that the FS is loop free requires that the AD of the backup route be *less than* the FD of the successor. When the AD of the second-best route is greater than or equal to the FD of the

successor, an FS cannot be chosen. In this case, a discovery process that uses EIGRP queries and replies must be used to find any alternative paths to the lost networks.

The following example examines partial entries for network 10.1.1.0/24 in the topology tables for Routers C, D, and E in Figure 2-9, to give you a better understanding of EIGRP behavior. The partial topology tables shown in Figure 2-9 indicate the following:

Figure 2-9. DUAL Example, Step 1.



- **AD**— The advertised distance is equal to the cost of the path to network 10.1.1.0/24 as advertised by neighboring routers. For example, consider Router E's neighbors: Router D has an AD of 2 and Router C has an AD of 3, for 10.1.1.0/24.
- **FD**— The feasible distance is equal to the sum of the AD for a neighbor to reach 10.1.1.0/24, plus the metric to reach that neighbor. For example, again consider Router E: The FD of the route to 10.1.1.0/24 via Router D is Router D's AD (2) plus the metric to reach Router D from Router E (1), for a total of 3. The FD of the route to 10.1.1.0/24 via Router C is Router C's AD (3) plus the metric to reach Router C from Router E (1), for a total of 4.
- **Successor**— The successor is the forwarding path used to reach network 10.1.1.0/24. The cost of this path is equal to the FD. For example, Router E chooses the path to 10.1.1.0/24 with the lowest FD, which is via Router D. This is the route that Router E puts in its routing table.
- **FS**— The feasible successor is an alternative loop-free path to reach network 10.1.1.0/24. For example, in Router C, the path via Router D is an FS because the AD (2) is less than the FD (3) via the successor Router B. Routers D and E do not have any FSs because the AD of the alternate routes are not less than the FD of their current successors.

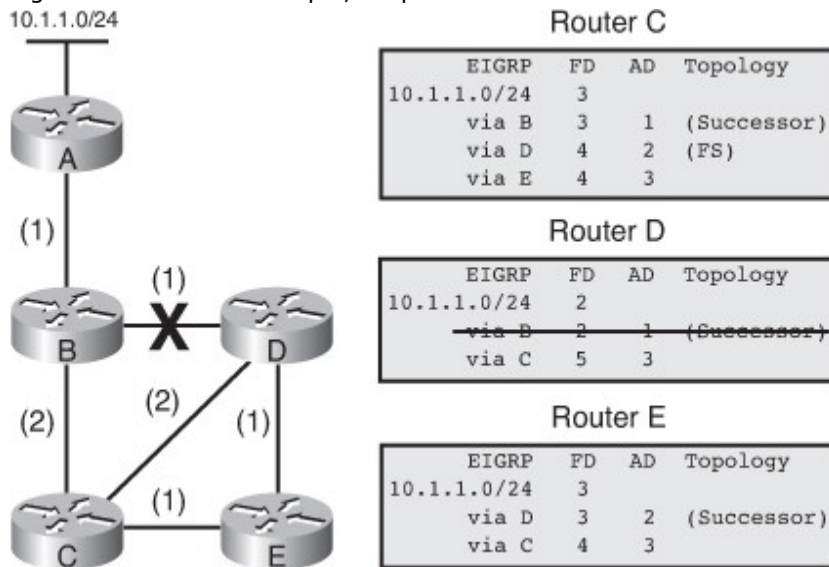
The network shown in Figure 2-9 is stable and converged.

Note

As mentioned earlier, EIGRP implements split horizon. For example, Router E does not pass its route for network 10.1.1.0/24 to Router D, because Router E uses Router D as its next hop to network 10.1.1.0/24.

In Figure 2-10, Routers B and D detect a link failure. In Router D, DUAL marks the path to network 10.1.1.0/24 through Router B as unusable after being notified of the link failure,

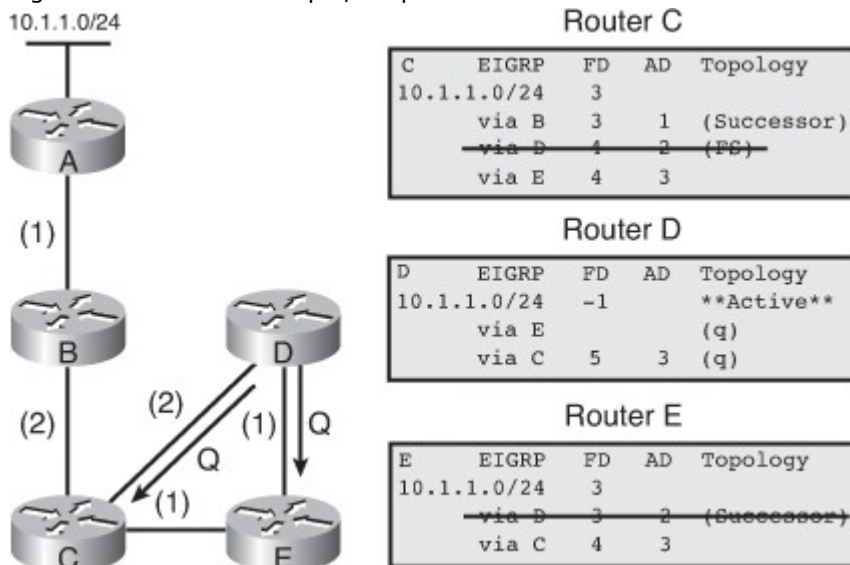
Figure 2-10. DUAL Example, Step 2.



The following steps then occur, as shown in Figure 2-11:

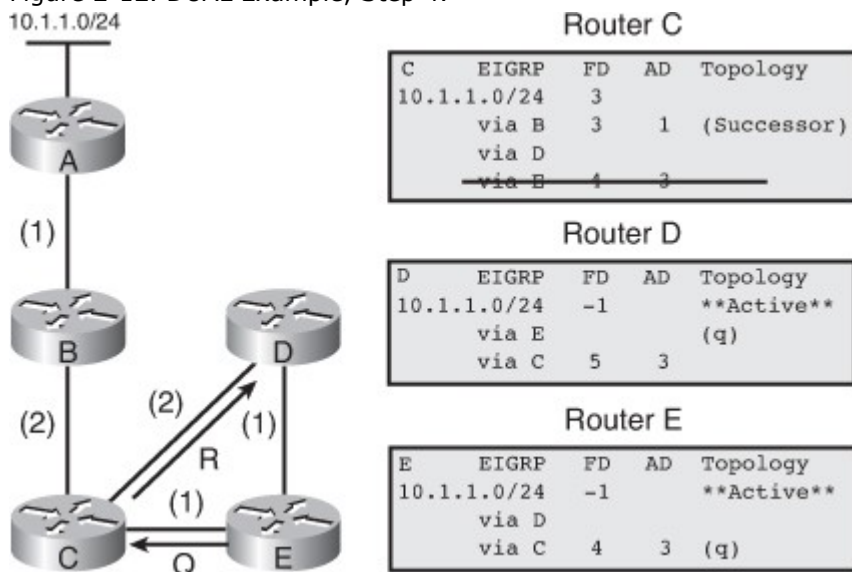
- At Router D, there is no FS to network 10.1.1.0/24, because the AD via Router C (3) is greater than the FD via Router B (2). Therefore, DUAL does the following:
 - Sets the metric to network 10.1.1.0/24 as unreachable (-1 is unreachable).
 - Because an FS cannot be found in the topology table, the route changes from the passive state to the active state. In the active state, the router sends out queries to neighboring routers looking for a new successor.
 - Sends a query to Routers C and E for an alternative path to network 10.1.1.0/24.
 - Marks Routers C and E as having a query pending (q).
- At Router E, DUAL marks the path to network 10.1.1.0/24 through Router D as unusable.
- At Router C, DUAL marks the path to network 10.1.1.0/24 through Router D as unusable.

Figure 2-11. DUAL Example, Step 3.



The following steps then occur, as shown in Figure 2-12:

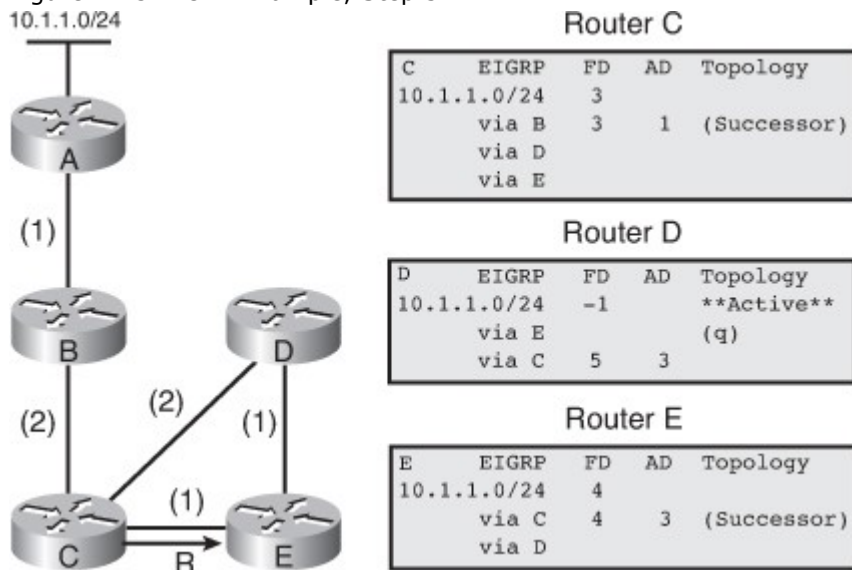
Figure 2-12. DUAL Example, Step 4.



- At Router D:
 - DUAL receives a reply from Router C that indicates no change to the path to network 10.1.1.0/24.
 - DUAL removes the query flag from Router C.
 - DUAL stays active on network 10.1.1.0/24, awaiting a reply from Router E to its query (q).
- At Router E, there is no FS to network 10.1.1.0/24, because the AD from Router C (3) is not less than the original FD (also 3).
 - DUAL generates a query to Router C.
 - DUAL marks Router C as query pending (q).
- At Router C, DUAL marks the path to network 10.1.1.0/24 through Router E as unusable.

The following steps then occur, as shown in Figure 2-13:

Figure 2-13. DUAL Example, Step 5.



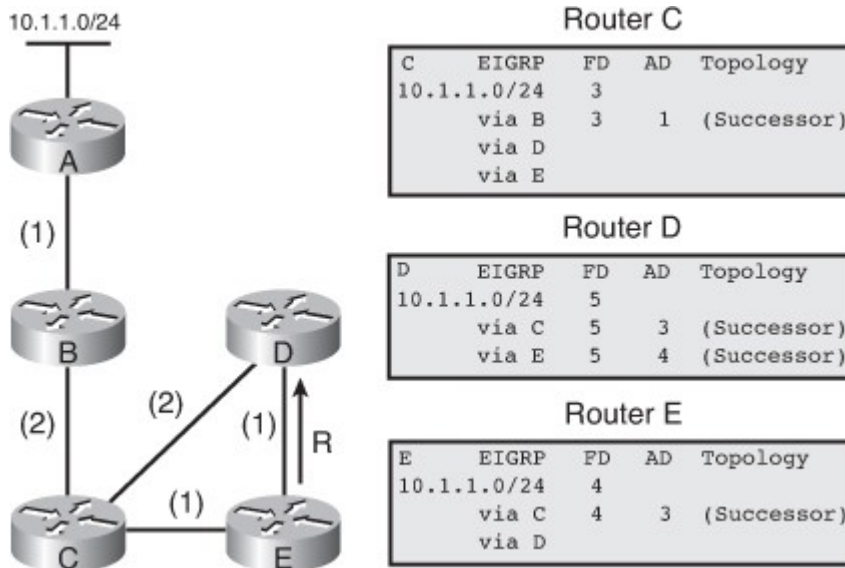
- At Router D, DUAL stays active on network 10.1.1.0/24, awaiting a reply from Router E (q).
- At Router E:
 - DUAL receives a reply from Router C indicating no change.

- b. It removes the query flag from Router C.
- c. It calculates a new FD and installs a new successor route in the topology table.
- d. It changes the route to network 10.1.1.0/24 from active to passive (converged).

The following steps then occur, as shown in Figure 2-14 at Router D:

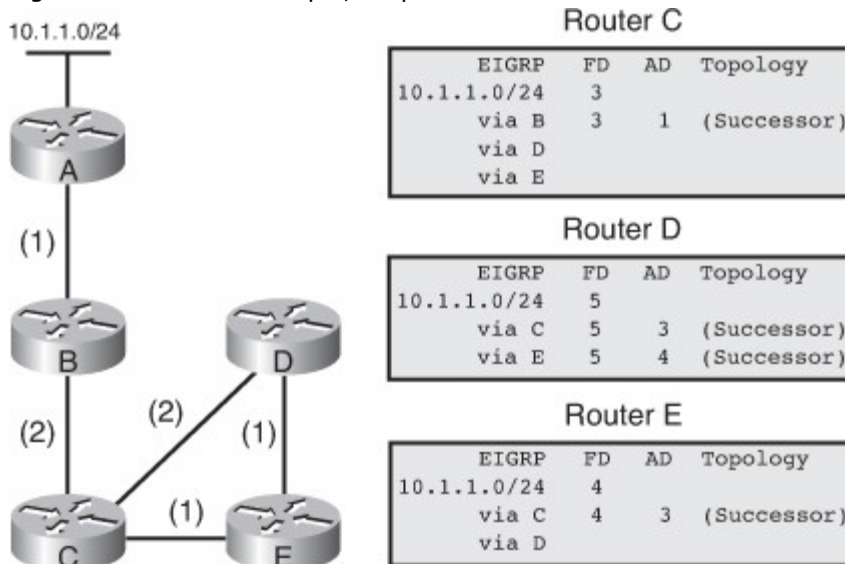
1. DUAL receives a reply from Router E.
2. It removes the query flag from Router E.
3. It calculates a new FD.
4. It installs new successor routes in the topology table. Two routes (through Routers C and E) have the same FD, and both are marked as successors.
5. It changes the route to network 10.1.1.0/24 from active to passive (converged).

Figure 2-14. DUAL Example, Step 6.



The following steps then occur, as shown in Figure 2-15:

Figure 2-15. DUAL Example, Step 7.



1. At Router D, two successor routes are in the topology table for network 10.1.1.0/24. Both successor

routes are listed in the routing table, and equal-cost load balancing is in effect.

2. The network is stable and converged.

Figure 2-9, the original topology before the link failure, shows traffic from Router E for 10.1.1.0/24 passing through Routers D and B. In Figure 2-15, the new topology shows traffic from Routers D and E for 10.1.1.0/24 going through Routers C and B. Notice that throughout the entire convergence process, routes to network 10.1.1.0/24 become active only on Routers D and E. The route to network 10.1.1.0/24 on Router C remains passive because the link failure between Routers B and D does not affect the successor route from Router C to network 10.1.1.0/24.

Note

When DUAL decides that a packet needs to be transmitted to a neighbor, the packets are not actually generated until the moment of transmission. Instead, the transmit queues contain small, fixed-size structures that indicate which parts of the topology table to include in the packet when it is finally transmitted. This means that the queues do not consume large amounts of memory. It also means that only the latest information is transmitted in each packet. If a route changes state several times, only the last state is transmitted in the packet, thus reducing link utilization.

EIGRP Metric Calculation

DUAL selects routes based on the EIGRP composite metric. Five criteria are associated with the EIGRP composite metric, but EIGRP uses only two by default:

- **Bandwidth**— The smallest (slowest) bandwidth between the source and destination
- **Delay**— The cumulative interface delay along the path

The following criteria, although available, are not commonly used, because they typically result in frequent recalculation of the topology table:

- **Reliability**— The worst reliability between the source and destination, based on keepalives.
- **Loading**— The worst load on a link between the source and destination based on the packet rate and the interface's configured bandwidth.
- **Maximum transmission unit (MTU)**— The smallest MTU in the path. (MTU is included in the EIGRP update but is actually not used in the metric calculation.)

EIGRP calculates the metric by adding together weighted values of different variables of the path to the network in question. The default constant weight values are $K1 = K3 = 1$, and $K2 = K4 = K5 = 0$.

In EIGRP metric calculations, when $K5$ is 0 (the default), variables (bandwidth, bandwidth divided by load, and delay) are weighted with the constants $K1$, $K2$, and $K3$. The following is the formula used:

$$\text{metric} = (K1 * \text{bandwidth}) + [(K2 * \text{bandwidth}) / (256 - \text{load})] + (K3 * \text{delay})$$

If these K values are equal to their defaults, the formula becomes

$$\text{metric} = (1 * \text{bandwidth}) + [(0 * \text{bandwidth}) / (256 - \text{load})] + (1 * \text{delay})$$

$$\text{metric} = \text{bandwidth} + [0] + \text{delay}$$

$$\text{metric} = \text{bandwidth} + \text{delay}$$

If $K5$ is not equal to 0, the following additional operation is performed:

$$\text{metric} = \text{metric} * [K5 / (\text{reliability} + K4)]$$

K values are carried in EIGRP hello packets. Mismatched K values can cause a neighbor to be reset (even though only $K1$ and $K3$ are used, by default, in metric compilation). These K values should be modified only after careful planning; changing these values can prevent your network from converging and is generally not recommended.

It is important to note that the format of the delay and bandwidth values is different from those displayed by the **show interfaces** command, as follows:

- The EIGRP delay value is the sum of the delays in the path, in tens of microseconds, multiplied by 256. The **show interfaces** command displays delay in microseconds.
- The EIGRP bandwidth is calculated using the minimum bandwidth link along the path, represented in kilobits per second (kbps). 10^7 is divided by this value, and then the result is multiplied by 256.

EIGRP uses the same metric formula as IGRP did, but EIGRP represents its metrics in a 32-bit format rather than the 24-bit representation used by IGRP. This representation allows a more granular decision to be made when determining the successor and feasible successor.

The EIGRP metric value ranges from 1 to 4,294,967,296. (The IGRP metric value ranged from 1 to 16,777,216.) EIGRP metrics are backward compatible with IGRP, as illustrated in Figure 2-16. When integrating IGRP routes into an EIGRP domain using redistribution, the router multiplies the IGRP metric by

256 to compute the EIGRP-equivalent metric. When sending EIGRP routes to an IGRP routing domain, the router divides each EIGRP metric by 256 to achieve the proper 24-bit metric.

Figure 2-16. The EIGRP Metric Is Backward Compatible with the IGRP Metric.

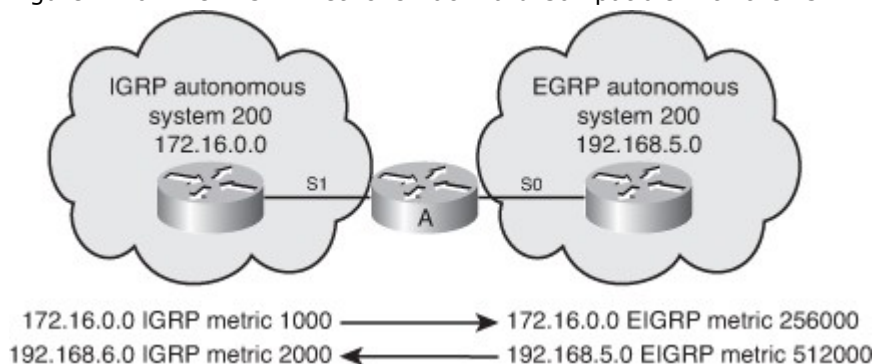
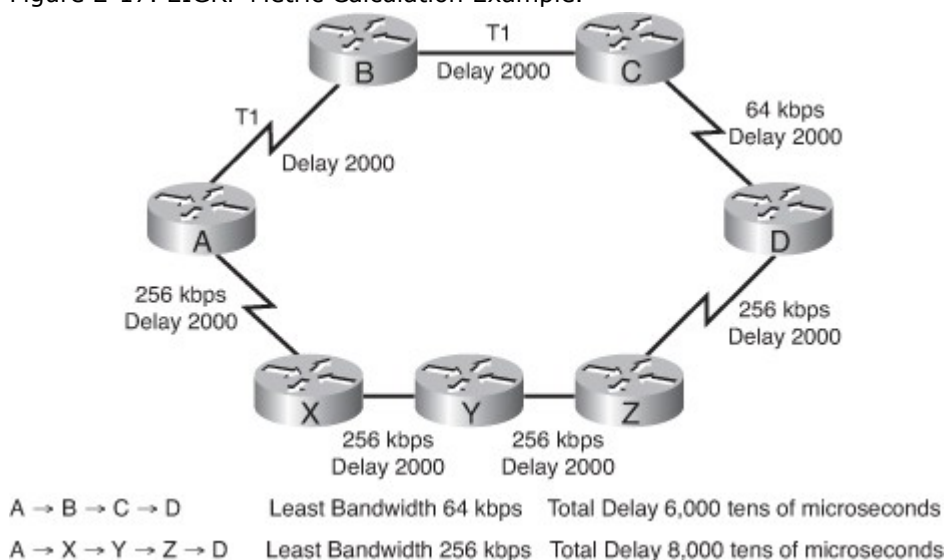


Figure 2-17 illustrates an example network used to illustrate the metric calculation. In this figure, Router A has two paths to reach Router D (and therefore any networks behind Router D). The bandwidths (in kbps) and the delays (in tens of microseconds) of the various links are also shown in the figure.

Figure 2-17. EIGRP Metric Calculation Example.



The least bandwidth along the top path (A → B → C → D) is 64 kbps. The EIGRP bandwidth calculation for this path is as follows:

$$\text{bandwidth} = (10^7 / \text{least bandwidth in kbps}) * 256$$

$$\text{bandwidth} = (10,000,000 / 64) * 256 = 156,250 * 256 = 40,000,000$$

The delay through the top path is as follows:

$$\text{delay} = [(\text{delay A} \rightarrow \text{B}) + (\text{delay B} \rightarrow \text{C}) + (\text{delay C} \rightarrow \text{D})] * 256$$

$$\text{delay} = [2000 + 2000 + 2000] * 256$$

$$\text{delay} = 1,536,000$$

Therefore, the EIGRP metric calculation for the top path is as follows:

$$\text{metric} = \text{bandwidth} + \text{delay}$$

$$\text{metric} = 40,000,000 + 1,536,000$$

$$\text{metric} = 41,536,000$$

The least bandwidth along the lower path (A → X → Y → Z → D) is 256 kbps. The EIGRP bandwidth calculation for this path is as follows:

$$\text{bandwidth} = (10^7 / \text{least bandwidth in kbps}) * 256$$

$$\text{bandwidth} = (10,000,000 / 256) * 256 = 10,000,000$$

The delay through the lower path is as follows:

$$\text{delay} = [(\text{delay A} \rightarrow \text{X}) + (\text{delay X} \rightarrow \text{Y}) + (\text{delay Y} \rightarrow \text{Z}) + (\text{delay Z} \rightarrow \text{D})] * 256$$
$$\text{delay} = [2000 + 2000 + 2000 + 2000] * 256$$
$$\text{delay} = 2,048,000$$

Therefore, the EIGRP metric calculation for the lower path is as follows:

$$\text{metric} = \text{bandwidth} + \text{delay}$$
$$\text{metric} = 10,000,000 + 2,048,000$$
$$\text{metric} = 12,048,000$$

Router A therefore chooses the lower path, with a metric of 12,048,000, over the top path, with a metric of 41,536,000. Router A installs the lower path with a next-hop router of X and a metric of 12,048,000 in the IP routing table.

The bottleneck along the top path, the 64-kbps link, can explain why the router takes the lower path. This slow link means that the rate of transfer to Router D would be at a maximum of 64 kbps. Along the lower path, the lowest speed is 256 kbps, making the throughput rate up to that speed. Therefore, the lower path represents a better choice, such as to move large files quickly.

This section concludes the discussion of EIGRP terminology and operation. The rest of the chapter explores planning, configuring, and verifying EIGRP implementation.

Planning EIGRP Routing Implementations

This section describes how to plan and document an EIGRP deployment.

When preparing to deploy EIGRP in a network, you first need to gather the requirements, determine the existing network state, and consider different deployment options. Considerations for EIGRP include the following:

- **IP addressing plan**— The IP addressing plan governs how EIGRP can be deployed and how well the EIGRP deployment will scale. A detailed IP subnet and addressing plan must be produced, and should be hierarchical to enable EIGRP summarization, allow the network to scale more easily, and to optimize EIGRP behavior.
- **Network topology**— The topology consists of the devices (routers, switches, and so on) and the links connecting them. A detailed network topology should be created to assess EIGRP scalability requirements and to determine which EIGRP features might be required (for example, EIGRP stub routing).
- **EIGRP traffic engineering**— By changing the interface metrics, EIGRP traffic engineering can be deployed to improve bandwidth utilization and enable the administrator to have control over traffic patterns.

After you have assessed the requirements, you can create the implementation plan. The information necessary to implement EIGRP routing includes the following:

- The IP addresses to be configured on individual router interfaces.
- The EIGRP autonomous system number, used to enable EIGRP. The autonomous system number must be the same on all the routers in the EIGRP domain.
- A list of routers on which EIGRP is to be enabled along with the connected networks that are to run EIGRP and that need to be advertised (per individual router).
- Metrics that need to be applied to specific interfaces, for EIGRP traffic engineering. The required metric and the interface where the metric needs to be applied should be specified.

In the implementation plan, the list of tasks for each router in the network must be defined. For EIGRP, the tasks include the following:

- Enabling the EIGRP routing protocol
- Configuring the proper network statements
- Optionally configuring the metric to appropriate interfaces

After implementing EIGRP, it should be verified to confirm proper deployment on each router. Verification tasks include the following:

- Verifying the EIGRP neighbor relationships

- Verifying that the EIGRP topology table is populated with the necessary information
- Verifying that IP routing table is populated with the necessary information
- Verifying that there is connectivity in the network between routers and to other devices
- Verifying that EIGRP behaves as expected in a case of a topology change, by testing link failure and router failure events

After a successful EIGRP deployment, document the solution, the verification process, and the results, for future reference. Documentation should include a topology map, the IP addressing plan, the autonomous system number used, the networks included in EIGRP on each router, and any special metrics configured.

Configuring and Verifying EIGRP

This section covers the commands used to configure and verify EIGRP features. The following topics are discussed:

- Planning and configuring basic EIGRP
- Verifying EIGRP operation
- Using the passive-interface command with EIGRP
- Propagating an EIGRP default route
- EIGRP route summarization

Planning and Configuring Basic EIGRP

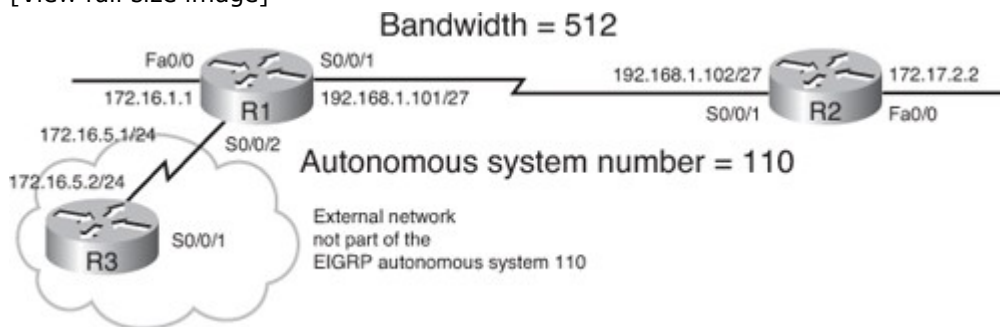
This section describes how to plan and configure basic EIGRP, and provides some detailed examples.

Planning for Basic EIGRP

The example network in Figure 2-18 is to be configured for EIGRP. The implementation plan for this network should include the following steps:

Figure 2-18. Sample Network for Planning and Implementing Basic EIGRP.

[View full size image]



- Step 1. Define the network requirements.
- Step 2. Gather the required parameters.
- Step 3. Define the EIGRP routing parameters.
- Step 4. Configure basic EIGRP.
- Step 5. Verify the EIGRP configuration.

Requirements and Parameters

Figure 2-18 shows three routers. Routers R1 and R2 are in EIGRP autonomous system 110. Router R3 is in an external network that is not part of the EIGRP autonomous system. Requirements for basic EIGRP are as follows:

- **IP addressing**— IP addresses for all device interfaces. Figure 2-18 includes the IP addresses for this example.
- **EIGRP routing protocol autonomous system number**— Recall that routers in the same EIGRP domain must have the same autonomous system number because each EIGRP process must be started with the same autonomous system number. Autonomous system number 110 is used in this example.

- **Interfaces for EIGRP neighborhood and networks participating in EIGRP**— The interfaces included in EIGRP are used to exchange routing updates and other EIGRP packets between neighbors. The interfaces on which EIGRP is to run and the network numbers that are to participate in EIGRP must be defined. Both Router R1 and R2 have one serial interface and one Fast Ethernet interface included in EIGRP process. EIGRP routers advertise their local networks to all neighbors. Both Router R1 and R2 will advertise their directly to connected networks that are part of the EIGRP domain.
- **Interface bandwidth**— Because interface bandwidth is part of the EIGRP metric, changing the bandwidth changes the metric of the link. To influence path selection, interface bandwidth can be defined. The actual bandwidth on the serial link between Router R1 and R2 is 512 kbps and the configuration should reflect this so that EIGRP can select the proper route.

Note

In earlier Cisco IOS releases, the default bandwidth on all serial ports was T1, or 1.544 megabits per second (Mbps). In the latest Cisco IOS releases, the default bandwidth varies with interface type.

Note

Similarly, the **delay tens-of-microseconds** interface configuration command can be configured if the default delay value on the interface does not reflect the delay that should be used in the metric calculation. The parameter specifies the delay in tens of microseconds. Use the **show interfaces** command to see the default delay.

Basic EIGRP Configuration

The implementation plan should define the following tasks to configure basic EIGRP:

- Step 1. **Define EIGRP as the routing protocol:** To enable EIGRP and define the autonomous system, use the **router eigrp autonomous-system-number** global configuration command. In this command, the *autonomous-system-number* identifies the local autonomous system and is used to identify this router to the other EIGRP routers that belong within the internetwork. This value must match on all routers within the internetwork. This command puts the router into router configuration mode.
- Step 2. Define the attached networks participating in EIGRP: To indicate which networks are part of the EIGRP autonomous system, use the **network network-number [wildcard-mask]** router configuration command. Table 2-1 summarizes the parameters of this command. This command defines the interfaces over which EIGRP will attempt to establish EIGRP neighbor relationships and the networks that will be advertised to the EIGRP neighbors.

Table 2-1. *network* Command Parameters

Parameter	Description
network-number	This parameter can be a network, a subnet, or the address of a directly connected interface. It determines which links on the router to advertise to, which links to listen to advertisements on, and which networks are advertised. Network commands should be configured only for interfaces on which the router will send and receive updates.
wildcard-mask	(Optional) An inverse mask used to determine how to interpret the <i>network-number</i> . The mask has wildcard bits, where 0 is a match and 1 is do not care. For example, 0.0.255.255 indicates a match in the first 2 octets.

If you do not use the optional wildcard mask, the EIGRP process assumes that all directly connected networks that are part of the major network will participate in the EIGRP routing process, and EIGRP will attempt to establish EIGRP neighbor relationships from each interface that is part of the overall Class A, B, or C network.

Use the optional wildcard mask to identify a specific IP address, subnet, or network. The router interprets the network number using the wildcard mask to determine which connected interfaces will participate in the EIGRP routing process. The router then attempts to establish neighbor relationships on those interfaces. If you want to specify an interface address, use the mask 0.0.0.0

to match all 4 octets of the address. An address and wildcard mask combination of 0.0.0.0 255.255.255.255 matches all interfaces on the router.

There is no limit to the number of **network** commands that can be configured on the router.

Note

If you incorrectly use a subnet mask in the network command, you probably assume that the router would interpret the mask incorrectly. We assumed that. However, when we tested the configuration, surprisingly, the router "corrected" our mistake instead. For example, the following commands were entered on a router:

```
RouterA(config)#router eigrp 100
```

```
RouterA(config-router)#network 10.1.0.0 255.255.0.0
```

The following shows the configuration on the router; notice how the mask has been "fixed".

```
RouterA#sh run | begin router eigrp
```

```
router eigrp 100
```

```
network 10.1.0.0 0.0.255.255
```

```
<output omitted>
```

Because we could not find any documentation on this feature, we encourage you to check your mask configurations carefully.

- Step 3. Define the interface bandwidth: For serial links, the link's bandwidth may be specified for the purposes of sending routing update traffic on the link. If you do not define the bandwidth value for these interfaces, EIGRP assumes that the bandwidth on the link is the default, which varies with interface type. Recall that EIGRP uses bandwidth as part of its metric calculation. If the link is actually slower than the default, the router might not be able to converge, or routing updates might become lost. (The percent of the interface's bandwidth that EIGRP uses can also be limited, as described in the "EIGRP Link Utilization" section, later in this chapter.) To define the bandwidth, use the bandwidth kilobits interface configuration command. In this command, kilobits indicates the intended bandwidth in kbps.

The bandwidth command sets an informational parameter only; you cannot adjust the actual bandwidth of an interface with this command. For some media, such as Ethernet, the bandwidth is fixed. For other media, such as serial lines, you can change the actual bandwidth by adjusting hardware parameters. For both classes of media, you can use the bandwidth configuration command to communicate the current bandwidth to the higher-level protocols.

For generic serial interfaces such as PPP and HDLC, set the bandwidth to match the line speed. For Frame Relay point-to-point interfaces, set the bandwidth to the committed information rate (CIR). For Frame Relay multipoint connections, set the bandwidth to the sum of all CIRs, or if the permanent virtual circuits (PVCs) have different CIRs, set the bandwidth to the lowest CIR multiplied by the number of PVCs on the multipoint connection.

Note

The apparent bandwidth of the outgoing interface is also used for other purposes. For example, the TCP protocol adjusts the initial retransmission parameters based on this bandwidth.

Basic Configuration Example

The configuration for Router R1 in Figure 2-18 is shown in Example 2-2. On Router R1, EIGRP is enabled in autonomous system 110 using the router eigrp 110 command. The network 172.16.1.0 0.0.0.255 command starts EIGRP on the Fast Ethernet 0/0 interface and allows Router R1 to advertise this network. With the wildcard mask used, this command specifies that only interfaces on the 172.16.1.0/24 subnet will participate in EIGRP. (However, the full Class B network 172.16.0.0 will be advertised to Router R2 because EIGRP automatically summarizes routes on the major network boundary by default.) The network 192.168.1.0 command starts EIGRP on the Serial 0/0/1 interface and allows Router R1 to advertise this network.

Example 2-2. Configuration of Router R1 in Figure 2-18

```
interface FastEthernet0/0
ip address 172.16.1.1 255.255.255.0

interface Serial0/0/1
bandwidth 512
ip address 192.168.1.101 255.255.255.224

interface Serial0/0/1
ip address 172.16.5.1 255.255.255.0

router eigrp 110
network 172.16.1.0 0.0.0.255
network 192.168.1.0
```

Router R1 is connected to the Router R3 and subnet 172.16.5.0 is used on the link between the two routers. Because Router R1 is not configured to run EIGRP on 172.16.5.0, Router R1 does not try to form an adjacency with the Router R3. If the wildcard mask was not used on Router R1, it would run EIGRP on all interfaces in 172.16.0.0/16 and would send EIGRP packets to Router R3. This would waste bandwidth and CPU cycles and would provide unnecessary information to the external network. The wildcard mask tells EIGRP to establish a relationship with EIGRP routers from interfaces that are part of subnet 172.16.1.0/24 only.

The **bandwidth 512** command on interface Serial 0/0/1 sets the bandwidth to 512 kbps.

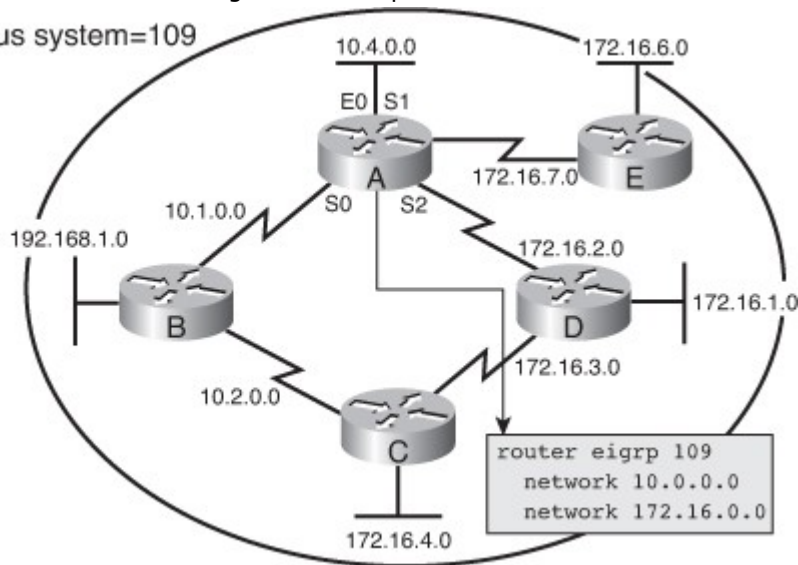
The configuration on Router R2 would be similar to Router R1's configuration.

Another Basic EIGRP Configuration Example

Figure 2-19 shows another sample network, including the configuration of Router A for EIGRP.

Figure 2-19. Basic EIGRP Configuration Sample Network.

Autonomous system=109



All routers in the network are part of autonomous system 109, so they can establish a neighbor relationship. Because the wildcard mask is not used in Router A's configuration, all interfaces on Router A that are part of network 10.0.0.0/8 and network 172.16.0.0/16 participate in the EIGRP routing process. In this case, this includes all four interfaces. Note that network 192.168.1.0 is not configured in the EIGRP configuration on Router A, because Router A does not have any interfaces in that network. Instead, suppose that the configuration in Example 2-3 was entered onto Router A.

Example 2-3. Alternative Configuration of Router A in Figure 2-19

```
routerA(config)#router eigrp 109  
routerA(config-router)#network 10.1.0.0  
routerA(config-router)#network 10.4.0.0  
routerA(config-router)#network 172.16.7.0  
routerA(config-router)#network 172.16.2.0
```

Because no wildcard mask was specified in Example 2-3, Router A would automatically change the network commands to have classful networks, and the resulting configuration would be as shown in Example 2-4.

Example 2-4. Router A's Interpretation of the Configuration in Example 2-3

```
router eigrp 109  
network 10.0.0.0  
network 172.16.0.0
```

Alternatively, consider what would happen if the configuration shown in Example 2-5 was entered for Router A.

Example 2-5. Another Alternative Configuration of Router A in Figure 2-19

```
routerA(config)#router eigrp 109  
routerA(config-router)#network 10.1.0.0  
0.0.255.255  
routerA(config-router)#network 10.4.0.0  
0.0.255.255  
routerA(config-router)#network 172.16.2.0  
0.0.0.255  
routerA(config-router)#network 172.16.7.0  
0.0.0.255
```

In this case, Router A uses the wildcard mask to determine which directly connected interfaces participate in the EIGRP routing process for autonomous system 109. All interfaces that are part of networks 10.1.0.0/16, 10.4.0.0/16, 172.16.2.0/24, and 172.16.7.0/24 participate in the EIGRP routing process for autonomous system 109. In this case, all four interfaces participate.

Verifying EIGRP Operation

This section discusses commands used to verify EIGRP operation.

Table 2-2 describes some show commands used to verify EIGRP operation. Other options might be available with these commands. Use the Cisco IOS integrated help feature to see the full command syntax.

Table 2-2. EIGRP *show* Commands

Command	Description
show ip eigrp neighbors	Displays neighbors discovered by EIGRP.
show ip route	Displays the current entries in the IP routing table for all configured routing protocols.
show ip route eigrp	Displays the current EIGRP entries in the IP routing table.
show ip protocols	Displays the parameters and current state of the active routing protocol processes. For EIGRP, this command shows the EIGRP autonomous system number, filtering and redistribution numbers, and neighbors and distance information.
show ip eigrp interfaces	Displays information about interfaces configured for EIGRP.

Command	Description
show ip eigrp topology	Displays the EIGRP topology table. This command shows the topology table, the active or passive state of routes, the number of successors, and the FD to the destination. Note that only successor and feasible successor routes are displayed. Add the all-links keyword to display all routes, including those not eligible to be successor or feasible successor routes.
show ip eigrp traffic	Displays the number of EIGRP packets sent and received. This command displays statistics on hello packets, updates, queries, replies, and acknowledgments.

Table 2-3 describes debug commands used to verify EIGRP operation. Other options might be available with these commands. Use the Cisco IOS integrated help feature to see the full command syntax.

Table 2-3. EIGRP *debug* Commands

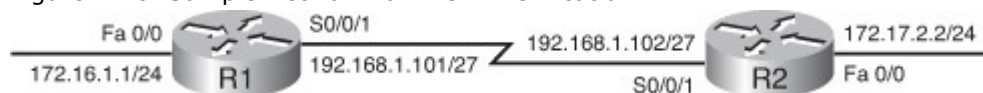
Command	Description
debug eigrp packets	Displays the types of EIGRP packets sent and received. A maximum of 11 packet types can be selected for individual or group display.
debug ip eigrp	Displays packets that are sent and received on an interface. Because this command generates large amounts of output, use it only when traffic on the network is light.
debug ip eigrp summary	Displays IP EIGRP summary route processing.
debug eigrp neighbors	Displays neighbors discovered by EIGRP and the contents of the hello packets.

Caution

Use caution when executing **debug** commands, because they may consume a lot of router resources and could cause problems in a busy production network. Debugging output takes priority over other network traffic. Too much debug output may severely reduce the performance of the router or even render it unusable in the worst case.

The following sections provide sample output from some of these commands, using the network in Figure 2-20 to illustrate the configuration, verification, and troubleshooting of EIGRP. Example 2-6 shows the configuration of the R1 router.

Figure 2-20. Sample Network for EIGRP Verification.



Example 2-6. Configuration for Router R1 in Figure 2-20

```

R1#show running-config
<output omitted>
interface FastEthernet0/0
 ip address 172.16.1.1 255.255.255.0

<output omitted>
interface Serial0/0/1
 bandwidth 64

```

```
ip address 192.168.1.101 255.255.255.224
```

<output omitted>

```
router eigrp 100
```

```
network 172.16.1.0 0.0.0.255
```

```
network 192.168.1.0
```

On the R1 router, EIGRP is enabled in autonomous system 100. The **network 172.16.1.0 0.0.0.255** command starts EIGRP on the Fast Ethernet 0/0 interface and allows Router R1 to advertise this network. With the wildcard mask used, this command specifies that only interfaces on the 172.16.1.0/24 subnet will participate in EIGRP. Note, however, the full Class B network 172.16.0.0 will be advertised, because EIGRP automatically summarizes routes on the major network boundary by default. The **network 192.168.1.0** command starts EIGRP on the serial 0/0/1 interface, and allows Router R1 to advertise this network.

Example 2-7 shows the configuration of the R2 router.

Example 2-7. Configuration for Router R2 in Figure 2-20

```
R2#show running-config
```

<output omitted>

```
interface FastEthernet0/0
```

```
ip address 172.17.2.2 255.255.255.0
```

<output omitted>

```
interface Serial0/0/1
```

```
bandwidth 64
```

```
ip address 192.168.1.102 255.255.255.224
```

<output omitted>

```
router eigrp 100
```

```
network 172.17.2.0 0.0.0.255
```

```
network 192.168.1.0
```

EIGRP is also enabled in autonomous system 100 on the R2 router. The **network 172.17.2.0 0.0.0.255** command starts EIGRP on the Fast Ethernet 0/0 interface and allows Router R2 to advertise this network. With the wildcard mask used, this command specifies that only interfaces on the 172.17.2.0/24 subnet will participate in EIGRP. Note, however, the full Class B network 172.17.0.0 will be advertised, because EIGRP automatically summarizes routes on the major network boundary by default. The **network 192.168.1.0** command starts EIGRP on the serial 0/0/1 interface and allows Router R2 to advertise this network.

Verifying EIGRP Neighbors

The show ip eigrp neighbors command displays the contents of the IP EIGRP neighbor table. Example 2-8 provides R1's IP EIGRP neighbor table.

Example 2-8. Sample Output for the show ip eigrp neighbors Command

```
R1#show ip eigrp neighbors
```

```
IP-EIGRP neighbors for process 100
```

H	Address	Interface	Hold (sec)	Uptime (ms)	SRTT Cnt	RTO Num	Q	Seq
0	192.168.1.102	Se0/0/1	10	00:07:22	10	2280	0	5

```
R1#
```

Note

The “EIGRP Neighbors” section, earlier in this chapter, provides a full description of the output from the show ip eigrp neighbors command.

Detailed neighbor information can be examined with the show ip eigrp neighbor detail command, as shown in Example 2-9. The additional information provided in this command includes the number of items a packet has been retransmitted (two in this example), the number of times an attempt was made to retransmit a packet (two in this example), the packets that are currently waiting to be sent (R1 has three updates waiting to be sent in this example), and the neighboring router IOS version (12.4 in this example). The sequence number (seq in the example) increments each time a query, update, or reply packet is sent, whereas the serial number (ser in the example) increments each time the topology table changes.

Example 2-9. Sample Output for the *show ip eigrp neighbors detail* Command

```
R1#show ip eigrp neighbors detail
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold Uptime  SRTT  RTO  Q  Seq
      (sec)          (ms)      Cnt Num
0  192.168.1.102 Se0/0/1    14  00:17:55  0   4500 3  274
Last startup serial 569
Version 12.4/1.0, Retrans: 2, Retries: 2, Waiting for Init Ack
UPDATE seq 307 ser 29-569 Sent 8924 Init Sequenced
UPDATE seq 310 ser 570-573 Sequenced
UPDATE seq 312 ser 574-578 Sequenced
```

Verifying EIGRP Routes

To verify that the router recognizes EIGRP routes for any neighbors, use the show ip route eigrp command, as shown in Example 2-10. Example 2-11 exhibits the show ip route command, which displays the full IP routing table, including the EIGRP routes.

Example 2-10. *show ip route eigrp* Command Output

```
R1#show ip route eigrp
D 172.17.0.0/16 [90/40514560] via 192.168.1.102, 00:07:01, Serial0/0/1
   172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
D   172.16.0.0/16 is a summary, 00:05:13, Null0
   192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
D   192.168.1.0/24 is a summary, 00:05:13, Null0
```

Example 2-11. *show ip route* Command Output

```
R1#show ip route
<output omitted>
Gateway of last resort is not set
D 172.17.0.0/16 [90/40514560] via 192.168.1.102, 00:06:55, Serial0/0/1
   172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
D   172.16.0.0/16 is a summary, 00:05:07, Null0
C   172.16.1.0/24 is directly connected, FastEthernet0/0
   192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C   192.168.1.96/27 is directly connected, Serial0/0/1
D   192.168.1.0/24 is a summary, 00:05:07, Null0
```

Using the highlighted line in Example 2-10 as an example, the fields in the routing table are interpreted as follows:

- Internal EIGRP routes are identified with a D in the far left column. (External EIGRP routes, not shown in this example, are identified with a D EX in the far left column.)
- The next column is the network number (172.17.0.0/16 in this example).

- After each network number is a field in brackets (90/40514560 in this example). The second number in brackets is the EIGRP metric. As discussed in the “EIGRP Metric Calculation” section, earlier in this chapter, the default EIGRP metric is the least-cost bandwidth plus the accumulated delays. The EIGRP metric for a network is the same as its FD in the EIGRP topology table.

The first number, 90 in this case, is the administrative distance. Recall from Chapter 1 that the administrative distance is used to select the best path when a router learns two or more routes to exactly the same destination from different routing sources. For example, consider that this router uses Routing Information Protocol (RIP) and EIGRP and that RIP has a route to network 172.17.0.0 that is three hops away. The router, without the administrative distance, cannot compare three hops to an EIGRP metric of 40,514,560. The router does not know the bandwidth associated with hops, and EIGRP does not use hop count as a metric. To correct this problem, Cisco established an administrative distance for each routing protocol: the lower the value, the more preferred the route is. By default, EIGRP internal routes have an administrative distance of 90, and RIP has an administrative distance of 120. Because EIGRP has a metric based on bandwidth and delay, it is preferred over RIP’s hop count metric. As a result, in this example, the EIGRP route would be installed in the routing table.

Note

Remember that routers use the administrative distance only if the two routes are to the exact same destination (address and mask). For example, a router will choose a RIP route over an EIGRP route if the RIP route is a more specific route than the EIGRP route.

- The next field, via 192.168.1.102 in this example, is the address of the next-hop router to which this router passes packets destined for 172.17.0.0/16. The next-hop address in the routing table is the same as the successor in the EIGRP topology table.
- The route also has a time associated with it (00:07:01 in this example). This is the length of time Because EIGRP last advertised this network to this router. EIGRP does not refresh routes periodically. It resends routing information only when neighbor adjacencies change.
- The interface, Serial 0/0/1 in this case, indicates the interface out which packets for 172.17.0.0/16 are sent.

Notice that the routing table includes routes, to null0, for the advertised (summarized) routes. Cisco IOS Software automatically puts these routes in the table when it autosummarizes. They are called summary routes. Null 0 is a directly connected, software-only interface. The use of the null0 interface prevents the router from trying to forward traffic to other routers in search of a more precise, longer match. For example, if the R1 router in Figure 2-20 receives a packet to an unknown subnet that is part of the summarized range—172.16.3.5 for example—the packet matches the summary route based on the longest match. The packet is forwarded to the null0 interface (in other words, it is dropped, or sent to the bit bucket), which prevents the router from forwarding the packet to a default route and possibly creating a routing loop.

Verifying EIGRP Operations

This section describes commands used to verify EIGRP operation.

show ip protocols Example

Use the **show ip protocols** command to provide information about any and all dynamic routing protocols running on the router.

As shown in Example 2-12, the command output displays that EIGRP process (autonomous system) 100 is running, and indicates any route filtering occurring on EIGRP outbound or inbound updates. It also identifies whether EIGRP is generating a default network or receiving a default network in EIGRP updates and provides information about additional settings for EIGRP, such as K values, hop count, and variance.

Example 2-12. *show ip protocols* Command Output

Code View: Scroll / Show All

R1#**show ip protocols**

Routing Protocol is "eigrp 100"

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 metric weight K1=1, K2=0, K3=1, K4=0, K5=0

```

EIGRP maximum hopcount 100
EIGRP maximum metric variance 1
Redistributing: eigrp 100
EIGRP NSF-aware route hold timer is 240s
Automatic network summarization is in effect
Automatic address summarization:
  192.168.1.0/24 for FastEthernet0/0
    Summarizing with metric 40512000
  172.16.0.0/16 for Serial0/0/1
    Summarizing with metric 28160
Maximum path: 4
Routing for Networks:
  172.16.1.0/24
  192.168.1.0
Routing Information Sources:
  Gateway        Distance    Last Update
  (this router)      90      00:09:38
  Gateway        Distance    Last Update
  192.168.1.102      90      00:09:40
Distance: internal 90 external 170

```

Note

Two routers must have identical K values for EIGRP to establish an adjacency. The **show ip protocols** command is helpful in determining the current K value settings before an adjacency is attempted.

The output in Example 2-12 also indicates that automatic summarization is enabled and that the router is allowed to load balance over a maximum of four paths. Cisco IOS Software allows configuration of up to 16 paths for equal-cost load balancing, using the maximum-paths router configuration command.

The networks for which the router is routing are also displayed. As shown in Example 2-12, the format of the output varies, depending on the use of the wildcard mask in the network command. If a wildcard mask is used, the network address is displayed with a prefix length. If a wildcard mask is not used, the Class A, B, or C major network is displayed.

The routing information source portion of this command output identifies all other routers that have an EIGRP neighbor relationship with this router. The **show ip eigrp neighbors** command provides a detailed display of EIGRP neighbors.

The **show ip protocols** command output also provides the two administrative distances for EIGRP. An administrative distance of 90 applies to networks from other routers inside the same autonomous system. These are considered internal networks. An administrative distance of 170 applies to networks introduced to this EIGRP autonomous system through redistribution. These are called external networks.

show ip eigrp interfaces Example

The show ip eigrp interfaces command displays information about interfaces configured for EIGRP. Example 2-13 demonstrates show ip eigrp interfaces command output. This output includes the following key elements:

- **Interface**— Interface over which EIGRP is configured
- **Peers**— Number of directly connected EIGRP neighbors
- **Xmit Queue Un/Reliable**— Number of packets remaining in the Unreliable and Reliable retransmit queues
- **Mean SRTT**— Mean SRTT interval, in milliseconds
- **Pacing Time Un/Reliable**— Pacing time used to determine when EIGRP packets should be sent out the interface (for unreliable and reliable packets)

- **Multicast Flow Timer**— Maximum number of seconds that the router will wait for an ACK packet after sending a multicast EIGRP packet, before switching from multicast to unicast
- **Pending Routes**— Number of routes in the packets in the retransmit queue waiting to be sent

Example 2-13. *show ip eigrp interfaces* Command Output

Code View: Scroll / Show All

R1#**show ip eigrp interfaces**

IP-EIGRP interfaces for process 100

Interface	Peers	Xmit Queue Un/Reliable	Mean SRTT	Pacing Time Un/Reliable	Multicast Flow Timer	Pending Routes
Fa0/0	0	0/0	0	0/10	0	0
Se0/0/1	1	0/0	10	10/380	424	0

show ip eigrp topology Example

Another command used to verify EIGRP operations is the show ip eigrp topology command. This command displays the DUAL states and helps troubleshoot possible DUAL issues. Example 2-14 demonstrates output generated from this command.

Example 2-14. *show ip eigrp topology* Command Output

R1#**show ip eigrp topology**

IP-EIGRP Topology Table for AS(100)/ID(192.168.1.101)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
r - reply Status, s - sia Status

P 192.168.1.96/27, 1 successors, FD is 40512000
via Connected, Serial0/0/1

P 192.168.1.0/24, 1 successors, FD is 40512000
via Summary (40512000/0), Null0

P 172.16.0.0/16, 1 successors, FD is 28160
via Summary (28160/0), Null0

P 172.16.1.0/24, 1 successors, FD is 28160
via Connected, FastEthernet0/0

P 172.17.0.0/16, 1 successors, FD is 40514560
via 192.168.1.102 (40514560/28160), Serial0/0/1

The command output illustrates that Router R1 has a router ID of 192.168.1.101 and is in autonomous system 100. The EIGRP router ID is chosen as the highest IP address on an active interface on the router, unless loopback interfaces are configured, in which case it is the highest IP address assigned to a loopback interface. Alternatively, if the **eigrp router-id ip-address** router configuration command is used, it will override the use of the address of a physical or loopback interface as the router ID.

The command output also lists the networks known by this router through the EIGRP routing process. The codes used in the first column of this output indicate the state of the entry. Passive and Active refer to the EIGRP state with respect to this destination. Update, Query, and Reply refer to the type of packet being sent. The codes are as follows:

- **Passive (P)**— This network is available, and installation can occur in the routing table. Passive is the correct state for a stable network, indicating that no EIGRP computations are being performed for this route.
- **Active (A)**— This network is currently unavailable, and installation cannot occur in the routing table. Being active means that outstanding queries exist for this network, indicating that EIGRP computations are being performed for this route.
- **Update (U)**— This network is being updated (indicating that an update packet is being sent). This code also applies if the router is waiting for an acknowledgment for this update packet.

- **Query (Q)**— There is an outstanding query packet for this network, indicating that a query packet was sent. This code also applies if the router is waiting for an acknowledgment for a query packet.
- **Reply (R)**— The router is generating a reply for this network, indicating that a reply packet was sent, or is waiting for an acknowledgment for the reply packet.
- **Reply status (r)**— Indicates the flag that is set after the software has sent a query and is waiting for a reply.
- **Stuck-in-active (s)**— There is an EIGRP convergence problem for this network. (The “Stuck-in-Active Connections in EIGRP” section, later in this chapter, describes this problem and how it can be prevented.)

The number of successors available for a route is indicated in the command output. The number of successors corresponds to the number of best routes with equal cost. All networks in Example 2-14 have one successor.

For each network, the FD is listed next, followed by an indication of how the route was learned, such as the next-hop address if the route was learned via another router. Next is a field in brackets. The first number in the brackets is the FD for that network through the next-hop router, and the second number in the brackets is the AD from the next-hop router to the destination network.

show ip eigrp traffic Example

To display the number of various EIGRP packets sent and received, use the show ip eigrp traffic command, as illustrated in Example 2-15. For example, in this network, Router R1 has sent 429 hello messages and received 192 hello messages, has sent 4 updates and received 4 updates, has sent 1 query and received 0 queries, has sent 0 replies and received 1 reply, and has sent 4 ACKs and received 3 ACKs.

Example 2-15. *show ip eigrp traffic* Command Output

```
R1#show ip eigrp traffic
IP-EIGRP Traffic Statistics for AS 100
  Hellos sent/received: 429/192
  Updates sent/received: 4/4
  Queries sent/received: 1/0
  Replies sent/received: 0/1
  Acks sent/received: 4/3
  Input queue high water mark 1, 0 drops
  SIA-Queries sent/received: 0/0
  SIA-Replies sent/received: 0/0
  Hello Process ID: 113
  PDM Process ID: 73
```

debug eigrp packets Examples

You can use the debug eigrp packets command to verify EIGRP connectivity. This command displays the types of EIGRP packets sent and received by the router on which this command is executed. Different packet types can be selected for individual or group display. Example 2-16 shows some output from this command on R2, when an interface on R1 comes up.

Example 2-16. *debug eigrp packets* Command Output on R2 When a Neighbor’s Interface Comes Up

```
Code View: Scroll / Show All
R2#debug eigrp packets
EIGRP Packets debugging is on
  (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
  SIAREPLY)
*May 11 04:02:55.821: EIGRP: Sending HELLO on Serial0/0/1
*May 11 04:02:55.821:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibbQ un/rely 0/0
R2#
*May 11 04:02:58.309: EIGRP: Received HELLO on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:58.309:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/0
```

```

*May 11 04:02:58.585: EIGRP: Sending HELLO on FastEthernet0/0
*May 11 04:02:58.585:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibbQ un/rely 0/0
*May 11 04:02:59.093: EIGRP: Received UPDATE on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:59.093:  AS 100, Flags 0x0, Seq 5/4 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/0
*May 11 04:02:59.093: EIGRP: Enqueueing ACK on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:59.093:  Ack seq 5 iibbQ un/rely 0/0 peerQ un/rely 1/0
*May 11 04:02:59.097: EIGRP: Sending ACK on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:59.097:  AS 100, Flags 0x0, Seq 0/5 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 1/0
*May 11 04:02:59.109: EIGRP: Enqueueing UPDATE on Serial0/0/1 iibbQ un/rely 0/1
serno 9-9
*May 11 04:02:59.113: EIGRP: Enqueueing UPDATE on Serial0/0/1 nbr 192.168.1.101
iibbQ un/rely 0/0 peerQ un/rely 0/0 serno 9-9
*May 11 04:02:59.113: EIGRP: Sending UPDATE on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:59.113:  AS 100, Flags 0x0, Seq 5/5 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/1 serno 9-9
*May 11 04:02:59.133: EIGRP: Received ACK on Serial0/0/1 nbr 192.168.1.101
*May 11 04:02:59.133:  AS 100, Flags 0x0, Seq 0/5 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/1
*May 11 04:02:59.133: EIGRP: Serial0/0/1 multicast flow blocking cleared
R2#
*May 11 04:03:00.441: EIGRP: Sending HELLO on Serial0/0/1
*May 11 04:03:00.441:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibbQ un/rely 0/0
R2#
*May 11 04:03:03.209: EIGRP: Received HELLO on Serial0/0/1 nbr 192.168.1.101
*May 11 04:03:03.209:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/0

```

The debug eigrp packets command traces transmission and receipt of EIGRP packets. The output in Example 2-16 shows normal transmission and receipt of EIGRP packets. The serial link is an HDLC point-to-point link. Therefore, the default hello time interval is 5 seconds. Hello packets are sent unreliably, so the sequence number (Seq) does not increment for hello packets.

In this sample output, when R2 receives an update from R1, values appear in the sequence number field. Seq 5/4 indicates that 192.168.1.101 (R1) is sending this packet as sequence number 5 to R2 and that sequence number 4 has been received from R2 by neighbor R1. R1 is expecting to receive sequence number 5 in the next reliable packet from R2.

R2 returns an ACK packet with Seq 0/5. The acknowledgment is sent as an unreliable packet. The neighbor unreliable/reliable flag (un/rel 1/0) is set, which means that the acknowledgment was sent in response to a reliable packet.

The serial number (serno 9-9) reflects the number of changes that the two neighbors register in their EIGRP topology tables. The sequence number increments each time a query, update, or reply packet is sent, whereas the serial number increments each time the topology table changes. A single update can contain more than 100 networks, for example if they all become unavailable. Therefore, if the topology table has more than 100 changes, the serial number increases substantially, but the sequence number may only increase by 1.

When an interface on R1 is shut down, the resulting output on R2 is shown in Example 2-17. R1 sends a query packet to R2 to determine whether R2 knows a path to the lost network. R2 responds with an ACK packet to acknowledge the query packet—a reliable packet must be explicitly acknowledged with an ACK packet. R2 also responds to the query with a reply packet. The serial number reference (10-12) represents

the number of changes to the topology table since the start of the neighbor relationship between these two EIGRP neighbors.

Example 2-17. *debug eigrp packets* Command Output on R2 When a Neighbor's Interface Is Shut Down

Code View: Scroll / Show All

R2#**debug eigrp packets**

```
*May 11 04:20:43.361: EIGRP: Received QUERY on Serial0/0/1 nbr 192.168.1.101
*May 11 04:20:43.361: AS 100, Flags 0x0, Seq 6/5 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/0
*May 11 04:20:43.361: EIGRP: Enqueueing ACK on Serial0/0/1 nbr 192.168.1.101
*May 11 04:20:43.361: Ack seq 6 iibbQ un/rely 0/0 peerQ un/rely 1/0
*May 11 04:20:43.365: EIGRP: Sending ACK on Serial0/0/1 nbr 192.168.1.101

*May 11 04:20:43.365: AS 100, Flags 0x0, Seq 0/6 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 1/0
*May 11 04:20:43.373: EIGRP: Enqueueing REPLY on Serial0/0/1 nbr 192.168.1.101
iibbQ un/rely 0/1 peerQ un/rely 0/0 serno 10-12
*May 11 04:20:43.377: EIGRP: Requeued unicast on Serial0/0/1
R2#
*May 11 04:20:43.381: EIGRP: Sending REPLY on Serial0/0/1 nbr 192.168.1.101
*May 11 04:20:43.381: AS 100, Flags 0x0, Seq 6/6 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/1 serno 10-12
*May 11 04:20:43.405: EIGRP: Received ACK on Serial0/0/1 nbr 192.168.1.101
*May 11 04:20:43.405: AS 100, Flags 0x0, Seq 0/6 idbQ 0/0 iibbQ un/rely 0/0
peerQ un/rely 0/1
```

debug ip eigrp Examples

You can use the debug ip eigrp command to verify EIGRP operation. This command displays IP EIGRP packets that this router sends and receives. Example 2-18 shows the contents of the updates that are reported when the debug ip eigrp command is used on R2 to monitor when an interface on R1 comes up.

Example 2-18. *debug ip eigrp* Command Output on R2 When a Neighbor's Interface Comes Up

Code View: Scroll / Show All

R2#**debug ip eigrp**

IP-EIGRP Route Events debugging is on

R2#

```
*May 11 04:24:05.261: IP-EIGRP(Default-IP-Routing-Table:100): Processing incoming
UPDATE packet
*May 11 04:24:05.261: IP-EIGRP(Default-IP-Routing-Table:100): Int 192.168.1.0/24
M 4294967295 - 40000000 4294967295 SM 4294967295 - 40000000 4294967295
*May 11 04:24:05.261: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.0.0/16
M 40514560 - 40000000 514560 SM 28160 - 25600 2560
*May 11 04:24:05.261: IP-EIGRP(Default-IP-Routing-Table:100): route installed
for 172.16.0.0 ( )
*May 11 04:24:05.277: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.0.0/16
metric 40514560 - 40000000 514560
```

In this example, an internal route (indicated by Int) for 172.16.0.0/16 is advertised to R2.

Recall that by default the EIGRP metric is equal to the bandwidth plus the delay. The EIGRP process uses the source metric (SM) information in the update to calculate the AD and place it in the EIGRP topology table. In this example, the SM information is SM 28160 – 25600 2560, which means the source metric (the AD) = 28160 = 25600 (the bandwidth) + 2560 (the delay).

The EIGRP metric calculation for the total delay uses the metric (M) information in the update. In this example, the M information is M 40514560 – 40000000 514560, which means the metric (the FD) = 40514560 = 40000000 (the bandwidth) + 514560 (the delay).

The EIGRP metric for this route is equal to the FD and, therefore, is 40,514,560.

Example 2-19 illustrates what occurs when R2 processes an incoming query packet for network 172.16.0.0/16 when the interface on the neighboring router (R1) that leads to that network is shut down. Note that comments (preceded by an exclamation point [!]) have been added to this output for easier understanding.

Example 2-19. *debug ip eigrp* Command Output on R2 When a Neighbor's Interface Is Shut Down

Code View: Scroll / Show All

R2#**debug ip eigrp**

IP-EIGRP Route Events debugging is on

R2#

! An interface on EIGRP neighbor R1 was shutdown

! R2 receives a query looking for a lost path from R1

*May 11 04:35:44.281: IP-EIGRP(Default-IP-Routing-Table:100): Processing incoming QUERY packet

*May 11 04:35:44.281: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.1.0/24
M 4294967295 - 0 4294967295 SM 4294967295 - 0 4294967295

*May 11 04:35:44.281: IP-EIGRP(Default-IP-Routing-Table:100): Int 192.168.1.0/24
M 4294967295 - 0 4294967295 SM 4294967295 - 0 4294967295

*May 11 04:35:44.281: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.0.0/16
M 4294967295 - 0 4294967295 SM 4294967295 - 0 4294967295

! R2 realizes that if it cannot use R1 for this network then

! it does not have an entry in the routing table for this network
continues

*May 11 04:35:44.281: IP-EIGRP(Default-IP-Routing-Table:100): 172.16.0.0/16 routing table not updated thru 192.168.1.101

R2#

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): 172.16.1.0/24 - not in IP routing table

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.1.0/24
metric 4294967295 - 0 4294967295

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): 192.168.1.0/24 -
poison advertise out Serial0/0/1

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): Int 192.168.1.0/24
metric 40512000 - 40000000 512000

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): 172.16.0.0/16 - not
in IP routing table

! R2 sends an update to R1 saying it does not know how to reach that network either

*May 11 04:35:44.301: IP-EIGRP(Default-IP-Routing-Table:100): Int 172.16.0.0/16
metric 4294967295 - 40000000 4294967295

R2#

The neighbor previously advertised 172.16.0.0/16 to this router. The query performs the following two functions:

- R2 discovers that its neighbor no longer knows how to get to network 172.16.0.0/16. The metric value (4,294,967,295) is the highest possible value for a 32-bit number—it indicates that the route is unreachable. R2 removes this entry from the EIGRP topology table and looks for alternative EIGRP routes.
- The **debug** output indicates that the routing table is not updated, which means that EIGRP did not find an alternative route to the network. The next statement verifies that the EIGRP process has removed the old route and that the route is not in the IP routing table. R2 then informs the neighbor that it does not have a path to this network either.

Using the passive-interface Command with EIGRP

There are times when you must or want to include a subnet in a routing protocols' **network** command, although you do not want the interface on which the subnet is connected to participate in the routing protocol.

The **passive-interface** {type number} | **default** router configuration command prevents a routing protocol's routing updates from being sent through the specified router interface. This command is used to set either a particular interface or all router interfaces to passive; use the **default** option to set all router interfaces to passive.

Table 2-4 describes the parameters of this command.

Table 2-4. *passive-interface* Command

Parameter	Description
type number	Specifies the type of interface and interface number that will not send routing updates (or establish neighbor relationships for link-state routing protocols and EIGRP)
default	Sets all interfaces on the router as passive by default

For EIGRP, the **passive-interface** command does the following:

- It prevents a neighbor relationship from being established over a passive interface.
- It stops routing updates from being processed or sent over passive interface.
- It allows a subnet on a passive interface to be announced in an EIGRP process.

When you use the **passive-interface** command with EIGRP, hello messages are not sent out of the specified interface. Neighboring router relationships do not form with other routers that can be reached through that interface (because the hello protocol is used to verify bidirectional communication between routers). Because no neighbors are found on an interface, no other EIGRP traffic is sent.

Recall that the **network** command defines the interfaces over which EIGRP will attempt to establish neighbor relationships and the networks that will be advertised to EIGRP neighbors. Configuring an interface as passive only disables the neighbor relationship establishment. The router will still advertise the network to its EIGRP neighbors.

In the example network shown in Figure 2-20 used earlier, Routers R1 and R2 have no EIGRP neighbors available over their Fast Ethernet 0/0 interfaces, so there is no need to try to establish adjacencies over those interfaces. If EIGRP packets are sent on these interfaces, they would be ignored, but would still consume bandwidth and CPU resources.

In the original configurations of Router R1 and R2 shown in Example 2-6 and Example 2-7, network commands were configured for the Fast Ethernet 0/0 interfaces to allow the router to advertise those subnets to its neighbor. Example 2-20 provides alternate EIGRP configurations for both routers.

Example 2-20. Passive-Interface Configurations for Routers in Figure 2-20

```
R1#
router eigrp 110
  passive-interface FastEthernet0/0
  network 172.16.1.0 0.0.0.255
  network 192.168.1.0
R2#
router eigrp 110
  passive-interface FastEthernet0/0
```

```
network 172.17.2.0 0.0.0.255
network 192.168.1.0
```

Note

EIGRP will not bring up adjacencies on a passive interface even if a neighbor command is configured over that interface. The neighbor command is described in the "EIGRP Unicast Neighbors" section, later in this chapter.

In Internet service providers (ISPs) and large enterprise networks, many distribution routers have more than 100 interfaces. Before the **passive-interface default** command was introduced in Cisco IOS Software Release 12.0, network administrators would configure the routing protocol on all interfaces and then manually set the **passive-interface** command on the interfaces on which they did not require adjacencies to be established. However, this solution meant entering many **passive-interface** commands. A single **passive-interface default** command can now be used to set all interfaces to passive by default. To enable routing on individual interfaces where you require adjacencies to be established, use the **no passive-interface** command.

Example 2-21 provides an alternate configuration for Router R1, using the passive-interface default command to make all interfaces passive. The no passive-interface serial0/0/1 command allows EIGRP adjacencies to be established over Serial 0/0/1.

Example 2-21. Alternate Passive-Interface Configurations for Router R1 in Figure 2-20

```
R1#
router eigrp 110
  passive-interface default
  no passive-interface Serial0/0/1
  network 172.16.1.0 0.0.0.255
  network 192.168.1.0
```

To verify EIGRP operation when using passive interfaces, the following commands are used:

- The **show ip eigrp neighbors** command, to verify that all neighbor relationships are established.
- The show ip protocols command, to see which interfaces are configured as passive interface. The output in Example 2-22 for Router R1 shows that interface Fast Ethernet 0/0 is defined as a passive interface.

Example 2-22. show ip protocols Command Output on Router R1 in Figure 2-20

```
R1#show ip protocols
Routing Protocol is "eigrp 110"
  <output omitted>
  Automatic network summarization is in effect
  Automatic address summarization:
    172.16.0.0/16 for Serial0/0/1
    Summarizing with metric 28160
  Maximum path: 4
  Routing for Networks:
    172.16.1.0/24
    192.168.1.0
  Passive Interface(s):
    FastEthernet0/0
  <output omitted>
```

Propagating an EIGRP Default Route

Default routes usually decrease the size of the routing tables in routers that receive them. For example, routers on stub networks or at the access layer do not typically need to know all the routes in the entire

network. Instead, they can use a default route to send traffic to routers that have more detailed routing tables.

A statically configured default route can be created with the **ip route 0.0.0.0 0.0.0.0 {next-hop | interface [next-hop]}** global configuration command. The *interface* is an outgoing interface through which all packets with unknown destinations will be forwarded. The *next-hop* is the IP address to which packets with unknown destinations will be forwarded. (Notice that both the interface and next-hop address can be specified in this command.)

Alternatively, any major network residing in the local routing table can become an EIGRP default route when used in the **ip default-network network-number** global configuration command. A router configured with this command considers the *network-number* the last-resort gateway that it will announce to other routers with the exterior flag set. The network must be reachable by the router that uses this command before it announces it as a candidate default route to other EIGRP routers. The network number in this command must also be passed to other EIGRP routers so that those routers can use this network as their default network and set their gateway of last resort to this default network. This means that the network must either be an EIGRP-derived network in the routing table, or be generated with a static route and redistributed into EIGRP.

Note

In EIGRP default routes cannot be directly injected (as they can in OSPF with the **default-information originate** command).

Note

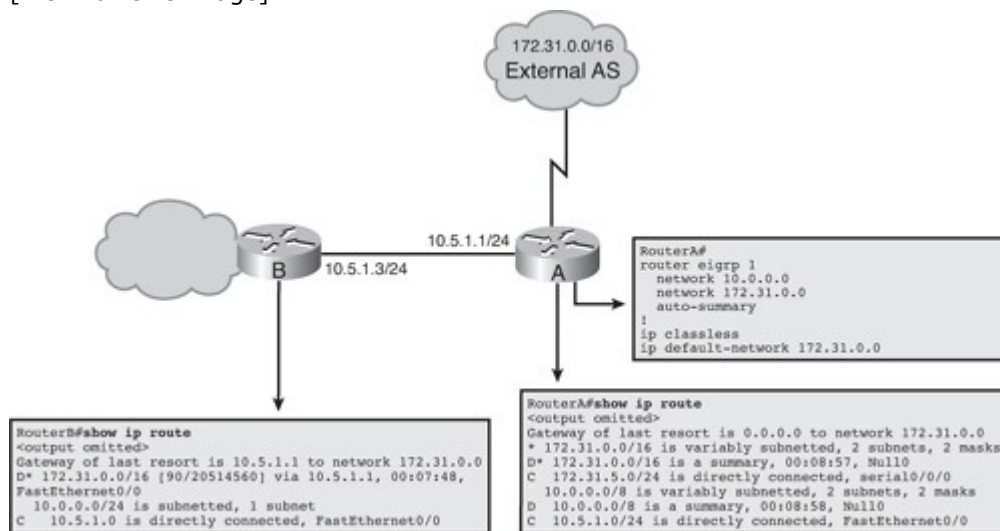
Any route residing in the routing table, including non-EIGRP routes, can be marked as a default candidate.

Multiple default networks can be configured. Downstream routers then use the EIGRP metric to determine the best default route. When the best default route is selected, the router sets the gateway of last resort to the next-hop address of the selected candidate, unless the best candidate route is one of the router's directly connected routes.

For example, in Figure 2-21, Router A is directly attached to external network 172.31.0.0/16. Router A is configured with the 172.31.0.0 network as a candidate default network using the **ip default-network 172.31.0.0** command. Router A also has that network listed in a **network** command under the EIGRP process and, therefore, passes it to Router B.

Figure 2-21. EIGRP *ip default-network* Sample Network.

[View full size image]



To verify default network information, use the **show ip route** command to view the routing table. On Router B, the EIGRP-learned 172.31.0.0 network is flagged as a candidate default network (indicated by the * in the routing table). Router B also sets the gateway of last resort as 10.5.1.1 (Router A) to reach the default network of 172.31.0.0. Router A also has the 172.31.0.0 network flagged as a default network and has its gateway of last resort set.

Note

In earlier versions of the IOS software, the router on which the `ip default-network` command was configured would not set the gateway of last resort, so it appeared that this command did not benefit that router directly. Figure 2-21 illustrates that it now does set the gateway of last resort as 0.0.0.0 to the network specified in the `ip default-network` command.

Note

When you configure the **`ip default-network`** command and specify a subnet, a static route (the **`ip route`** command) is generated in the router's configuration; however, the IOS does not display a message to indicate that this has been done. The entry appears as a static route in the routing table of the router where the command is configured. This can be confusing when you want to remove the default network. The configuration must be removed with the **`no ip route`** command, not with the **`no ip default-network`** command.

Note

EIGRP (and IGRP) behave differently than RIP when using the `ip route 0.0.0.0 0.0.0.0` command. For example, EIGRP does not redistribute the 0.0.0.0 0.0.0.0 default route by default. However, if the `network 0.0.0.0` command is added to the EIGRP configuration, it redistributes a default route as a result of the `ip route 0.0.0.0 0.0.0.0 interface` command (but not as a result of the `ip route 0.0.0.0 0.0.0.0 address` or `ip default-network` command). For example, the partial configuration shown in Example 2-23 results in the 0.0.0.0 route being passed to the router's EIGRP neighbors.

Example 2-23. EIGRP Passes a Default Route Only if It Is Configured to Do So

```
Router#show run
<output omitted>
interface serial 0/0/0
  ip address 10.1.1.1 255.255.255.0
!
ip route 0.0.0.0 0.0.0.0 serial 0/0/0
!
router eigrp 100
  network 0.0.0.0
<output omitted>
```

EIGRP Route Summarization

Some EIGRP features, such as automatically summarizing routes at a major network boundary, have distance vector and classful characteristics. Traditional distance vector protocols, which are classful routing protocols, must summarize at network boundaries. They cannot presume the mask for networks that are not directly connected, because masks are not exchanged in the routing updates.

Summarizing routes at classful major network boundaries creates smaller routing tables. Smaller routing tables, in turn, make the routing update process less bandwidth intensive (because updates are smaller), and less CPU intensive (because there is less update information to process). Cisco distance vector routing protocols have autosummarization enabled by default. EIGRP automatic summarization on the major network boundary can be turned on or off.

The inability to create summary routes at arbitrary boundaries with a major network has been a drawback of distance vector protocols since their inception. EIGRP has added functionality to allow administrators to create one or more summary routes within a network on any bit boundary, on any router within the network, as long as a more specific route exists in the routing table. When the last specific route of the summary goes away, the summary route is deleted from the routing table.

It is important to note that the minimum metric of the specific routes is used as the metric of the summary route.

When summarization is configured on a router's interface, a summary route is added to that router's routing table, with the route's next-hop interface set to null0—a directly connected, software-only interface. As described earlier for autosummarization, the use of the null0 interface prevents the router from trying to forward traffic to other routers in search of a more precise, longer match, thus preventing traffic from looping within the network. For example, if the summarizing router receives a packet to an unknown subnet that is

part of the summarized range, the packet matches the summary route based on the longest match, and the packet is forwarded to the null0 interface and therefore is dropped.

For effective summarization, blocks of contiguous addresses (subnets) should funnel back to a common router so that a single summary route can be created and then advertised. In other words, the IP addressing plan for the network must have been created properly, with summarization in mind (it is unlikely that you would be lucky enough to be able to summarize randomly-assigned subnets). The number of subnets that can be represented by a summary route is calculated by the formula 2^n , where n equals the difference in the number of bits between the summary and subnet masks. For example, if the summary mask contains 3 fewer bits than the subnet mask, eight ($2^3 = 8$) subnets can be aggregated into one advertisement.

For example, if network 10.0.0.0 is divided into /24 subnets and some of these subnets are summarized to the summarization block 10.1.8.0/21, the difference between the /24 networks and the /21 summarizations is 3 bits. Therefore, $2^3 = 8$ subnets can be aggregated. The summarized subnets range from 10.1.8.0/24 through 10.1.15.0/24.

When creating summary routes, the administrator needs to specify the IP address of the summary route and the summary mask. The Cisco IOS handles the details of proper implementation, such as metrics, loop prevention, and removal of the summary route from the routing table if none of the more specific routes are valid.

Configuring Manual Route Summarization

In some cases you may want to turn off the EIGRP automatic summarization feature. For example, if you have discontinuous subnets, you need to disable autosummarization. Note that an EIGRP router does not perform automatic summarization of networks in which it does not participate.

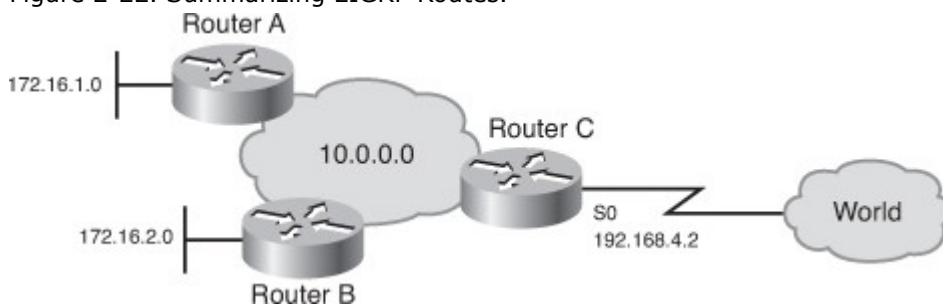
To turn off automatic summarization, use the `no auto-summary` router configuration command. Use the `ip summary-address eigrp as-number address mask [admin-distance]` interface configuration command to manually create a summary route at an arbitrary bit boundary, as long as a more specific route exists in the routing table. Table 2-5 summarizes the parameters for this command.

Table 2-5. *ip summary-address eigrp* Command Parameters

Parameter	Description
as-number	EIGRP autonomous system number.
address	The IP address being advertised as the summary address. This address does not need to be aligned on Class A, B, or C boundaries.
mask	The IP subnet mask used to create the summary address.
admin-distance	(Optional) Administrative distance. A value from 0 to 255.

For example, Figure 2-22 shows a discontinuous network 172.16.0.0. By default, both Routers A and B summarize routes at the classful boundary. As a result, Router C would have two equally good routes to network 172.16.0.0 and would perform load balancing between Router A and Router B. This would not be correct routing behavior.

Figure 2-22. Summarizing EIGRP Routes.



As shown in Example 2-24, you can disable the automatic route summarization on Router A. The same configuration would be done on Router B. With this configuration, Router C knows precisely that 172.16.1.0 is reached via Router A and that 172.16.2.0 is reached only via Router B. The routing tables of the routers in the 10.0.0.0 network, including Router C, now include these discontinuous subnets.

Example 2-24. Turning Off EIGRP Autosummarization on Router A (and Router B) in Figure 2-22

```
RouterA(config)#router eigrp 1
RouterA(config-router)#network 10.0.0.0
RouterA(config-router)#network 172.16.0.0
RouterA(config-router)#no auto-summary
```

An EIGRP router autosummarizes routes for only networks to which it is attached. If a network was not autosummarized at the major network boundary, as is the case in this example on Routers A and B because autosummarization is turned off, all the subnet routes are carried into Router C's routing table. Router C will not autosummarize the 172.16.1.0 and 172.16.2.0 subnets because it does not own the 172.16.0.0 network. Therefore, Router C would send routing information about the 172.16.1.0 subnet and the 172.16.2.0 subnet to the WAN.

Forcing a summary route out Router C's interface s0/0/0, as shown in Example 2-25, helps reduce route advertisements about network 172.16.0.0 to the WAN.

Example 2-25. Forcing Summarization on Router C in Figure 2-22

```
RouterC#show run
<output omitted>
router eigrp 1
  network 10.0.0.0
  network 192.168.4.0
!
<output omitted>
int s0/0/0
  ip address 192.168.4.2 255.255.255.0
  ip summary-address eigrp 1 172.16.0.0 255.255.0.0
<output omitted>
```

Note

You can use the ip summary-address eigrp as-number 0.0.0.0 0.0.0.0 command to inject a default route to a neighbor, as an alternative to the methods described earlier in the "Propagating an EIGRP Default Route" section. However, the automatically generated route to null0 may cause problems in some topologies.

Verifying Manual Route Summarization

To verify manual route summarization, examine the IP routing tables. Example 2-26 illustrates Router C's routing table. Router C has both 172.16.1.0 and 172.16.2.0, the discontinuous subnets, in its routing table, and the summary route to null0. Only the summary, network 172.16.0.0/16, is advertised out the Serial 0/0/0 interface.

Example 2-26. Routing Table of Router C in Figure 2-22

```
RouterC#show ip route
<output omitted>
Gateway of last resort is not set
  172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
D    172.16.0.0/16 is a summary, 00:00:04, Null0
D    172.16.1.0/24 [90/156160] via 10.1.1.2, 00:00:04, FastEthernet0/0
D    172.16.2.0/24 [90/20640000] via 10.2.2.2, 00:00:04, Serial0/0/1
C    192.168.4.0/24 is directly connected, Serial0/0/0
  10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C    10.2.2.0/24 is directly connected, Serial0/0/1
C    10.1.1.0/24 is directly connected, FastEthernet0/0
D    10.0.0.0/8 is a summary, 00:00:05, Null0
```

RouterC#

For manual summarization, the summary is advertised only if a component of the summary route (a more specific entry that is represented in the summary) is present in the routing table.

Note

IP EIGRP summary routes are given an administrative distance value of 5. Standard EIGRP routes receive an administrative distance of 90, and external EIGRP routes receive an administrative distance of 170.

You will notice the EIGRP summary route with an administrative distance of 5 only on the local router that is performing the summarization (with the **ip summary-address eigrp** command), by using the **show ip route network** command, where *network* is the specified summarized route.

Configuring and Verifying EIGRP in an Enterprise WAN

This section provides insight into EIGRP deployment over various WAN technologies including physical Frame Relay, multipoint and point-to-point Frame-Relay subinterfaces, Multiprotocol Label Switching (MPLS) virtual private networks (VPNs), and Ethernet over Multiprotocol Label Switching (EoMPLS). Advanced configuration options for EIGRP load balancing and limiting EIGRP bandwidth utilization on WAN links are also explored.

EIGRP over Frame Relay and on a Physical Interface

This section reviews Frame Relay and describes how EIGRP can be deployed over Frame Relay physical interfaces.

Frame Relay Overview

Frame Relay is a switched WAN technology where virtual circuits (VCs) are created by a service provider (SP) through the network. Frame Relay allows multiple logical VCs to be multiplexed over a single physical interface. The VCs are typically PVCs that are identified by a data-link connection identifier (DLCI). DLCIs are locally significant between the local router and the Frame Relay switch to which the router is connected. Therefore, each end of the PVC may have a different DLCI. The SP's network takes care of sending the data through the PVC. To provide IP layer connectivity, a mapping between IP addresses and DLCIs must be defined, either dynamically or statically.

By default, a Frame Relay network is an NBMA network. In an NBMA environment all routers are on the same subnet, but broadcast (and multicast) packets cannot be sent just once as they are in a broadcast environment such as Ethernet.

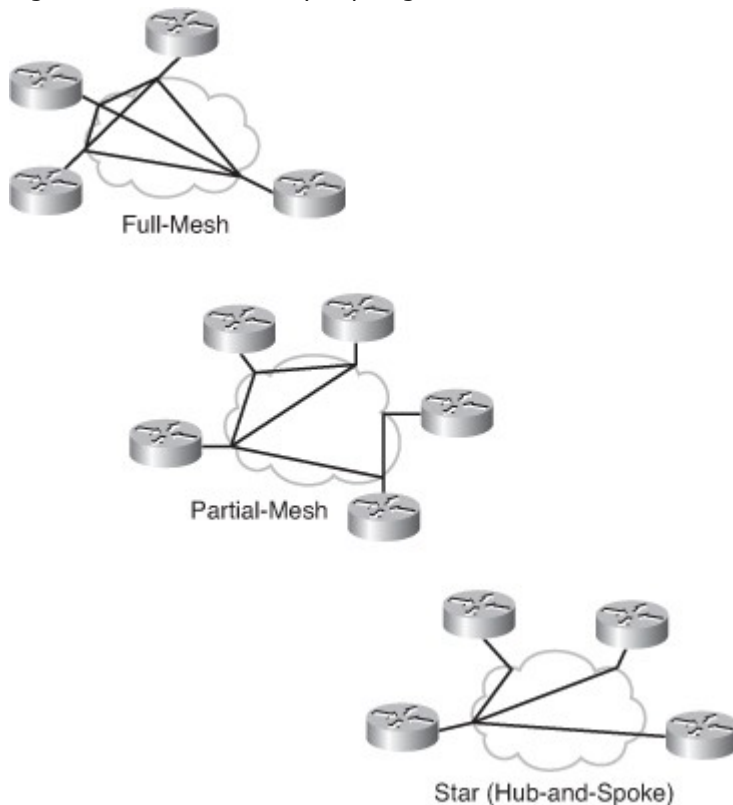
To emulate the LAN broadcast capability that is required by IP routing protocols (for example, to send EIGRP hello or update packets to all neighbors reachable over an IP subnet), the Cisco IOS implements pseudo-broadcasting, in which the router creates a copy of the broadcast or multicast packet for each neighbor reachable through the WAN media, and sends it over the appropriate PVC for that neighbor.

In environments where a router has a large number of neighbors reachable through a single WAN interface, pseudo-broadcasting has to be tightly controlled because it could increase the CPU resources and WAN bandwidth used. Pseudo-broadcasting can be controlled with the **broadcast** option on static maps in a Frame Relay configuration. However, pseudo-broadcasting cannot be controlled for neighbors reachable through dynamic maps created via Frame Relay Inverse Address Resolution Protocol (ARP). Dynamic maps always allow pseudo-broadcasting.

Frame Relay neighbor loss is detected only after the routing protocol hold time expires or if the interface goes down. An interface is considered to be up as long as at least one PVC is active.

Frame Relay allows remote sites to be interconnected using full-mesh, partial-mesh, and hub-and-spoke (also called star) topologies, as shown in Figure 2-23.

Figure 2-23. Frame Relay Topologies.

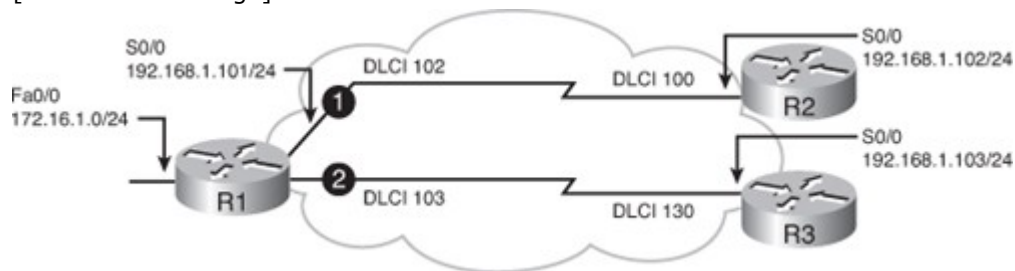


EIGRP on a Physical Frame Relay Interface with Dynamic Mapping

To deploy EIGRP over a physical interface using Inverse ARP dynamic mapping is easy because it is the default. Figure 2-24 illustrates an example network. Example 2-27 is the configuration of Router R1 in the figure. The physical interface Serial 0/0 is configured for Frame Relay encapsulation and an IP address is assigned. Inverse ARP is on by default and will automatically map the IP addresses of the devices at the other ends of the PVCs to the local DLCI number. EIGRP is enabled using autonomous system number 110, and the proper interfaces and networks are included in EIGRP using the network commands under the EIGRP routing process.

Figure 2-24. EIGRP on a Physical Frame Relay Interface.

[View full size image]



Example 2-27. Configuration of Router R1 in Figure 2-24 with Dynamic Mapping

```
interface Serial0/0
 encapsulation frame-relay
 ip address 192.168.1.101 255.255.255.0
 !
router eigrp 110
 network 172.16.1.0 0.0.0.255
 network 192.168.1.0
```

Split horizon is disabled by default on Frame Relay physical interfaces. Therefore, routes from Router R2 can be sent to Router R3, and vice versa. Note that Inverse ARP does not provide dynamic mapping for the communication between Routers R2 and R3 because they are not connected with a PVC. You must configure this mapping manually.

The sample outputs of the show ip eigrp neighbors command in Example 2-28 show the neighbors of Routers R1 and R3. Router R1 forms the adjacency with Router R2 and R3 over the Serial 0/0 physical interface. Likewise, Router R3 (and R2) forms an adjacency with Router R1. In this example, no EIGRP relationship exists between Routers R2 and R3.

Example 2-28. Sample Output from Routers R1 and R3 in Figure 2-24

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime  SRTT  RTO  Q  Seq
   (sec)        (ms)      Cnt  Num
0  192.168.1.102 Se0/0      10    00:07:22  10   2280  0  5
1  192.168.1.103 Se0/0      10    00:09:34  10   2320  0  9

R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime  SRTT  RTO  Q  Seq
   (sec)        (ms)      Cnt  Num
0  192.168.1.101 Se0/0      10    00:11:45  10   1910  0  6
```

EIGRP on a Frame Relay Physical Interface with Static Mapping

To deploy EIGRP over the physical interface on Router R1 using static mapping, thus disabling the Inverse ARP, no changes are needed to the basic EIGRP configuration. Example 2-29 illustrates the configuration on Routers R1 and R3. EIGRP is enabled using autonomous system number 110, and the proper interfaces and networks are included in EIGRP using the network commands under the EIGRP routing process. Manual IP-to-DLCI mapping commands on the serial 0/0 interface are necessary on all three routers. (Router R2's configuration is similar to R3's.)

Example 2-29. Configuration of Router R1 and R3 in Figure 2-24 with Static Mapping

```
Code View: Scroll / Show All
R1#show run
<output omitted>
interface Serial0/0
 encapsulation frame-relay
 ip address 192.168.1.101 255.255.255.0
 frame-relay map ip 192.168.1.101 101
 frame-relay map ip 192.168.1.102 102 broadcast
 frame-relay map ip 192.168.1.103 103 broadcast
!
router eigrp 110
 network 172.16.1.0 0.0.0.255
 network 192.168.1.0
<output omitted>

R3#show run
<output omitted>
interface Serial0/0
 encapsulation frame-relay
 ip address 192.168.1.103 255.255.255.0
```



```

frame-relay map ip 192.168.1.101 130 broadcast
!
router eigrp 110
 network 192.168.1.0
<output omitted>

```

The frame-relay map protocol protocol-address dlci [broadcast] [ietf | cisco] [payload-compress {packet-by-packet | frf9 stac}] interface configuration command statically maps the remote router's network layer address to the local DLCI. Table 2-6 details the parameters for this command.

Table 2-6. *frame-relay map* Command Parameters

Parameter	Description
protocol	Defines the supported protocol, bridging, or logical link control.
protocol-address	Defines the network layer address of the destination router interface
dlci	Defines the local DLCI that is used to connect to the remote protocol address
broadcast	(Optional) Allows broadcasts and multicasts over the VC, permitting the use of dynamic routing protocols over the VC
ietf cisco	Enables IETF or Cisco encapsulations
payload-compress	(Optional) Enables payload compression
packet-by-packet	(Optional) Enables packet-by-packet payload compression, using the Stacker method, a Cisco proprietary compression method
frf9 stac	(Optional) Enables FRF.9 compression using the Stacker method

Because split horizon is disabled by default on Frame Relay physical interfaces, routes from Router R2 can be sent to Router R3, and vice versa.

Note

Router R1's configuration includes a Frame Relay map to its own IP address on the Frame Relay interface so that the Serial 0/0 local IP address can be pinged from Router R1 itself.

Example 2-30 displays the adjacency formed between Router R1 and Routers R2 and R3 over the Serial 0/0 physical interface. The adjacencies formed on R1 using static mapping are the same as those formed using dynamic mapping. Routers R2 and R3 also form an adjacency with Router R1. Routers R2 and R3 can also form an EIGRP adjacency to each other if the IP-to-DLCI mapping for that connectivity is provided. Example 2-30 shows that Router R3 has only one neighbor, Router R1, indicating that this mapping was not provided on R3.

Example 2-30. Output on Routers R1 and R3 in Figure 2-24 with Static Mapping

```

R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime   SRTT  RTO  Q  Seq
                               (sec)   (ms)    Cnt  Num
0  192.168.1.102 Se0/0     10    00:06:20 10   2280 0  5
1  192.168.1.103 Se0/0     10    00:08:31 10   2320 0  9

R3#show ip eigrp neighbors

```

IP-EIGRP neighbors for process 110

H	Address	Interface	Hold (sec)	Uptime (ms)	SRTT Cnt	RTO Num	Q	Seq
0	192.168.1.101	Se0/0	10	00:10:44	10	1910	0	6

EIGRP over Frame Relay Multipoint Subinterfaces

This section describes how you can deploy EIGRP using Frame Relay multipoint subinterfaces and explores the use of EIGRP unicast neighbors.

Frame Relay Multipoint Subinterfaces

You can create one or several multipoint subinterfaces over a single Frame Relay physical interface. These multipoint subinterfaces are logical interfaces emulating a multiaccess network. They act like an NBMA physical interface, and therefore use a single subnet, preserving the IP address space.

Frame Relay multipoint is applicable to partial-mesh and full-mesh topologies. Partial-mesh Frame Relay networks must deal with split-horizon issues, which prevent routing updates from being retransmitted on the same interface on which they were received.

EIGRP neighbor loss detection is particularly slow on multipoint subinterfaces configured over low-speed WAN links because the default values of the EIGRP timers on these interfaces are 60 seconds for the hello timer and 180 seconds for the hold timer. In the worst case, neighbor loss detection can take up to 3 minutes. Also, on Frame Relay multipoint subinterfaces, all of the PVCs attached to the subinterface must be lost for the subinterface to be declared down.

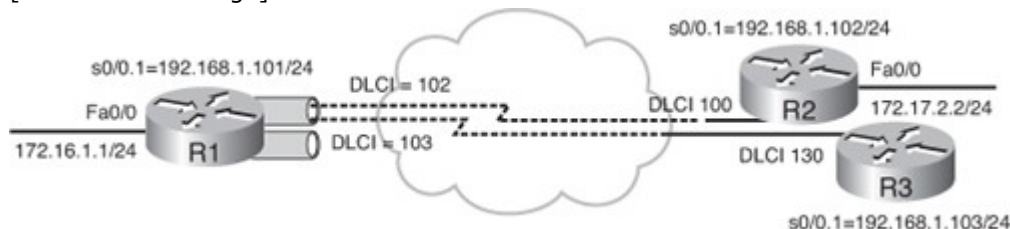
EIGRP over Multipoint Subinterfaces

Multipoint subinterfaces are configured with the **interface serial number.subinterface-number multipoint** command. For Frame Relay, the IP address-to-DLCI mapping on multipoint subinterfaces is done by either specifying the local DLCI value (using the **frame-relay interface-dlci dlci** command) and relying on Inverse ARP, or using manual IP address-to-DLCI mapping.

Figure 2-25 illustrates an example network. Example 2-31 illustrates the configuration of Routers R1 and R3 in the figure. The physical interface Serial 0/0 on each router is configured for Frame Relay encapsulation. The physical interface does not have an IP address assigned to it. (Note that in this example R3 only has one Frame Relay DLCI, so does not really need to be multipoint, or even a subinterface.)

Figure 2-25. EIGRP on a Multipoint Frame Relay Subinterface.

[View full size image]



Example 2-31. Configuration of Router R1 and R3 in Figure 2-25

Code View: Scroll / Show All

R1#**show run**

<output omitted>

interface Serial0/0

no ip address

encapsulation frame-relay

!

interface Serial0/0.1 multipoint

ip address 192.168.1.101 255.255.255.0

no ip split-horizon eigrp 110

frame-relay map ip 192.168.1.101 101

frame-relay map ip 192.168.1.102 102 broadcast

frame-relay map ip 192.168.1.103 103 broadcast

```

!
router eigrp 110
 network 172.16.1.0 0.0.0.255
 network 192.168.1.0
<output omitted>

R3#show run
<output omitted>
interface Serial0/0
 no ip address
 encapsulation frame-relay
!
interface Serial0/0.1 multipoint
 ip address 192.168.1.103 255.255.255.0
 frame-relay map ip 192.168.1.101 130 broadcast
!
router eigrp 110
 network 192.168.1.0
<output omitted>

```

A multipoint subinterface Serial 0/0.1 is created and an IP address is assigned to it.

Split horizon is enabled by default on Frame Relay multipoint subinterfaces. In this example, Routers R2 and R3 need to provide connectivity between their connected networks, so EIGRP split horizon is disabled on the multipoint subinterface of Router R1 with the **no ip split-horizon eigrp as-number** command.

Manual IP address-to-DLCI mapping is also configured, using the **frame-relay map** commands with the **broadcast** keyword.

The EIGRP configuration is not changed from the basic deployment. In Example 2-31, EIGRP is enabled using autonomous system number 110 and the proper interfaces, and networks are included in EIGRP using the network commands under the EIGRP routing process.

Note that the Router R1 configuration includes **frame-relay map** command to its own IP address on the multipoint serial subinterface so that the Serial 0/0.1 local IP address can be pinged from Router R1 itself.

To verify the operation of the EIGRP routing protocol over Frame Relay multipoint subinterfaces, use the show ip eigrp neighbors command. Example 2-32 shows sample outputs for Routers R1 and R3. Router R1 forms an adjacency with Routers R2 and R3 over the serial0/0.1 multipoint subinterface. Routers R2 and R3 form the adjacency with Router R1. (R2 and R3 could also form an adjacency between each other if the IP address-to-DLCI mapping for that connectivity was provided.)

Example 2-32. Output on Routers R1 and R3 in Figure 2-25

```

R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime  SRTT  RTO  Q  Seq
      (sec)      (ms)      Cnt  Num
0  192.168.1.102 Se0/0.1    10    00:06:41  10   2280  0  5
1  192.168.1.103 Se0/0.1    10    00:08:52  10   2320  0  9

R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime  SRTT  RTO  Q  Seq
      (sec)      (ms)      Cnt  Num
0  192.168.1.101 Se0/0.1    10    00:10:37  10   1910  0  6

```

EIGRP Unicast Neighbors

The **neighbor** {*ip-address* | *ipv6-address*} *interface-type interface-number* router configuration command is used to define a neighboring router with which to exchange EIGRP routing information. Instead of using multicast packets, EIGRP exchanges routing information with the specified neighbor using unicast packets. The router does not process any EIGRP multicast packets coming inbound on that interface and it stops sending EIGRP multicast packets on that interface. Multiple **neighbor** statements can be used to establish peering sessions with multiple specific EIGRP neighbors. The interface through which EIGRP will exchange routing updates must be specified in the **neighbor** statement. The interfaces through which two EIGRP neighbors exchange routing updates must be configured with IP addresses from the same network.

Note

EIGRP neighbor adjacencies cannot be established or maintained over an interface that is configured as passive.

In Example 2-33, Router R1 (from Figure 2-25) is configured with a neighbor command for Router R2. Router R1 will therefore not accept multicast packets on Serial 0/0.1 anymore. To establish an adjacency with Router R1, Router R2 must also be configured with a neighbor command, for Router R1. R2's configuration is provided in Example 2-34. The neighbor command enables Router R2 to use unicast packets, which will be accepted by Router R1. In this scenario, Router R3 is not configured with a neighbor command for Router R1, nor is Router R1 configured with a neighbor command for Router R3. Therefore, Router R1 and R3 will not form an adjacency.

Example 2-33. Configuration of Router R1 in Figure 2-25 for Unicast Neighbor

```
R1#show run
<output omitted>
interface Serial0/0
  no ip address
  encapsulation frame-relay
!
interface Serial0/0.1 multipoint
  ip address 192.168.1.101 255.255.255.0
  frame-relay map ip 192.168.1.102 102 broadcast
  frame-relay map ip 192.168.1.103 103 broadcast
!
router eigrp 110
  network 172.16.1.0 0.0.0.255
  network 192.168.1.0
  neighbor 192.168.1.102 serial0/0.1
```

Example 2-34. Configuration of Router R2 in Figure 2-25 for Unicast Neighbor

```
R2#show run
<output omitted>
interface Serial0/0
  no ip address
  encapsulation frame-relay
!
interface Serial0/0.1 multipoint
  ip address 192.168.1.102 255.255.255.0
  frame-relay map ip 192.168.1.101 100 broadcast
!
router eigrp 110
  network 172.17.2.0 0.0.0.255
```

```
network 192.168.1.0
neighbor 192.168.1.101 serial0/0.1
```

To verify the EIGRP unicast neighbor configuration, use the `show ip eigrp neighbors` command, as shown in Example 2-35. Routers R1 and R2 have formed a neighbor relationship. The sample output does not show that the `neighbor` command was used on both routers, but it does indicate that a neighbor relationship is established, proof that the configuration was successfully completed.

Because Router R3 is not using the **neighbor** command, it tries to communicate with multicast packets on its Serial 0/0/.1. However, neighborship is not established because neither Router R1 nor Router R2 is accepting multicast packets.

Example 2-35. Output on Routers R1 and R2 in Figure 2-25 for Unicast Neighbor

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold  Uptime   SRTT  RTO  Q  Seq
                               (sec)   (ms)    Cnt  Num
0  192.168.1.102    Se0/0/1   10    00:07:22  10    2280 0  5

R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold  Uptime   SRTT  RTO  Q  Seq
                               (sec)   (ms)    Cnt  Num
0  192.168.1.101    Se0/0/1   10    00:17:02  10    1380 0  5
```

EIGRP over Frame Relay Point-to-Point Subinterfaces

This section describes how you can deploy EIGRP using Frame Relay point-to-point subinterfaces.

Frame Relay Point-to-Point Subinterfaces

One or several point-to-point subinterfaces can be created over a single Frame Relay physical interface. These point-to-point subinterfaces are logical interfaces emulating a leased line network and provide a routing equivalent to point-to-point physical interfaces. As with physical point-to-point interfaces, each interface requires its own subnet. Frame Relay point-to-point is applicable to hub and spoke topologies.

EIGRP neighbor loss detection is quite fast on point-to-point subinterfaces because the default values of the EIGRP hello timer and the EIGRP hold timer are identical to the values used on point-to-point physical links (5 seconds for the hello timer and 15 seconds for the hold timer). In the worst case, the neighbor loss is detected within 15 seconds. Another reason that neighbor loss is fast is that a Frame Relay point-to-point subinterface is declared down if the DLCI attached to the interface is lost—neighbor loss detection is immediate. Recall that for multipoint subinterfaces, all the PVCs attached to it must be lost for the interface to be declared down.

Note

Neighbor loss detection because of DLCI loss only works if the Frame Relay network supports end-to-end integrated Local Management Interface (LMI) signaling. On some Frame Relay networks, one end of the connection might fail (for example, because of a router failure), but the DLCI would still be declared operational at the other end of the connection.

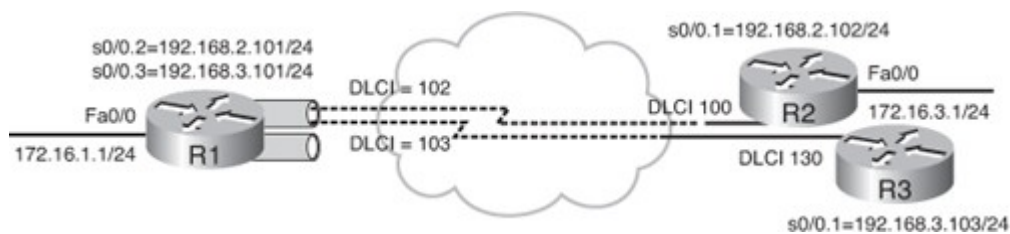
EIGRP on Frame Relay Point-to-Point Subinterfaces

Point-to-point subinterfaces are created with the **interface serial number.subinterface-number point-to-point** command. For Frame Relay, IP address-to-DLCI mapping on point-to-point subinterfaces is done by specifying the local DLCI value, using the **frame-relay interface-dlci dlci** command.

Figure 2-26 illustrates an example network; Example 2-36 is the configuration of Router R1, and Example 2-37 is the configuration of Router R3. The physical interface Serial 0/0 is configured for Frame Relay encapsulation. The physical interface does not have an IP address assigned to it.

Figure 2-26. EIGRP over a Point-to-Point Frame Relay Subinterface.

[View full size image]



Example 2-36. Configuration of Router R1 in Figure 2-26

```
R1#show run
<output omitted>
interface Serial0/0
  no ip address
  encapsulation frame-relay
  !
interface Serial0/0.2 point-to-point
  ip address 192.168.2.101 255.255.255.0
  frame-relay interface-dlci 102
  !
interface Serial0/0.3 point-to-point
  ip address 192.168.3.101 255.255.255.0
  frame-relay interface-dlci 103
  !
router eigrp 110
  network 172.16.1.0 0.0.0.255
  network 192.168.2.0
  network 192.168.3.0
```

Example 2-37. Configuration of Router R3 in Figure 2-26

```
R3#show run
<output omitted>
interface Serial0/0
  no ip address
  encapsulation frame-relay
  !
interface Serial0/0.1 point-to-point
  ip address 192.168.3.103 255.255.255.0
  frame-relay interface-dlci 130
  !
router eigrp 110
  network 172.16.3.0 0.0.0.255
  network 192.168.3.0
```

On Router R1, two point-to-point subinterfaces, Serial 0/0.2 and Serial 0/0.3, are created and IP addresses are assigned to them. The IP address-to-DLCI mapping is provided for each subinterface. The EIGRP configuration is not changed from the basic deployment. EIGRP is enabled using autonomous system number 110 and the proper interfaces, and networks are included in EIGRP using the **network** commands under the EIGRP routing process.

Router R3 is configured similar to Router R1, with one point-to-point subinterface. The show ip eigrp neighbors command can be used to verify the operation of the EIGRP routing protocol over the Frame Relay point-to-point subinterface. Example 2-38 shows sample outputs for Routers R1 and R3. Router R1 forms the

adjacency with Router R2 over its Serial 0/0.2 point-to-point interface and with Router R3 over its Serial 0/0.3 point-to-point subinterface. Likewise, Routers R2 and R3 form the adjacency with Router R1 over the Serial 0/0.1 point-to-point subinterface. Router R3 has one neighbor, Router R1, over its Serial 0/0.1 point-to-point subinterface.

Example 2-38. Output on Routers R1 and R3 in Figure 2-26

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold  Uptime  SRTT  RTO  Q  Seq
   (sec)            (ms)      Cnt  Num
0  192.168.2.102    Se0/0.2   10    00:08:04  10    2280  0  5
1  192.168.3.103    Se0/0.3   10    00:10:12  10    2320  0  9

R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold  Uptime  SRTT  RTO  Q  Seq
   (sec)            (ms)      Cnt  Num
0  192.168.3.101    Se0/0.1   10    00:13:25  10    1910  0  6
```

EIGRP over MPLS

This section provides an overview of MPLS and Layer 2 and Layer 3 MPLS VPN technologies and explains how you can deploy EIGRP in such environment.

MPLS

MPLS is an Internet Engineering Task Force (IETF) standard architecture that combines the advantages of Layer 3 routing with the benefits of Layer 2 switching.

With MPLS, short fixed-length labels are assigned to each packet at the edge of the network. Rather than examining the IP packet header information, MPLS nodes use this label to determine how to process the data.

This process results in a more scalable and flexible WAN solution. The MPLS standards evolved from the efforts of many companies, including Cisco's tag-switching technology.

MPLS enables scalable VPNs, end-to-end quality of service (QoS), and other IP services that allow efficient utilization of existing networks with simpler configuration, management, and quicker fault correction.

MPLS Operation

MPLS is a connection-oriented technology whose operation is based on a label attached to each packet as it enters the MPLS network. A label identifies a *flow* of packets (for example, voice traffic between two nodes), also called a *forwarding equivalence class* (FEC). An FEC is a grouping of packets. Packets belonging to the same FEC receive the same treatment in the network. The FEC can be determined by various parameters, including source or destination IP address or port numbers, IP protocol, IP precedence, or Layer 2 circuit identifier. Therefore, the FEC can define the flow's QoS requirements. In addition, appropriate queuing and discard policies can be applied for FECs.

The MPLS network nodes, called *label-switched routers* (LSRs), use the label to determine the next hop for the packet. The LSRs do not need to examine the packet's IP header; rather, they forward it based on the label.

After a path has been established, packets destined to the same endpoint with the same requirements can be forwarded based on these labels without a routing decision at every hop. Labels usually correspond to Layer 3 destination prefixes, which makes MPLS equivalent to destination-based routing.

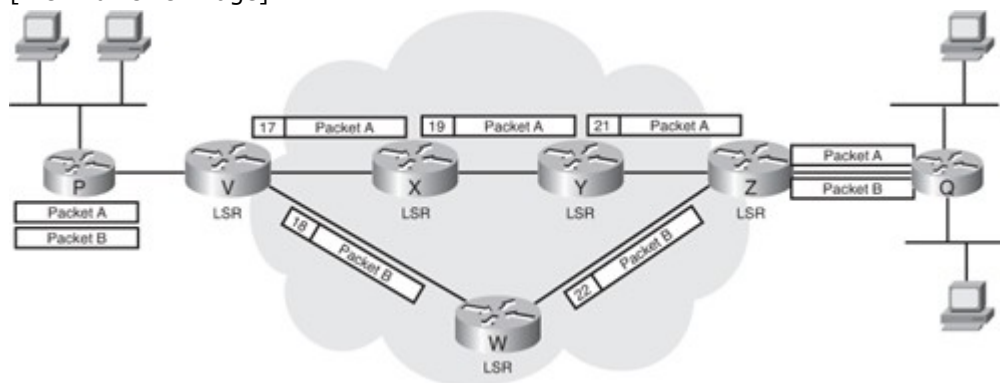
A *label-switched path* (LSP) must be defined for each FEC before packets can be sent. It is important to note that labels are locally significant to each MPLS node only. Therefore, the nodes must communicate what label to use for each FEC. One of two protocols is used for this communication: the Label Distribution Protocol or an enhanced version of the Resource Reservation Protocol. An interior routing protocol, such as OSPF or EIGRP is also used within the MPLS network to exchange routing information.

A unique feature of MPLS is its capability to perform label stacking, in which multiple labels can be carried in a packet. The top label, which is the last one in, is always processed first. Label stacking enables multiple LSPs to be aggregated, thereby creating tunnels through multiple levels of an MPLS network.

An MPLS label is a 32-bit field placed between a packet's data link layer header and its IP header. Figure 2-27 illustrates the flow of two packets through an MPLS network.

Figure 2-27. Labels Are Used to Assign a Path for a Packet Flow Through an MPLS Network.

[View full size image]



Note

The links shown in Figure 2-27 are meant to be generic. Therefore, they do not represent any particular type of interface.

In Figure 2-27, each of the MPLS nodes has previously communicated the labels it uses for each of the defined FECs to its neighboring nodes. Packet A and Packet B represent different flows. For example, Packet A might be from an FTP session, whereas Packet B is from a voice conversation. Without MPLS, these packets would take the same route through the network.

In Figure 2-27, Router V is the ingress edge LSR for Packets A and B, the point at which the packets enter the network. Router V examines each packet and determines the appropriate FEC. Packet A is assigned label 17 and is sent to Router X. Packet B is assigned label 18 and is sent to Router W. As each LSR receives a labeled packet, it removes the label, locates the label in its table, applies the appropriate outgoing label, and forwards the packet to the next LSR in the LSP. When the packets reach Router Z (the egress edge LSR, or the point at which the packets leave the MPLS network), Router Z removes the label and forwards the packets appropriately, based on its IP routing table.

It is important to note that packets sent between the same endpoints might belong to different MPLS FECs, and therefore might flow through different paths in the network.

Service Provider Offerings

Many ISPs offer Layer 2 transport services to their customers to interconnect networks at various offices via customer equipment (CE) routers. ISPs offer these services over a circuit-based infrastructure to build Layer 2 VPNs. Initial VPNs were built using leased lines with PPP and HDLC encapsulations. Later, service providers offered Layer 2 VPNs based on point-to-point data link layer connectivity, using ATM or Frame Relay virtual circuits. Customers built their own Layer 3 networks on top of these Layer 2 networks to accommodate IP traffic. Thus, separate networks existed for Layer 2 and Layer 3 traffic, which were difficult and costly to maintain.

MPLS VPNs were introduced to provide a unified network for Layer 3 VPN services. For customers still wanting Layer 2 connections, ISPs could deploy Ethernet VLAN extensions across a metropolitan area or ATM services. Any Transport over MPLS (AToM) was introduced to facilitate this Layer 2 connectivity across an MPLS backbone.

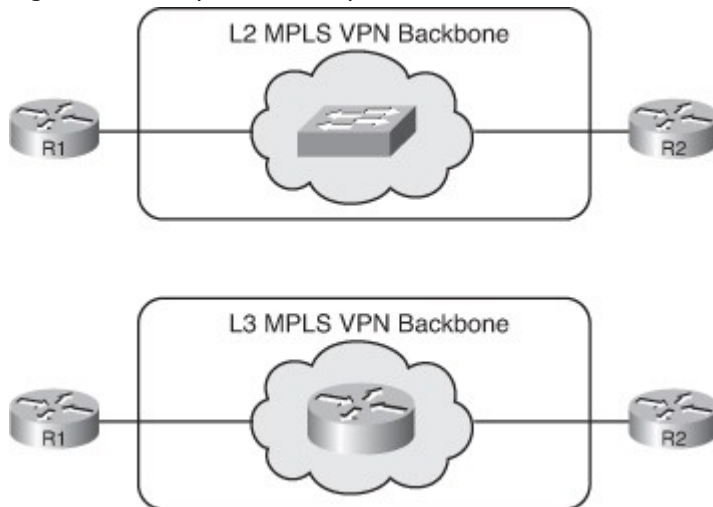
AToM enables sending Layer 2 frames across an MPLS backbone. It unifies Layer 2 and Layer 3 offerings over a common MPLS infrastructure. In AToM, VCs represent Layer 2 links, and MPLS labels identify VCs.

AToM allows more offerings from ISPs that currently offer Layer 2 connectivity to customers with traditional offerings such as ATM, Frame Relay, and serial PPP services, and those specializing in Ethernet connectivity in metropolitan areas. These Layer 2 VPN services appeal to the ISP's enterprise customers who may already run their own networks and desire only point-to-point connectivity between sites.

Layer 2 and Layer 3 MPLS VPN Solutions

Figure 2-28 illustrates the differences between a Layer 2 MPLS VPN and a Layer 3 MPLS VPN backbone solution. The customer routers (R1 and R2 in the figure) are connected across an MPLS VPN backbone.

Figure 2-28. Layer 2 and Layer 3 MPLS VPN Solutions.



The Layer 2 MPLS VPN provides a Layer 2 service across the backbone, where Routers R1 and R2 are connected together on the same IP subnet. Figure 2-28 represents connectivity through the backbone as a Layer 2 switch.

The Layer 3 MPLS VPN provides a Layer 3 service across the backbone, where Routers R1 and R2 are connected to ISP edge routers. On each side, a separate IP subnet is used. Figure 2-28 represents connectivity through the backbone as a router.

The following sections describe Layer 3 and Layer 2 MPLS VPNs.

Layer 3 MPLS VPNs

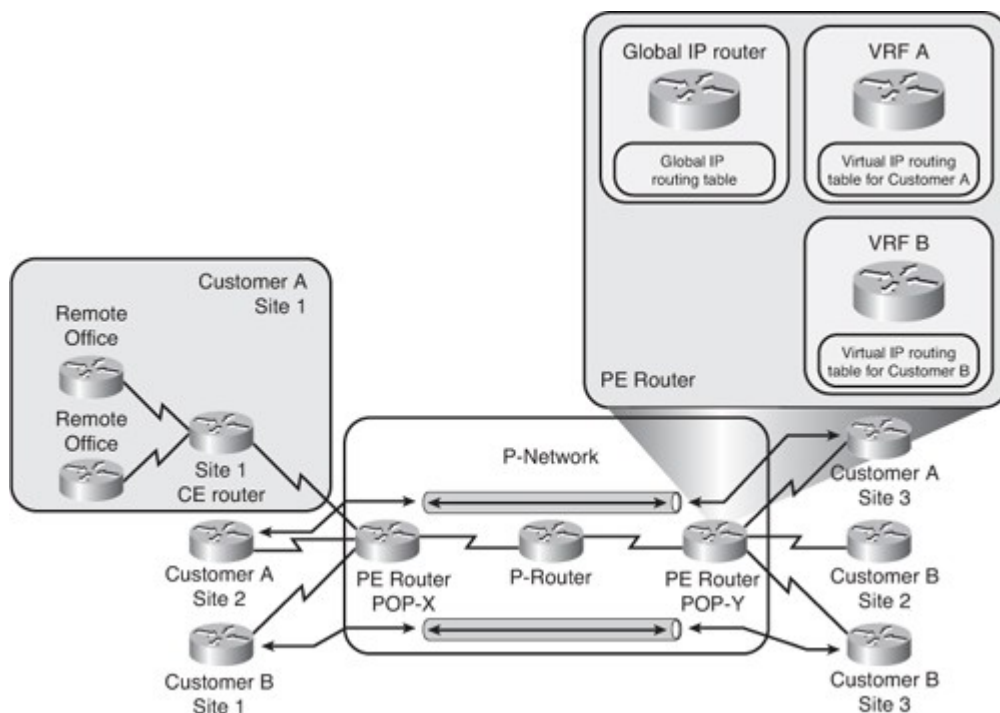
This section describes Layer 3 MPLS VPNs and how EIGRP is deployed over them.

Layer 3 MPLS VPN Overview

As shown in Figure 2-29, in MPLS VPN terminology, the network is divided into the customer-controlled part (the C-network) and the provider-controlled part (the P-network). Contiguous portions of C-network are called sites and are linked to the P-network via customer edge routers (CE routers). The CE routers are connected to the provider edge (PE) routers, which serve as the edge devices of the Provider network. The core devices in the Provider network (the P-routers) provide the transport across the provider backbone and do not carry customer routes. The service provider connects multiple customers over a common MPLS backbone using MPLS VPNs.

Figure 2-29. Layer 3 MPLS VPN.

[View full size image]



The architecture of a PE router in an MPLS VPN is similar to the architecture of a point of presence (POP) in a dedicated PE router peer-to-peer model; however, in an MPLS VPN, the whole architecture is condensed into one physical device. In an MPLS VPN, each customer is assigned an independent routing table—the virtual routing and forwarding (VRF) table in the PE router—that corresponds to the dedicated PE router in a traditional peer-to-peer model. Routing across the provider backbone is performed by a separate routing process that uses a global IP routing table. This routing process corresponds to the P-routers within the in traditional peer-to-peer model.

The MPLS VPN backbone in Figure 2-29 is a Layer 3 backbone. The CE routers treat the PE routers the same as they treat other customer routers in the path between sites. PE routers maintain separate routing tables for each customer.

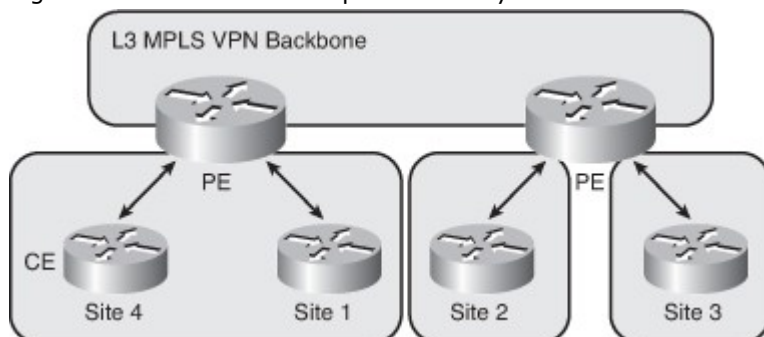
The MPLS VPN architecture provides ISPs with a peer-to-peer VPN architecture that combines the best features of an overlay VPN (including support for overlapping customer address spaces) with the best features of peer-to-peer VPNs, including the following:

- PE routers participate in customer routing, providing optimum routing between customer sites.
- PE routers carry a separate set of routes for each customer, isolating each customer from other customers.

Customer Perspective of Layer 3 MPLS VPNs

As illustrated in Figure 2-30, the MPLS VPN backbone looks like a standard corporate backbone to the CE routers. The CE routers run standard IP routing software and exchange routing updates with the PE routers, which appear to them as normal routers in the customer's network. Therefore, the customer and the SP must agree on EIGRP parameters.

Figure 2-30. Customer Perspective of Layer 3 MPLS VPNs.



The internal topology of the MPLS backbone is transparent to the customer. The internal P-routers are hidden from the customer's view and the CE routers are unaware of the MPLS VPN.

In a Layer 3 MPLS VPN, the following requirements must be met:

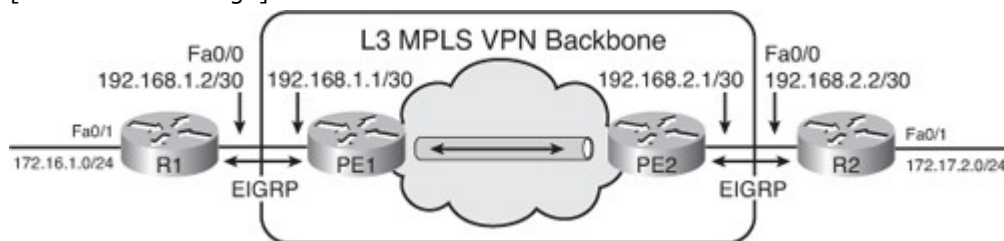
- The customer routers (the CE routers) should not be MPLS VPN aware. They should run standard IP routing software.
- The provider core routers (the P-routers) must not carry customer (VPN) routes, to make the MPLS VPN solution scalable.
- The PE routers must support MPLS VPN services and traditional IP services.

EIGRP over Layer 3 MPLS VPN

Figure 2-31 provides an example of EIGRP deployed over a Layer 3 MPLS VPN. The configurations for Routers R1 and R2 are shown in Example 2-39.

Figure 2-31. EIGRP over a Layer 3 MPLS VPN.

[View full size image]



Example 2-39. Configurations of Router R1 and R2 in Figure 2-31

```
R1#
interface FastEthernet0/0
ip address 192.168.1.2 255.255.255.252
!
router eigrp 110
network 172.16.1.0 0.0.0.255
network 192.168.1.0

R2#
interface FastEthernet0/0
ip address 192.168.2.2 255.255.255.252
!
router eigrp 110
network 172.17.2.0 0.0.0.255
network 192.168.2.0
```

Routers R1 and R2 are configured for EIGRP as if there were a corporate core network between them. The customer has to agree on the EIGRP parameters (such as the autonomous system number, authentication password, and so on) with the SP to ensure connectivity. The SP often governs these parameters.

The PE routers receive routing updates from the CE routers and install these updates in the appropriate VRF table. This part of the configuration and operation is the SP's responsibility.

Example 2-40 displays the resulting EIGRP neighbor tables on the R1 and R2 routers. Notice that Router R1 establishes an EIGRP neighbor relationship with the PE1 router, and Router R2 establishes an EIGRP neighbor relationship with the PE2 router. Routers R1 and R2 do not establish an EIGRP neighbor relationship with each other.

Example 2-40. Output on Router R1 and R2 in Figure 2-31

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H Address      Interface Hold Uptime  SRTT  RTO  Q Seq
```

```

                                (sec)      (ms)      Cnt Num
0  192.168.1.1    Fe0/0      10    00:07:22  10  2280  0  5

R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address          Interface  Hold  Uptime   SRTT   RTO  Q  Seq
                                (sec)      (ms)      Cnt Num
0  192.168.2.1    Fe0/0      10    00:17:02  10   1380  0  5

```

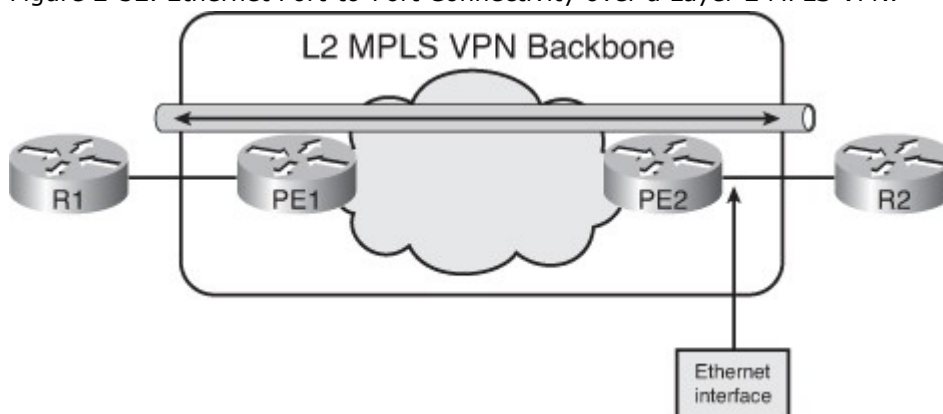
Layer 2 MPLS VPNs

This section describes Layer 2 MPLS VPNs and how EIGRP is deployed over them.

Ethernet Port-to-Port Connectivity over a Layer 2 MPLS VPN

With a Layer 2 MPLS VPN, an MPLS backbone provides a Layer 2 Ethernet port-to-port connection, such as between the two customer Routers R1 and R2 in Figure 2-32.

Figure 2-32. Ethernet Port-to-Port Connectivity over a Layer 2 MPLS VPN.



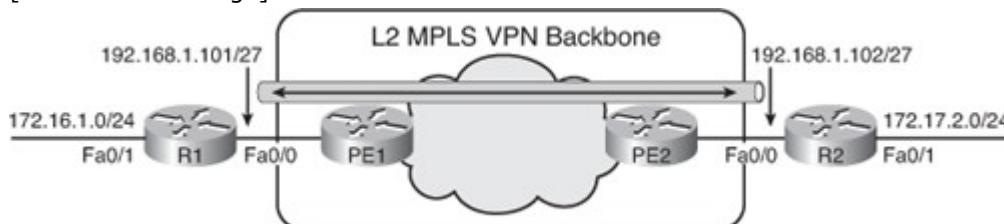
In Figure 2-32, Routers R1 and R2 are exchanging Ethernet frames. The PE1 router takes the Ethernet frame received from the directly connected Router R1, encapsulates it into an MPLS packet, and forwards it across the backbone to the PE2 router. The PE2 router decapsulates the MPLS packet and reproduces the Ethernet frame on its Ethernet link to Router R2. This process is a type of AToM called EoMPLS. (EoMPLS is also known as a type of Metro Ethernet service.)

AToM and EoMPLS do not include any MAC layer address learning and filtering. Therefore routers PE1 and PE2 do not filter any frames based on MAC addresses. AToM and EoMPLS also do not use the Spanning Tree Protocol (STP). Bridge protocol data units (BPDUs) are propagated transparently and not processed, so LAN loop detection must be performed by other devices or avoided by design. A service provider can use LAN switches in conjunction with AToM and EoMPLS to provide these features.

In the example in Figure 2-33, Routers R1 and R2 communicate with EIGRP over EoMPLS. The configuration for both routers is shown in Example 2-41.

Figure 2-33. EIGRP over EoMPLS.

[View full size image]



Example 2-41. Configurations of Router R1 and R2 in Figure 2-33

```

R1#

```

```

interface FastEthernet0/0
 ip address 192.168.1.101 255.255.255.224
 !
router eigrp 110
 network 172.16.1.0 0.0.0.255
 network 192.168.1.0

R2#
interface FastEthernet0/0
 ip address 192.168.1.102 255.255.255.224
 !
router eigrp 110
 network 172.17.2.0 0.0.0.255
 network 192.168.1.0

```

When deploying EIGRP over EoMPLS, there are no changes to the EIGRP configuration from the customer perspective. EIGRP needs to be enabled with the correct autonomous system number (the same on both Routers R1 and R2). The network commands must include all the interfaces that will run EIGRP, including the link toward the PE routers (routers PE1 and PE2) over which the Routers R1 and R2 will form their neighbor relationship. From the EIGRP perspective, the MPLS backbone and routers PE1 and PE2 are not visible. A neighbor relationship is established directly between Routers R1 and R2 over the MPLS backbone, and is displayed using the `show ip eigrp neighbors` command. Output from this command is shown in Example 2-42. Example 2-42. Output on Router R1 and R2 in Figure 2-33

```

R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime   SRTT  RTO  Q  Seq
      (sec)      (ms)      Cnt  Num
0  192.168.1.102 Fe0/0      10    00:07:22  10    2280  0  5

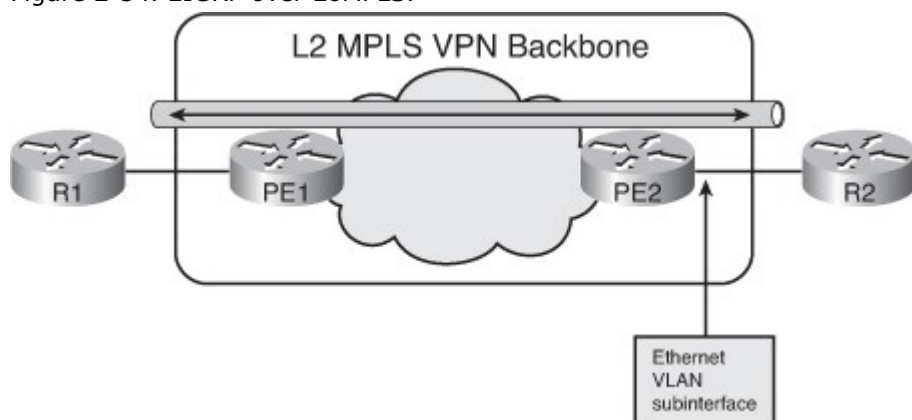
R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 110
H  Address      Interface  Hold  Uptime   SRTT  RTO  Q  Seq
      (sec)      (ms)      Cnt  Num
0  192.168.1.101 Fe0/0      10    00:17:02  10    1380  0  5

```

Ethernet VLAN Connectivity

Figure 2-34 illustrates an alternative scenario in which the two customer Routers R1 and R2 are connected to the MPLS edge routers PE1 and PE2 via VLAN subinterfaces. Different subinterfaces in the PE routers are used to connect to different VLANs. The PE1 subinterface to the VLAN where the Router R1 is connected is used for AToM forwarding. The Ethernet frame that arrives from Router R1 on the specific VLAN subinterface is encapsulated into MPLS and forwarded across the backbone to Router PE2. The PE2 router decapsulates the packet and reproduces the Ethernet frame on its outgoing VLAN subinterface to Router R2.

Figure 2-34. EIGRP over EoMPLS.



EIGRP Load Balancing

This section describes EIGRP equal-cost and unequal-cost load balancing.

EIGRP Equal-Cost Load Balancing

Equal-cost load balancing is a router's capability to distribute traffic over all the routers that have the same metric for the destination address. All IP routing protocols on Cisco routers can perform equal-cost load balancing.

Notice that the terminology is *equal-cost* even though the metric used in the routing protocol may not be called *cost* (as is the case for EIGRP).

Load balancing increases the utilization of network segments, thus increasing effective network bandwidth.

By default, the Cisco IOS balances between a maximum of four equal-cost paths for IP. Using the **maximum-paths** *maximum-path* router configuration command, you can request that up to 16 equally good routes be kept in the routing table. Set the *maximum-path* parameter to 1 to disable load balancing. When a packet is process-switched, load balancing over equal-cost paths occurs on a per-packet basis. When packets are fast-switched, load balancing over equal-cost paths is on a per-destination basis.

Note

If you are testing load balancing, do not ping to or from the routers with the fast-switching interfaces, because these locally router-generated packets are process-switched rather than fast-switched and might produce confusing results.

Note

Load balancing is performed only on traffic that passes *through* the router, not traffic generated by the router.

Figure 2-35 illustrates an example network with sample metrics indicated (the metrics are smaller than actual values for easier computation in the example). Example 2-43 shows the EIGRP configuration on Router R1. Table 2-7 displays the contents of Router R1's topology table for the 172.16.2.0/24 route.

Figure 2-35. EIGRP Equal-Cost Load Balancing.

[View full size image]

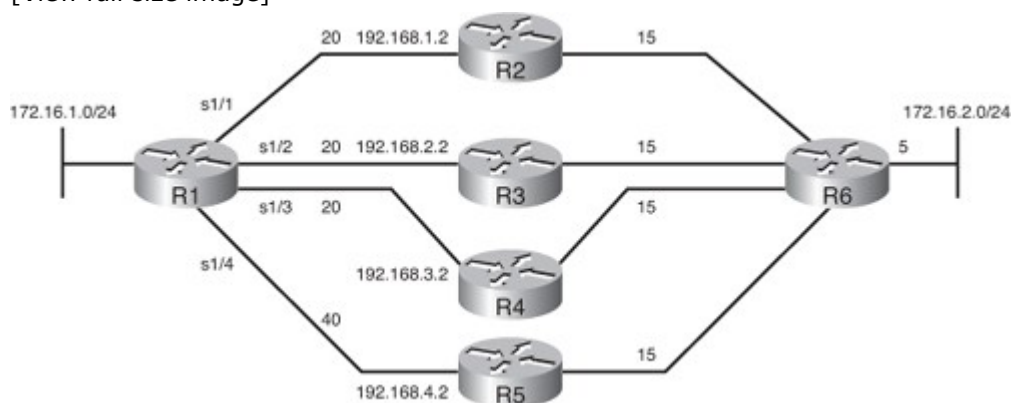


Table 2-7. Topology Table for 172.16.2.0/24 on Router R1 in Figure 2-35

Network	Neighbor	AD	FD
172.16.2.0/24	R2	20	40
	R3	20	40
	R4	20	40
	R5	20	60

Example 2-43. Configuration of Router R1 in Figure 2-35

```
router eigrp 110
 network 172.16.1.0 0.0.0.255
 network 192.168.1.0
 network 192.168.2.0
 network 192.168.3.0
 network 192.168.4.0
 maximum-paths 3
```

Router R1 is configured to support up to three equal-cost paths. Router R1 will keep the routes via R2, R3, and R4 in its routing table because the three paths have the same metric of 40 (they are equal cost), as shown in the FD column in Table 2-7. The path through router R5 is not used because the metric is bigger than 40 (it is 60). Even if this metric were the same as the others, only three of the four routes would be used because of the `maximum-paths 3` command.

EIGRP Unequal-Cost Load Balancing

EIGRP can also balance traffic across multiple routes that have different metrics—this is called *unequal-cost load balancing*.

The degree to which EIGRP performs load balancing is controlled by the **variance multiplier** router configuration command. The *multiplier* is a variance value, between 1 and 128, used for load balancing. The default is 1, which means equal-cost load balancing. The multiplier defines the range of metric values that are accepted for load balancing. Setting a variance value greater than 1 allows EIGRP to install multiple loop-free routes with unequal cost in the routing table. EIGRP will always install successors (the best routes) in the routing table. The variance allows feasible successors to also be installed in the routing table.

Only paths that are feasible can be used for load balancing, and the routing table only includes feasible paths. The two feasibility conditions are as follows:

- The route must be loop free. As noted earlier, this means that the best metric (the AD) learned from the next router must be less than the local best metric (the current FD). In other words, the next router in the path must be closer to the destination than the current router.
- The metric of the entire path (the FD of the alternative route) must be lower than the variance multiplied by the local best metric (the current FD). In other words, the metric for the entire alternate path must be within the variance.

If both of these conditions are met, the route is called feasible and can be added to the routing table.

The default value for the EIGRP variance is 1, which indicates equal-cost load balancing. In this case, only routes with the same metric as the successor are installed in the local routing table. The **variance** command does not limit the maximum number of paths. Instead, it defines the range of metric values that are accepted for load balancing by the EIGRP process. If the variance is set to 2, any EIGRP-learned route with a metric less than two times the successor metric will be installed in the local routing table. For example, if the **variance** command allows EIGRP to install nine paths and the **maximum-path** command sets the maximum paths value to 3, just the first three paths out of the available nine will be in the IP routing table.

Note

EIGRP itself does not load-share between multiple routes. It only installs the routes in the local routing table. The local routing table enables the router's switching hardware or software to load share between the multiple paths.

To control how traffic is distributed among routes when multiple routes exist for the same destination network and they have different metrics, use the **traffic-share [balanced | min across-interfaces]** router configuration command. With the keyword **balanced** (the default behavior), the router distributes traffic proportionately to the ratios of the metrics associated with the different routes. With the **min across-interfaces** option, the router uses only routes that have minimum costs. (In other words, all routes that are feasible and within the variance are kept in the routing table, but only those with the minimum cost are used.) This latter option allows feasible backup routes to always be in the routing table, but only be used if the primary route becomes unavailable so that it is no longer in the routing table.

Figure 2-36 illustrates the same network as in Figure 2-35, but with modified metrics. Example 2-44 shows the command added to the R1 router, setting the variance to 2. Table 2-8 displays the new topology table on the R1 router for the 172.16.2.0/24 route.

Figure 2-36. EIGRP Unequal-Cost Load Balancing.

[View full size image]

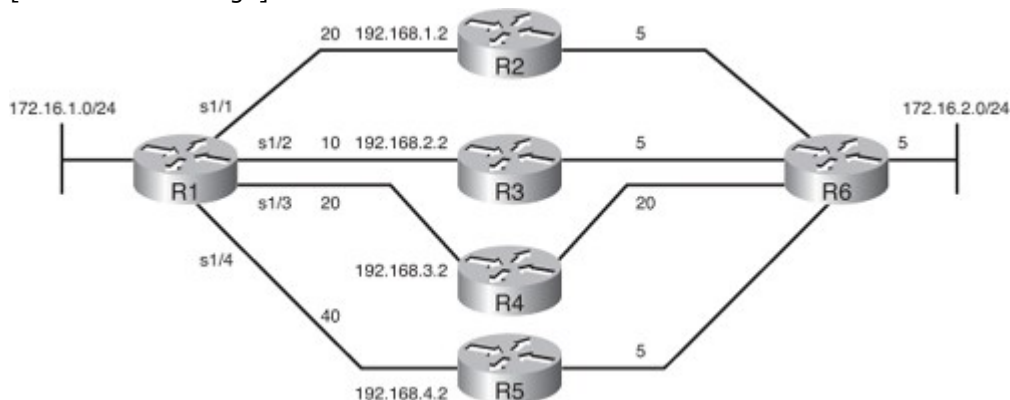


Table 2-8. New Topology Table for 172.16.2.0/24 on Router R1 in Figure 2-36

Network	Neighbor	AD	FD
172.16.2.0/24	R2	10	30
	R3	10	20
	R4	25	45
	R5	10	50

Example 2-44. Additional EIGRP Configuration of Router R1 in Figure 2-36

```
router eigrp 110
 variance 2
```

Router R1 uses Router R3 as the successor because its FD is lowest (20). With the **variance 2** command applied to Router R1, the path through Router R2 meets the criteria for load balancing. In this case, the FD through Router R2 (30) is less than twice the FD through the successor Router R3 ($2 * 20 = 40$).

Router R4 is not considered for load balancing because the FD through Router R4 (45) is greater than twice the FD for the successor (Router R3) ($2 * 20 = 40$). In this example, however, Router R4 would never be an FS no matter what the variance is, because its AD of 25 is greater than Router R1's FD of 20. Therefore, to avoid a potential routing loop, Router R4 is not considered closer to the destination than Router R1 and cannot be an FS.

Router R5 is not considered for load balancing with this variance because the FD through router R5 (50) is more than twice the FD for the successor through Router R3 ($2 * 20 = 40$). In this example, however, router R5 would be an FS because router R5's AD of 10 is lower than Router R3's FD of 20.

The load in Figure 2-36 is balanced proportional to the bandwidth. The FD of the route via Router R2 is 30, and the FD of the route via Router R3 is 20. The ratio of traffic between the two paths (via R2 : via R3) is therefore $3/5 : 2/5$.

In another example of unequal load balancing, four paths to a destination have the following metrics:

- Path 1: 1100
- Path 2: 1100
- Path 3: 2000
- Path 4: 4000

By default, the router routes to the destination using both Paths 1 and 2 because they have the same metric. Assuming no potential routing loops exist, the **variance 2** command would load balance over Paths 1, 2, and 3, because $1100 * 2 = 2200$, which is greater than the metric through Path 3. Similarly, the **variance 4** command would also include Path 4.

EIGRP Bandwidth Use Across WAN Links

As discussed earlier, EIGRP operates efficiently in WAN environments and is scalable on both point-to-point links and NBMA multipoint and point-to-point links. Because of the inherent differences in links' operational characteristics, default configuration of WAN connections might not be optimal. A solid understanding of EIGRP operation coupled with knowledge of link speeds can yield an efficient, reliable, scalable router configuration.

This section first introduces how EIGRP uses the bandwidth on interfaces, and then provides examples of EIGRP on various WANs.

EIGRP Link Utilization

By default, EIGRP uses up to 50 percent of the bandwidth declared on an interface or subinterface. EIGRP uses the bandwidth of the link set by the **bandwidth** command, or the link's default bandwidth if none is configured, when calculating how much bandwidth to use.

You can adjust this percentage on an interface or subinterface with the `ip bandwidth-percent eigrp as-number percent interface` configuration command. The as-number is the EIGRP autonomous system number. The percent parameter is the percentage of the configured bandwidth that EIGRP can use. You can set the percentage to a value greater than 100, which might be useful if the bandwidth is configured artificially low for routing policy reasons. Example 2-45 shows a configuration that allows EIGRP to use 40 kbps (200 percent of the configured bandwidth, 20 kbps) on the interface. It is essential to make sure that the line is provisioned to handle the configured capacity. (The next section, "Examples of EIGRP on WANs," provides some more examples of when this command is useful.)

Example 2-45. Adjusting the EIGRP Link Utilization

```
Router(config)#interface serial0/0/0
Router(config-if)#bandwidth 20
Router(config-if)#ip bandwidth-percent eigrp 1 200
```

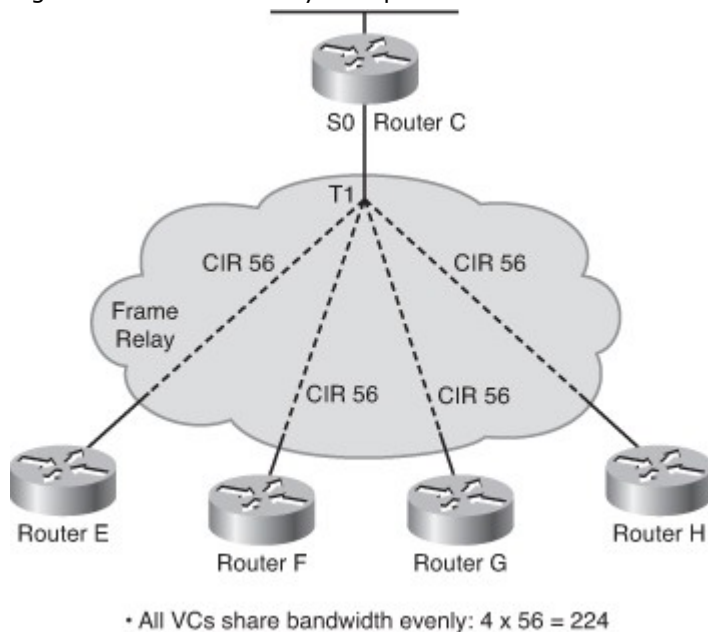
The Cisco IOS assumes that point-to-point Frame Relay subinterfaces are operating at the default speed of the interface. In many implementations, however, only fractional speeds (such as a fractional T1) are available. Therefore, when configuring these subinterfaces, set the bandwidth to match the contracted CIR. When configuring multipoint interfaces (especially for Frame Relay, but also for ATM and ISDN PRI), remember that the bandwidth is shared equally by all neighbors. That is, EIGRP uses the **bandwidth** command on the physical interface divided by the number of Frame Relay neighbors connected on that physical interface to get the bandwidth attributed to each neighbor. EIGRP configuration should reflect the correct percentage of the actual available bandwidth on the line.

Each installation has a unique topology, and with that comes unique configurations. Differing CIR values often require a hybrid configuration that blends the characteristics of point-to-point circuits with multipoint circuits. When configuring multipoint interfaces, configure the bandwidth to represent the minimum CIR times the number of circuits. This approach might not fully use the higher-speed circuits, but it ensures that the circuits with the lowest CIR will not be overdriven. If the topology has a small number of very low-speed circuits, these interfaces are typically defined as point-to-point so that their bandwidth can be set to match the provisioned CIR.

Examples of EIGRP on WANs

In Figure 2-37, Router C's interface has been configured for a bandwidth of 224 kbps. Four neighbors exist in this multipoint topology, so each circuit is allocated one-quarter of the configured bandwidth on the interface, resulting in $224 / 4 = 56$ kbps allocated per circuit. This 56-kbps allocation matches the provisioned CIR of each circuit.

Figure 2-37. Frame Relay Multipoint in Which All VCs Share the Bandwidth Evenly.



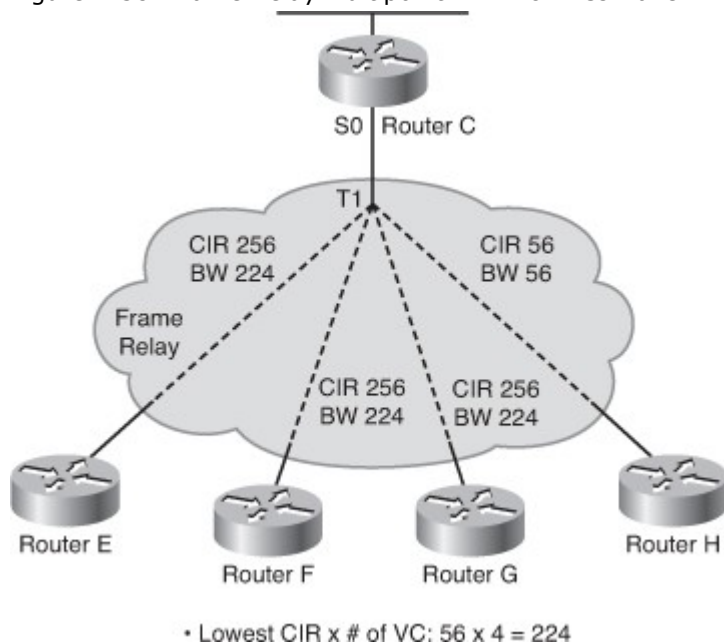
Example 2-46 shows the configuration for Router C's Serial 0 interface.

Example 2-46. Adjusting the bandwidth Command on an Interface on Router C in Figure 2-37

```
RouterC(config)#interface serial 0  
RouterC(config-if)#encapsulation frame-relay  
RouterC(config-if)#bandwidth 224
```

In Figure 2-38, one circuit has been provisioned for a 56-kbps CIR, and the other three circuits have a higher CIR of 256 kbps. The interface on Router C has been configured for a bandwidth equal to the lowest CIR multiplied by the number of circuits being supported ($56 \times 4 = 224$), as shown in Example 2-47. This configuration protects against overwhelming the slowest-speed circuit in the topology.

Figure 2-38. Frame Relay Multipoint in Which VCs Have Different CIRs.

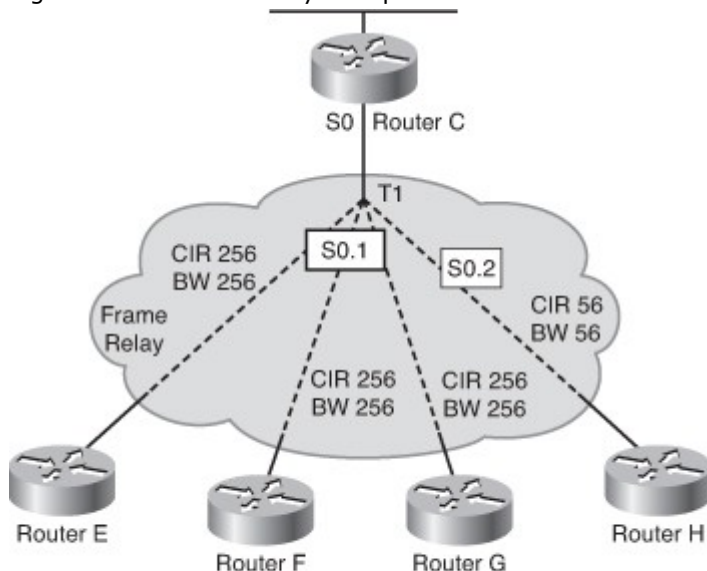


Example 2-47. Adjusting the bandwidth Command on an Interface on Router C in Figure 2-38

```
RouterC(config)#interface serial 0  
RouterC(config-if)#encapsulation frame-relay  
RouterC(config-if)#bandwidth 224
```

Figure 2-39 presents a hybrid solution to this network.

Figure 2-39. Frame Relay Multipoint and Point-to-Point.



- Configure lowest CIR VC as point-to-point, specify BW = CIR
- Configure higher CIR VCs as multipoint, combine CIRs

Example 2-48 shows the configuration applied to Router C in Figure 2-39.

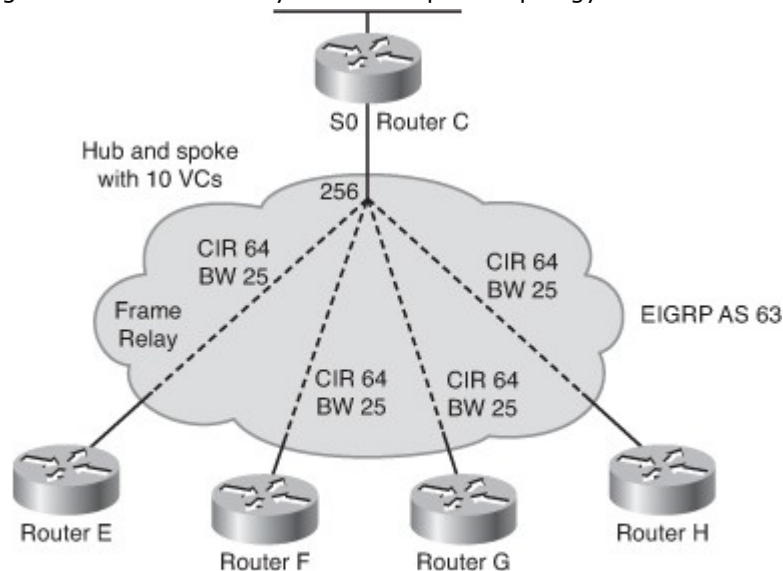
Example 2-48. Adjusting the Bandwidth for a Frame Relay Subinterface on Router C in Figure 2-39

```
RouterC(config)#interface serial 0.1 multipoint  
RouterC(config-subif)#bandwidth 768  
RouterC(config-subif)#exit  
RouterC(config)#interface serial 0.2 point-to-point  
RouterC(config-subif)#bandwidth 56
```

Example 2-48 shows the low-speed circuit configured as point-to-point. The remaining circuits are designated as multipoint, and their respective CIRs are added up to set the interface's bandwidth ($256 + 256 + 256 = 768$). On the multipoint interface, the bandwidth is shared equally among all circuits. Thus, the bandwidth will be split into three, with 256 kbps allocated to each circuit.

Figure 2-40 illustrates a common hub-and-spoke oversubscribed topology with 10 VCs to the remote sites. (Only 4 of the 10 remote sites are shown in the figure.) Example 2-49 shows the configuration used on Routers C and G of Figure 2-40.

Figure 2-40. Frame Relay Hub-and-Spoke Topology.



- Configure each VC as point-to-point, specify BW = 1/10 of link capacity
- Increase EIGRP utilization to 50% of actual VC capacity

Example 2-49. EIGRP WAN Configuration: Point-to-Point Links on Routers C and G in Figure 2-40

```
RouterC(config)#interface serial 0.1 point-to-point
RouterC(config-subif)#bandwidth 25
RouterC(config-subif)#ip bandwidth-percent eigrp 63
128
<output omitted>
RouterC(config)#interface serial 0.10 point-to-point
RouterC(config-subif)#bandwidth 25
RouterC(config-subif)#ip bandwidth-percent eigrp 63
128

RouterG(config)#interface serial 0
RouterG(config-if)#bandwidth 25
RouterG(config-if)#ip bandwidth-percent eigrp 63 128
```

The circuits are provisioned as 64-kbps links, but there is insufficient bandwidth on Router C (the hub) to support the allocation. For example, if the hub tries to communicate to all remote sites at the same time, the bandwidth that is required exceeds the available link speed of 256 kbps for the hub—10 times the CIR of 64 kbps equals 640 kbps. In a point-to-point topology, all VCs are treated equally and are therefore configured for exactly one-tenth of the available link speed (25 kbps). (Alternatively the Serial 0 main interface could be configured with the **bandwidth 256** command.)

As mentioned, by default EIGRP uses 50 percent of a circuit's configured bandwidth. EIGRP configuration should reflect the correct percentage of the actual available bandwidth on the line. Therefore, in an attempt to ensure that EIGRP packets are delivered through the Frame Relay network in Figure 2-40, each subinterface has the EIGRP allocation percentage raised to 128 percent of the specified bandwidth. This adjustment causes EIGRP packets to receive approximately 32 kbps of the provisioned 64 kbps on each circuit (because 128% of 25 kbps is 32 kbps). This extra configuration restores the 50-50 ratio that was tampered with when the bandwidth was set to an artificially low value. If the number of VCs changes, these calculations and configurations must be redone.

Note

Suppressing EIGRP ACKs also saves bandwidth. An ACK is not sent if a unicast data packet is ready for transmission. The ACK field in any reliable unicast packet (RTP packet) is sufficient to acknowledge the neighbor's packet, so the ACK packet is suppressed to save bandwidth. This is a significant feature for point-

to-point links and NBMA networks, because on those media, all data packets are sent as unicasts and, thus, can carry an acknowledgment themselves (this is also known as a piggyback ACK). In that instance, there is no need for an ACK packet.

Configuring and Verifying EIGRP Authentication

You can prevent your router from receiving fraudulent route updates by configuring neighbor router authentication. You can configure EIGRP neighbor authentication (also called *neighbor router authentication* or *route authentication*) such that routers can participate in routing based on predefined passwords.

This section first describes router authentication in general, followed by a discussion of how to configure and troubleshoot EIGRP message digest 5 (MD5) authentication.

Router Authentication

You can configure neighbor router authentication such that routers only participate in routing based on predefined passwords.

By default, no authentication is used for routing protocol packets. Without neighbor authentication, unauthorized or deliberately malicious routing updates could compromise the security of network traffic. For example, an unauthorized router connected to the network (by someone with malicious or innocent intentions) could send a fictitious routing update to convince your router to send traffic to an incorrect destination.

When neighbor router authentication has been configured on a router, the router authenticates the source of each routing protocol packet that it receives. This is accomplished by the exchange of an authentication key (also called a *password*) that is known to both the sending and the receiving router.

Simple Authentication Versus MD5 Authentication

Routers use two types of authentication:

- **Simple password authentication (also called plain-text authentication)**— Supported by Integrated System-Integrated System (IS-IS) Protocol, OSPF, and Routing Information Protocol Version 2 (RIPv2)
- **MD5 authentication**— Supported by OSPF, RIPv2, BGP, and EIGRP

Note

This book covers authentication for EIGRP, OSPF, and BGP.

Both forms of authentication work in the same way, with the exception that MD5 sends a message digest rather than the authenticating key itself. The message digest is created using the key (and a key ID with some protocols) and a message, but the key itself is not sent, preventing it from being read while it is being transmitted. Simple password authentication sends the authenticating key itself over the wire.

Note

Simple password authentication is not recommended for use as part of your security strategy because it is vulnerable to passive attacks. Anybody with a link analyzer could easily view the password on the wire. The primary use of simple password authentication is to avoid accidental changes to the routing infrastructure. Using MD5 authentication, however, is a recommended security practice.

Caution

As with all keys, passwords, and other security secrets, it is imperative that you closely guard the keys used in neighbor authentication. The security benefits of this feature are reliant upon keeping all authenticating keys confidential. Also, when performing router management tasks via Simple Network Management Protocol (SNMP), do not ignore the risk associated with sending keys using nonencrypted SNMP.

With simple password authentication, a password (key) is configured on a router; each participating neighbor router must be configured with the same key. When a packet is sent, the password is included in plain text.

MD5 authentication, described in RFC 1321, *The MD5 Message-Digest Algorithm*, is a cryptographic authentication. A key (password) and key ID are configured on each router. The router uses an algorithm based on the routing protocol packet, the key, and the key ID to generate a message digest (also called a *hash*) that is appended to the packet. Unlike the simple authentication, the key is not exchanged over the wire—the message digest is sent instead of the key, which ensures that nobody can eavesdrop on the line and learn keys during transmission.

Note

MD5 provides authentication but does not provide confidentiality. The contents of routing protocol packets are not encrypted.

MD5 Authentication for EIGRP

By default, no authentication is used for EIGRP packets. You can configure EIGRP to use MD5 authentication. When EIGRP neighbor authentication has been configured on a router, the router authenticates the source of each routing protocol packet that it receives. The MD5 keyed digest in each EIGRP packet prevents the introduction of unauthorized or false routing messages from unapproved sources.

For EIGRP MD5 authentication, you must configure an authenticating *key* and a *key ID* on both the sending router and the receiving router. Each EIGRP router takes the key and key ID and generates message digest that is appended to each routing protocol packet and sent to the neighbor. The receiving router computes the MD5 hash from the received EIGRP information. If the hash matches with the value received, the packet is accepted. If there is no match, the packet is silently dropped.

Each key has its own key ID, which is stored locally. The combination of the key ID and the interface associated with the message uniquely identifies the authentication algorithm and MD5 authentication key in use.

You can increase the security of EIGRP MD5 authentication by making frequent key changes. You can define and activate multiple keys based on time, as defined in the configuration. Transition between the keys is implemented in such way that it allows a nondisruptive operation of EIGRP exchanges. The key changes must be well planned and supported by time synchronization between the routers. The rollover to a new key works only if the time on adjacent routers is synchronized.

Note

EIGRP routers need to know the time to be able to rotate through keys in synchronization with the other participating routers, to ensure that all routers are using the same key at the same moment. Several mechanisms can be used for time synchronization; Network Time Protocol (NTP) is the most common.

EIGRP allows multiple keys to be managed using *key chains*. Each key definition within the key chain can include a time interval for which that key will be activated (known as its lifetime). Then, during a given key's lifetime, routing update packets are sent with this activated key. Only one authentication packet is sent, regardless of how many valid keys exist. When sending, the software examines the key numbers in order from lowest to highest, and it uses the first valid key it encounters. Incoming packets are checked using all valid keys.

When configuring EIGRP authentication, you specify the key ID (number), the key (password), and optionally the lifetime of the key.

The first (by key ID), valid (by lifetime) key is used when sending packets. In other words, when sending EIGRP packets, the valid key with the lowest key number on the key chain is used. When packets are received, all currently valid keys are checked until a match is found.

Keys cannot be used during time periods for which they are not activated. Therefore, it is recommended that for a given key chain, key activation times overlap to avoid any period of time for which no key is activated. If a time period occurs during which no key is activated, neighbor authentication cannot occur, and therefore routing updates will fail.

Planning for EIGRP Authentication

Before configuring EIGRP authentication, the network administrator must examine the existing EIGRP configuration and define the authentication requirements. The EIGRP authentication requirements include the authentication type (none or MD5), the number of keys used, and the optional lifetime parameters.

The parameters must then be defined in enough details for the network operator to configure EIGRP authentication. These parameters include the following:

- The EIGRP autonomous system number
- The authentication mode (MD5)
- The definition of one or more keys to authenticate EIGRP packets, according to the network security plan
- The keys' lifetime, if multiple keys are defined

At a high level, configuring EIGRP MD5 authentication requires the following steps:

- Step 1. Configure the authentication mode for EIGRP.
- Step 2. Configure the key chain.

Step 3. Optionally configure the keys' lifetime parameters.

Step 4. Enable authentication to use the keys in the key chain.

The next section details how to configure EIGRP MD5 authentication.

Configuring EIGRP MD5 Authentication

MD5 authentication must be configured on all interfaces between two neighboring routers. To configure MD5 authentication for EIGRP, complete the following steps:

Step 1. Enter configuration mode for the interface on which you want to enable authentication.

Step 2. Specify MD5 authentication for EIGRP packets using the **ip authentication mode eigrp autonomous-system md5** interface configuration command. The *autonomous-system* is the EIGRP autonomous system number in which authentication is to be used.

Step 3. Enter the key-chain configuration mode for the key chain (that you will later configure on the interface) using the **key chain name-of-chain** global configuration command.

Step 4. Identify a key ID to use and enter configuration mode for that key (the key-chain-key configuration mode) using the **key key-id** key-chain configuration command. The *key-id* is the ID number of an authentication key on a key chain. The range of keys is from 0 to 2147483647. The key ID numbers need not be consecutive.

Step 5. Identify the key string (the password) for this key using the **key-string key** key-chain-key configuration command. The *key* is the authentication key-string that is to be used to authenticate sent and received EIGRP packets. The key string can contain from 1 to 80 uppercase and lowercase alphanumeric characters, except that the first character cannot be a number. The key string for a given key ID must be the same on neighboring routers and is case sensitive.

Step 6. Optionally specify the time period during which this key will be accepted for use on received packets using the **accept-lifetime start-time{infinite | end-time | duration seconds}** key-chain-key configuration command. Table 2-9 describes the parameters for this command.

Table 2-9. *accept-lifetime* Command Parameters

Parameter	Description
start-time	Beginning time that the key specified by the key command is valid for use on received packets. The syntax can be either of the following: hh:mm:ss month date year hh:mm:ss date month year The time and date syntax is described as follows: hh—hours mm—minutes ss—seconds month—first three letters of the month date—date (1-31) year—year (four digits) The default start time and the earliest acceptable date is January 1, 1993.
infinite	Indicates the key is valid for use on received packets from the <i>start-time</i> value on.
end-time	Indicates the key is valid for use on received packets from the <i>start-time</i> value until the <i>end-time</i> value. The syntax is the same as that for the <i>start-time</i> value. The <i>end-time</i> value must be after the <i>start-time</i> value. The default end time is an infinite time period.
seconds	Length of time (in seconds) that the key is valid for use on received packets. The range is from 1 to 2147483646.

Step 7. Optionally specify the time period during which this key can be used for sending packets using the `send-lifetime start-time {infinite | end-time | duration seconds} key-chain-key` configuration command. Table 2-10 describes the parameters for this command.

Table 2-10. *send-lifetime* Command Parameters

Parameter	Description
start-time	Beginning time that the key specified by the key command is valid to be used for sending packets. The syntax can be either of the following: hh:mm:ss month date year hh:mm:ss date month year The time and date syntax is described as follows: hh—hours mm—minutes ss—seconds month—first three letters of the month date—date (1-31) year—year (four digits) The default start time and the earliest acceptable date is January 1, 1993.
infinite	Indicates the key is valid to be used for sending packets from the <i>start-time</i> value on.
end-time	Indicates the key is valid to be used for sending packets from the <i>start-time</i> value until the <i>end-time</i> value. The syntax is the same as that for the <i>start-time</i> value. The <i>end-time</i> value must be after the <i>start-time</i> value. The default end time is an infinite time period.
seconds	Length of time (in seconds) that the key is valid to be used for sending packets. The range is from 1 to 2147483646.

Step 8. Enable the authentication of EIGRP packets with a key specified in a key chain by using the **ip authentication key-chain eigrp autonomous-system name-of-chain** interface configuration command. The *autonomous-system* parameter specifies the EIGRP autonomous system number in which authentication is to be used. The *name-of-chain* parameter specifies the name of the configured key chain from which a key is to be obtained for this interface.

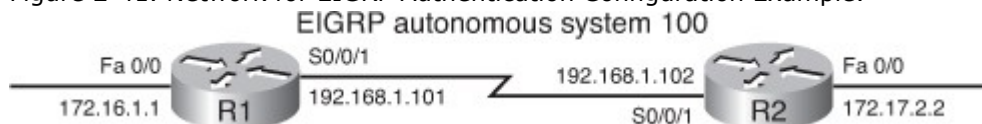
Note

If the **service password-encryption** command is not used when implementing EIGRP authentication, the key string will be stored as plain text in the router configuration. If you configure the **service password-encryption** command, the key string will be stored and displayed in an encrypted form; when it is displayed, there will be an *encryption-type* of 7 specified before the encrypted key string.

MD5 Authentication Configuration Example

Figure 2-41 shows the network used to illustrate the configuration, verification, and troubleshooting of MD5 authentication.

Figure 2-41. Network for EIGRP Authentication Configuration Example.



Example 2-50 shows the configuration of the R1 router.

Example 2-50. Configuration of Router R1 in Figure 2-41

```
Code View: Scroll / Show All
R1#show running-config
<output omitted>
key chain R1chain
  key 1
    key-string firstkey
    accept-lifetime 04:00:00 Jan 1 2009 infinite
    send-lifetime 04:00:00 Jan 1 2009 04:00:00 Jan 31 2009
  key 2
    key-string secondkey
    accept-lifetime 04:00:00 Jan 25 2009 infinite
    send-lifetime 04:00:00 Jan 25 2009 infinite
<output omitted>
interface FastEthernet0/0
ip address 172.16.1.1 255.255.255.0
!
interface Serial0/0/1
  bandwidth 64
  ip address 192.168.1.101 255.255.255.224
  ip authentication mode eigrp 100 md5
  ip authentication key-chain eigrp 100 R1chain
!
router eigrp 100
  network 172.16.1.0 0.0.0.255
  network 192.168.1.0
  auto-summary
```

EIGRP autonomous system 100 is used in this network.

MD5 authentication is configured on the serial 0/0/1 interface with the **ip authentication mode eigrp 100 md5** command. When MD5 authentication is configured, an MD5 keyed digest is added to each EIGRP packet sent and is checked in each received EIGRP packet.

The **key chain R1chain** command enters configuration mode for the *R1chain* key chain. Two keys are defined in this key chain. Each key has an authentication string and lifetime specified. The network administrator wants to change the keys on all the routers in the network each month to improve the security. The administrator configures an overlap of one week to change the keys on all the routers by configuring key 2 to be valid one week before key 1 expires.

Key 1 is set to *firstkey* with the **key-string firstkey** command. This key is acceptable for use on packets received by R1 from January 1, 2009 onward, as specified in the **accept-lifetime 04:00:00 Jan 1 2009 infinite** command. However, the **send-lifetime 04:00:00 Jan 1 2009 04:00:00 Jan 31 2009** command specifies that this key was only valid for use when sending packets until January 31, 2009. It is no longer valid for use in sending packets after January 31 2009.

Key 2 is set to *secondkey* with the **key-string secondkey** command. This key is acceptable for use on packets received by R1 from January 25, 2009 onward, as specified in the **accept-lifetime 04:00:00 Jan 25 2009 infinite** command. This key can also be used when sending packets from January 25, 2009 onward, as specified in the **send-lifetime 04:00:00 Jan 25 2009 infinite** command.

The **ip authentication key-chain eigrp 100 R1chain** command configured on the Serial 0/0/1 interface specifies that the EIGRP key chain R1chain is to be used on this interface.

Recall that the router uses the first, by key number, valid key for sending packets. As a result of this configuration, Router R1 will use key 1 for sending, from January 1 to 31, 2009, and will use key 2 for sending as of 4:00 a.m. on January 31, 2009. Router R1 will accept key 1 for received packets, from January

1, 2009, and will also accept key 2 for received packets, from January 25, 2009. All other MD5 packets will be dropped.

Example 2-51 shows the configuration of the R2 router.

Example 2-51. Configuration of Router R2 in Figure 2-41

Code View: Scroll / Show All

```
R2#show running-config
<output omitted>
key chain R2chain
key 1
  key-string firstkey
  accept-lifetime 04:00:00 Jan 1 2009 infinite
  send-lifetime 04:00:00 Jan 1 2009 infinite
key 2
  key-string secondkey
  accept-lifetime 04:00:00 Jan 25 2009 infinite
  send-lifetime 04:00:00 Jan 25 2009 infinite
<output omitted>
interface FastEthernet0/0
  ip address 172.17.2.2 255.255.255.0
!
interface Serial0/0/1
  bandwidth 64
  ip address 192.168.1.102 255.255.255.224
  ip authentication mode eigrp 100 md5
  ip authentication key-chain eigrp 100 R2chain
!
router eigrp 100
  network 172.17.2.0 0.0.0.255
  network 192.168.1.0
  auto-summary
```

MD5 authentication is configured on the Serial 0/0/1 interface with the **ip authentication mode eigrp 100 md5** command.

The **key chain R2chain** command enters configuration mode for the *R2chain* key chain. Two keys are defined. Key 1 is set to *firstkey* with the **key-string firstkey** command. This key is acceptable for use on packets received by R2 from January 1, 2009 onward, as specified in the **accept-lifetime 04:00:00 Jan 1 2009 infinite** command. This key can also be used when sending packets from January 1, 2009 onward, as specified in the **send-lifetime 04:00:00 Jan 1 2009 infinite** command.

Key 2 is set to *secondkey* with the **key-string secondkey** command. This key is acceptable for use on packets received by R2 from January 25, 2009 onward, as specified in the **accept-lifetime 04:00:00 Jan 25 2009 infinite** command. This key can also be used when sending packets from January 25, 2009 onward, as specified in the **send-lifetime 04:00:00 Jan 25 2009 infinite** command.

As a result of this configuration, Router R2 will use key 1 for sending, from January 1, 2009, because it is the first valid key in the key chain. (Of course, if key 1 is deleted in the future, key 2 will be used for sending.) Router R2 will accept key 1 for received packets, from January 1, 2009, and will also accept key 2 for received packets, from January 25, 2009. All other MD5 packets will be dropped.

The **ip authentication key-chain eigrp 100 R2chain** command configured on the Serial 0/0/1 interface specifies that the key chain R2chain is to be used on this interface.

Verifying MD5 Authentication for EIGRP

This section provides examples of MD5 authentication verification and troubleshooting.

EIGRP MD5 Authentication Verification

Example 2-52 provides the output of the show ip eigrp neighbors and show ip route commands on the R1 router depicted in the network in Figure 2-41. The neighbor table indicates that the two routers have successfully formed an EIGRP adjacency. The routing table verifies that the 172.17.0.0 network has been learned via EIGRP over the serial connection. Example 2-52 also shows the results of a ping to the R2 Fast Ethernet interface address to illustrate that the link is working.

Example 2-52. Output on Router R1 in Figure 2-41

```
Code View: Scroll / Show All
R1#
*Apr 21 16:23:30.517: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 192.168.1.102
(Serial0/0/1) is up: new adjacency

R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H  Address                Interface    Hold Uptime   SRTT  RTO  Q  Seq
      (sec)              (ms)        Cnt Num
0  192.168.1.102          Se0/0/1      12 00:03:10   17  2280  0  14
R1#show ip route
<output omitted>
Gateway of last resort is not set
D   172.17.0.0/16 [90/40514560] via 192.168.1.102, 00:02:22, Serial0/0/1
    172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
D     172.16.0.0/16 is a summary, 00:31:31, Null0
C     172.16.1.0/24 is directly connected, FastEthernet0/0
    192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C     192.168.1.96/27 is directly connected, Serial0/0/1
D     192.168.1.0/24 is a summary, 00:31:31, Null0
R1#ping 172.17.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.17.2.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/15/16 ms
```

You can use the show key chain [name-of-chain] verification command to see the key chain, key string, and the lifetime of the keys configured under the key chain. Example 2-53 shows the keys on the R1 router. This command is useful to verify that the keys are the same on both neighbors and that the lifetime is set properly.

Example 2-53. show key chain Command Output on Router R1 in Figure 2-41

```
R1#show key chain
Key-chain R1chain:
  key 1 — text "firstkey"
    accept lifetime (04:00:00 Jan 1 2009) - (always valid) [valid now]
    send lifetime (04:00:00 Jan 1 2009) - (04:00:00 Jan 31 2009)
  key 2 — text "secondkey"
    accept lifetime (04:00:00 Jan 25 2009) - (always valid) [valid now]
    send lifetime (04:00:00 Jan 25 2009) - (always valid) [valid now]
```

Key chain R1chain and both keys key 1 (with authentication string **firstkey**) and key 2 (with authentication string **secondkey**) are displayed. Under each key, the lifetime of the key is also shown. By observing the same output from the neighboring Router R2, the configuration can be verified.

The **show ip eigrp interface detail** command can also be used to display detailed information about the EIGRP interfaces for a specific EIGRP process, including the authentication mode and the key chain configured on the interface.

Troubleshooting MD5 Authentication

This section provides some examples of how to troubleshoot EIGRP MD5 authentication.

As discussed earlier, the **debug eigrp packets** command displays general debugging information and is useful for analyzing the messages traveling between the local and remote devices, including authentication messages.

Example of Successful MD5 Authentication

Example 2-54 shows output from the debug eigrp packets command on R1, which displays that R1 is receiving EIGRP packets with MD5 authentication, with a key ID equal to 1, from R2.

Example 2-54. debug eigrp packets Command Output on Router R1 in Figure 2-41

Code View: Scroll / Show All

R1#**debug eigrp packets**

EIGRP Packets debugging is on

(UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY, SIAREPLY)

*Apr 21 16:38:51.745: EIGRP: received packet with MD5 authentication, key id = 1

*Apr 21 16:38:51.745: EIGRP: Received HELLO on Serial0/0/1 nbr 192.168.1.102

*Apr 21 16:38:51.745: AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 0/0

Similarly, the output of the debug eigrp packets command on R2 shown in Example 2-55 illustrates that R2 is receiving EIGRP packets with MD5 authentication, with a key ID equal to 2, from R1. (Router R1 is using key 2 because this output was taken in April, after key 1 expired for sending in Router R1.)

Example 2-55. debug eigrp packets Command Output on Router R2 in Figure 2-41

Code View: Scroll / Show All

R2#**debug eigrp packets**

EIGRP Packets debugging is on

(UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY, SIAREPLY)

R2#

*Apr 21 16:38:38.321: EIGRP: received packet with MD5 authentication, key id = 2

*Apr 21 16:38:38.321: EIGRP: Received HELLO on Serial0/0/1 nbr 192.168.1.101

*Apr 21 16:38:38.321: AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 0/0

Example of Troubleshooting MD5 Authentication Problems

For this example, the key string for Router R1's key 2, the one that it uses when sending EIGRP packets, is changed to be different from the key string that Router R2 is expecting. Example 2-56 shows the changes to the configuration for R1.

Example 2-56. Changes to the Configuration of Router R1 in Figure 2-41

R1(config-if)#**key chain R1chain**

R1(config-keychain)#**key 2**


```
R1(config-keychain-key)#key-string  
wrongkey
```

The output of the debug eigrp packets command on R2 shown in Example 2-57 illustrates that R2 is receiving EIGRP packets with MD5 authentication, with a key ID equal to 2, from R1, but that there is an authentication mismatch. The EIGRP packets from R1 are ignored, and the neighbor relationship is declared to be down. The output of the show ip eigrp neighbors command confirms that R2 does not have any EIGRP neighbors.

Example 2-57. Output on Router R2 in Figure 2-41

```
Code View: Scroll / Show All  
R2#debug eigrp packets  
EIGRP Packets debugging is on  
  (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY,  
  SIAREPLY)  
R2#  
*Apr 21 16:50:18.749: EIGRP: pkt key id = 2, authentication mismatch  
*Apr 21 16:50:18.749: EIGRP: Serial0/0/1: ignored packet from 192.168.1.101,  
opcode = 5 (invalid authentication)  
*Apr 21 16:50:18.749: EIGRP: Dropping peer, invalid authentication  
*Apr 21 16:50:18.749: EIGRP: Sending HELLO on Serial0/0/1  
*Apr 21 16:50:18.749:  AS 100, Flags 0x0, Seq 0/0 idbQ 0/0 iibQ un/rely 0/0  
*Apr 21 16:50:18.753: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor  
192.168.1.101  
  (Serial0/0/1) is down: Auth failure  
R2#show ip eigrp neighbors  
IP-EIGRP neighbors for process 100  
R2#
```

The two routers keep trying to reestablish their neighbor relationship. Because of the different keys used by each router in this scenario, R1 will authenticate hello messages sent by R2 using key 1. When R1 sends a hello message back to R2 using key 2, however, an authentication mismatch exists. From R1's perspective, the relationship appears to be up for a while, but then it times out, as illustrated by the messages received on R1 shown in Example 2-58. The output of the show ip eigrp neighbors command on R1 also illustrates that R1 does have R2 in its neighbor table for a short time.

Example 2-58. Output on Router R1 in Figure 2-41

```
Code View: Scroll / Show All  
R1#  
*Apr 21 16:54:09.821: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor  
192.168.1.102  
(Serial0/0/1) is down: retry limit exceeded  
*Apr 21 16:54:11.745: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor  
192.168.1.102  
(Serial0/0/1) is up: new adjacency  
R1#show ip eigrp neighbors  
H  Address      Interface  Hold Uptime  SRTT  RTO  Q  Seq  
   (sec)         (ms)      Cnt Num  
0  192.168.1.102 Se0/0/1    13 00:00:38  1    5000 1  0
```

This section dealt with EIGRP authentication. The following section provides information on other features that can be implemented in EIGRP to optimize its operation.

Optimizing EIGRP Implementations

EIGRP is a scalable routing protocol that ensures that as a network grows larger, it operates efficiently and adjusts rapidly to changes. This section describes practical EIGRP-specific techniques to implement an effective, scalable enterprise network.

EIGRP Scalability in Large Networks

Operating one large flat EIGRP network is normally not scalable. Some issues to consider include the following:

- Large routing tables that need to be processed
- High memory demands resulting from a large topology table, a large number of routes in a routing table, and in some environments (for example, on central site concentration routers), a large number of neighbors in the neighbor table
- High bandwidth demands resulting from the exchange of a large number of routing updates, and sending many queries and replies within a large EIGRP domain (which possibly includes links with low bandwidth and links with a significant number of transmission errors)

The following are some of the many variables that affect network scalability:

- **The amount of information exchanged between neighbors—** If more information than necessary for routing to function correctly is exchanged between EIGRP neighbors, unnecessary work during routing startup and topology changes results.
- **The number of routers—** When a topology change occurs, the amount of resources consumed by EIGRP is directly related to the number of routers that must be involved in the change.
- **The topology's depth—** The topology's depth can affect the convergence time. Depth refers to the number of hops that information must travel to reach all routers. For example, a multinational network without route summarization has a large depth and therefore increased convergence time. A three-tiered network design (as described in Chapter 1) is highly recommended for all IP routing environments. The recommendation is that there should not be more than seven hops between any two routing devices on an enterprise internetwork because the propagation delay and the query process across multiple hops when changes occur can slow down the convergence of the network when routes are lost.
- **The number of alternative paths through the network—** A network should provide alternative paths to avoid single points of failure. However, too many alternative paths can create problems with EIGRP convergence, because the EIGRP routing process, using queries, needs to explore all possible paths for lost routes. This complexity creates an ideal condition for a router to become stuck-in-active (SIA)—described in the next section—as it awaits a response to queries that are being propagated through these many alternative paths.

When implementing EIGRP as the routing protocol, some design challenges need to be addressed. Three major factors are:

- The size of the topology and routing tables, which is affected by the number of neighboring routers.
- The number of changes in the network.
- The EIGRP load on the WAN.

These three factors dictate the EIGRP design and where query boundaries should be introduced (using summarization, redistribution, and so on, as described later in this chapter). Without any boundaries, queries are propagated throughout the EIGRP domain, and very often, all the routers become involved in a DUAL computation, resulting in high bandwidth utilization and CPU load. Frequent DUAL computations have an effect on all tables maintained by the routers, including EIGRP tables and the caches built during the forwarding process.

EIGRP Queries and Stuck-in-Active

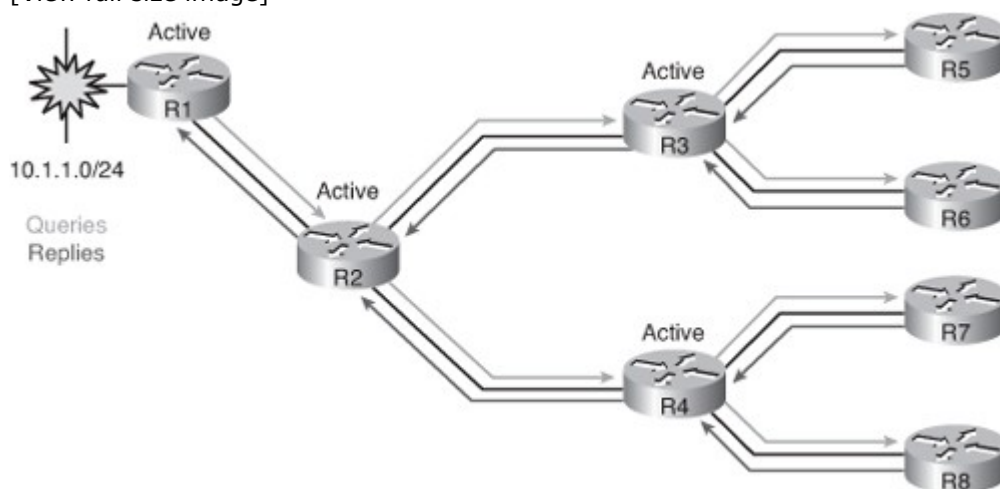
As an advanced distance vector routing protocol, EIGRP relies on its neighbors to provide routing information. Recall that when a router loses a route and does not have an FS in its topology table, it looks for an alternative path to the destination. This is known as *going active* on a route. (Recall that a route is considered passive when a router is not recomputing that route.) Recomputing a route involves sending query packets to all neighbors on interfaces other than the one used to reach the previous successor (because of split horizon), inquiring whether they have a route to the given destination. If a router has an

alternative route, it answers the query with a reply packet and does not propagate the query further. The reply includes the alternate route. If a neighbor does not have an alternative route, it queries each of its own neighbors for an alternative path. The queries then propagate through the network, thus creating an expanding tree of queries. When a router answers a query, it stops the spread of the query through that branch of the network. However, the query can still spread through other portions of the network as other routers attempt to find alternative paths, which might not exist.

Figure 2-42 presents a network example where a single lost route might result in an enormous number of queries sent throughout the EIGRP domain. The route to the network 10.1.1.0 on Router R1 is lost and Router R1 sends a query to all neighboring routers on all interfaces except the interface of the successor (because of split horizon). In this case, the query is sent to Router R2. Router R2 cascades the query to its neighbors because it has no information about the lost route, and so on. Each query requires a reply from the neighbor, and the amount of EIGRP traffic increases. In this network topology, no redundant path to network 10.1.1.0 is available, and the EIGRP query propagation process is far from being efficient. Many queries are sent, and each query is followed by a reply. Several solutions exist to optimize the query propagation process and to limit the amount of unnecessary EIGRP load on the links, including summarization, redistribution, and using the EIGRP stub routing feature.

Figure 2-42. EIGRP Query Process.

[View full size image]



Note

EIGRP summarization and stub routing features are explored later in this chapter. Details of how routes are redistributed are covered in Chapter 4.

The following sections describe how EIGRP may get stuck in the active state for a route, and how to prevent this from happening.

Stuck-in-Active Connections in EIGRP

Because of the reliable multicast approach used by EIGRP when searching for an alternative to a lost route, it is imperative that a reply be received for each query generated in the network. In other words, when a route goes active and queries are initiated, the only way this route can come out of the active state and transition to passive state is by receiving a reply for every generated query.

If the router does not receive a reply to all the outstanding queries within 3 minutes (the default time), the route goes to the SIA state.

Note

You can change the active-state time limit from its default of 3 minutes using the **timers active-time** [*time-limit* | **disabled**] router configuration command. The *time-limit* is in minutes.

When a route goes to SIA state, the router resets the neighbor relationships for the neighbors that failed to reply. This causes the router to recompute all routes known through that neighbor and to readvertise all the routes it knows about to that neighbor.

The most common reasons for SIA routes are as follows:

- **The router is too busy to answer the query**— Generally resulting from high CPU usage or memory problems; the router cannot allocate memory to process the query or build the reply packet.
- **The link between the two routers is not good**— This issue results in lost packets between the routers. The router receives an adequate number of packets to maintain the neighbor relationship, but does not receive all queries or replies.
- **A failure causes traffic on a link to flow in only one direction**— This is called a *unidirectional link*.

Note

Use the **eigrp log-neighbor-changes** command to enable logging of neighbor adjacency changes, to monitor the routing system's stability and to help detect problems related to SIA.

One erroneous approach for decreasing the chances of an SIA route is to use multiple EIGRP autonomous systems to bound the query range. Many networks have been implemented using multiple EIGRP autonomous systems (to somewhat simulate OSPF areas), with mutual redistribution between the different autonomous systems. Although this approach changes how the network behaves, it does not always achieve the results intended. If a query reaches the edge of the autonomous system (where routes are redistributed into another autonomous system), the original query is answered. However, the edge router then initiates a new query in the other autonomous system. Therefore, the query process has not been stopped, and the querying continues in the other autonomous system, where the route can potentially go in SIA.

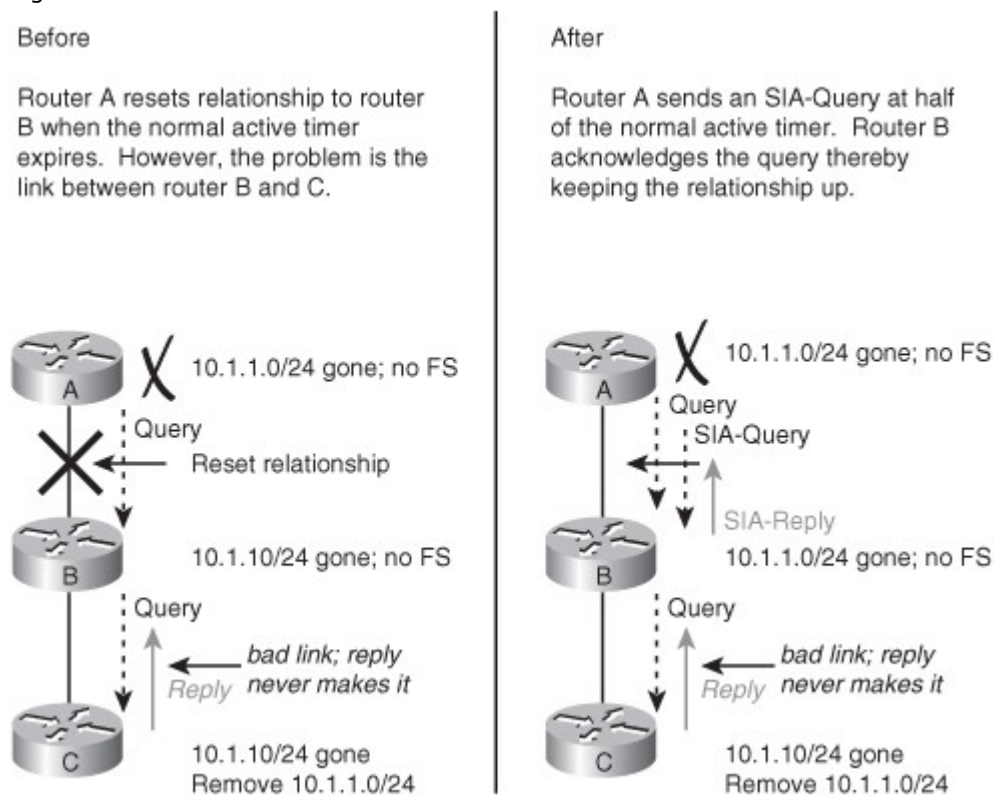
Another misconception about autonomous system boundaries is that implementing multiple autonomous systems protects one autonomous system from route flaps in another autonomous system. If routes are redistributed between autonomous systems, route transitions from one autonomous system are detected in the other autonomous systems.

Preventing SIA Connections

SIA-Query and SIA-Reply are two new additions to the Type, Length, Value (TLV) triplets in the EIGRP packet header. These packets are generated automatically with no configuration required, from Cisco IOS Software Release 12.1(5) and later, with the *Active Process Enhancement* feature. This feature enables an EIGRP router to monitor the progression of the search for a successor route and ensure that the neighbor is still reachable. The result is improved network reliability by reducing the unintended termination of neighbor adjacency.

The diagram on the left in Figure 2-43 illustrates what would happen before this feature was introduced. Router A sends a query for network 10.1.1.0/24 to Router B when it loses its connection to that network. Router B has no entry for this network, so it queries Router C. If problems exist between Router B and C, the reply packet from Router C to Router B might be delayed or lost. Router A has no visibility of downstream progress and assumes that no response indicates problems with Router B. After Router A's 3-minute active timer expires, the neighbor relationship with Router B is reset, along with all known routes from Router B.

Figure 2-43. Cisco IOS Active Process Enhancement.



In contrast, with the Active Process Enhancement feature, as illustrated in the diagram on the right in Figure 2-43, Router A queries downstream Router B (with an SIA-Query) at the midway point of the active timer (1.5 minutes by default) about the status of the route. Router B responds (with an SIA-Reply) that it is searching for a replacement route. Upon receiving this SIA-Reply response packet, Router A validates the status of Router B and does not terminate the neighbor relationship.

Meanwhile, Router B will send up to three SIA-Queries to Router C. If they go unanswered, Router B will terminate the neighbor relationship with Router C. Router B will then update Router A with an SIA-Reply indicating that the network 10.1.1.0/24 is unreachable. Routers A and B will remove the active route from their topology tables. The neighbor relationship between Routers A and B remains intact.

EIGRP Query Range

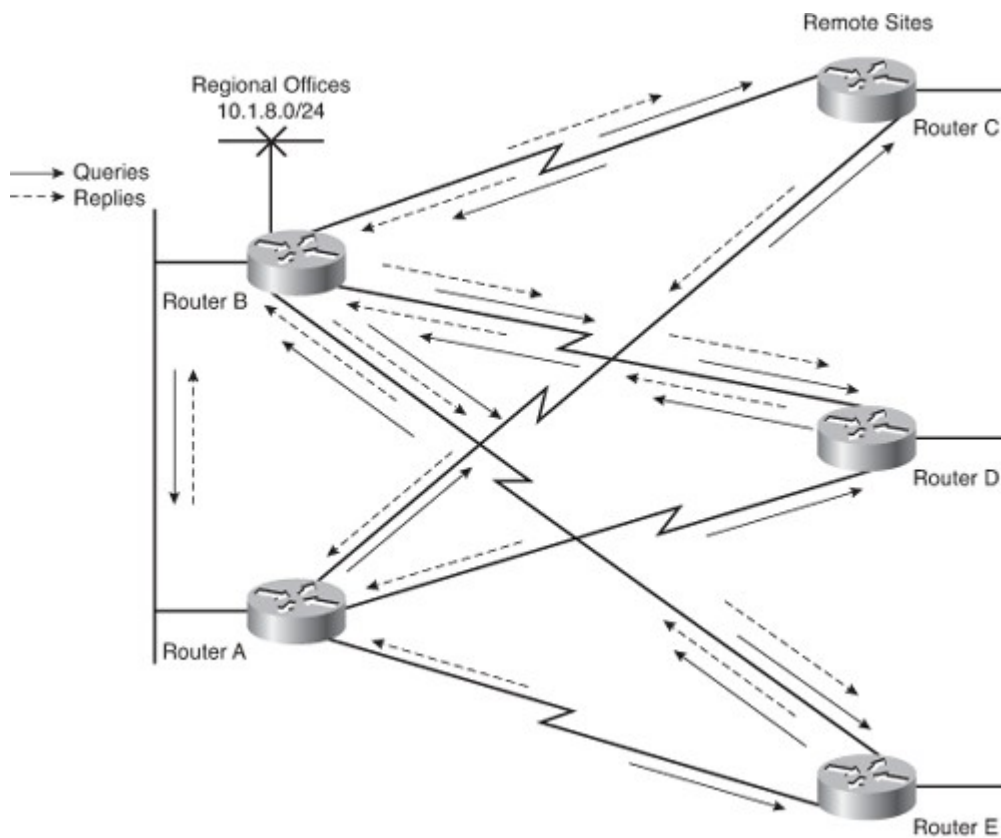
Limiting the scope of query propagation through the network (the query range), also known as *query scoping*, helps reduce incidences of SIA. Keeping the query packets close to the source reduces the chance that an isolated failure in another part of the network will restrict the convergence (query/reply) process. This section introduces an example that examines how to manage the query range.

Remote routers rarely need to know all the routes advertised in an entire network. Therefore, it is the network manager's responsibility to look at what information is necessary to properly route user traffic and to consider the use of a default route.

For example, in Figure 2-44, Router B notices the loss of network 10.1.8.0 and sends a query to Routers A, C, D, and E. In turn, these routers send queries to their neighbors, looking for a route to 10.1.8.0. When the query process starts, each path receives duplicate queries because of the redundant topology. Therefore, not only are the remote routers required to respond to queries from the regional offices, but they also continue the search by reflecting the queries back toward the other regional office's router. This significantly complicates the convergence process on the network.

Figure 2-44. Effect of the EIGRP Update and Query Process.

[View full size image]



In this sample network with only two regional and three remote routers, the problem might not be very significant. In a network with hundreds of remote offices, the problem can be severe.

Examine the query process for the 10.1.8.0/24 subnet. Initially, Router B advertises 10.1.8.0/24 to all other routers. The best path for Router A to reach 10.1.8.0/24 is over the Ethernet link to Router B. The remote routers (C, D, and E) use the serial link to Router B as their preferred path to reach 10.1.8.0/24 but still learn about an alternative path through Router A. For this example, assume that the EIGRP metric for Ethernet is 1000, and the metric for a serial link is 100,000.

Table 2-11 shows the content of the IP EIGRP topology table on Routers C, D, and E for network 10.1.8.0/24. Table 2-12 shows the content of the IP EIGRP topology table on Router A for network 10.1.8.0/24.

Table 2-11. IP EIGRP Topology Table for 10.1.8.0/24 on Routers C, D, and E in Figure 2-44

Neighbor	FD	AD
Router A	102,000	2000
Router B	101,000	1000

Table 2-12. IP EIGRP Topology Table for 10.1.8.0/24 on Router A in Figure 2-44

Neighbor	FD	AD
Router B	2000	1000
Router C	201,000	101,000
Router D	201,000	101,000
Router E	201,000	101,000

Note that Routers C, D, and E determine that for network 10.1.8.0/24, Router B is the successor and Router A is an FS (because the AD is 2000 through Router A, which is less than the FD through Router B). Also, note that Router A does not have an FS, because all paths through the remote routers have an AD larger than the FD through Router B.

When Router B loses the path to network 10.1.8.0/24, it queries all four of its neighbors. When the remote sites receive this query, they automatically install the path through Router A in their routing tables and respond to Router B with their supposedly good path through Router A. They also remove the bad path through Router B from their topology tables.

Router B now has responses to three of its four queries, but it must wait until Router A responds, too.

When Router A receives the query from Router B for network 10.1.8.0/24, Router A creates a query and sends it to Routers C, D, and E, because Router A does not have an FS but knows that a path exists through each remote site to reach 10.1.8.0/24.

Routers C, D, and E receive the query from Router A. They now know that their path through Router A is not good, so they check their topology tables for alternative paths. However, none of these routers currently has another path, because Router B has just informed them that it does not have a path to this network. Because the remote routers do not have an answer to the query from Router A, Routers C, D, and E create a query and send it to all neighbors except the neighbor (interface) that these routers received the original query from. In this case, the remote routers send the query only to Router B.

Router B learns from these queries that none of the remote routers has a path to network 10.1.8.0/24, but it cannot respond that it does not know of a path, because Router B is waiting for Router A to reply to a query. Router A is waiting for either Router C, D, or E to reply to its query, and these remote sites are waiting for Router B to reply to their queries. Because Router B sent out the first query, its SIA timer expires first, and Router B reaches the SIA state for network 10.1.8.0/24 first (in 3 minutes by default). Router B resets its neighbor relationship with Router A. As soon as the neighbor relationship goes down, Router B can respond to Routers C, D, and E immediately, saying that Router B does not have a path to 10.1.8.0/24. Routers C, D, and E can then respond to Router A that they do not have a path.

After the EIGRP neighbor relationship between Routers A and B is reestablished (just after the adjacency is reset), Router B, which no longer has a path to 10.1.8.0/24, does not pass the 10.1.8.0/24 network to Router A. Router A learns that the remote sites do not have a path to 10.1.8.0/24, and the new relationship with Router B does not include a path to 10.1.8.0/24, so Router A removes the 10.1.8.0 network from its IP EIGRP topology table.

In the network in Figure 2-44, redundancy is provided with dual links from the regional offices to the remote sites. The network architect does not intend for the traffic to go from a regional office to a remote office and back to a regional office, but unfortunately this is the situation. The design of the network shown in Figure 2-44 is sound, but because of EIGRP behavior, remote routers are involved in the convergence process.

If the remote sites are not supposed to act as transit sites between the regional sites, the regional routers can be configured to announce only a default route to the remote routers, and the remote routers can be configured to announce only their directly connected stub network to the regional routers, to reduce the complexity and the EIGRP topology table and routing table size.

The following sections describe other solutions for limiting the EIGRP query range.

Limiting the EIGRP Query Range

The network manager must determine the information necessary to properly route user traffic to the appropriate destination. The amount of information needed by the remote routers to achieve the desired level of path selection must be balanced against the bandwidth used to propagate this information. To achieve maximum stability and scalability, the remote routers can use a default route to reach the core. If some specific networks need knowledge of more routes to ensure optimum path selection, the administrator must determine whether the benefits of propagating the additional routing information outweigh the additional bandwidth required to achieve this goal.

In a properly designed network, each remote site has redundant WAN links to separate distribution sites. If both distribution sites pass a default route to the remote sites, the remote sites load balance to all networks behind the distribution site routers. This maximizes bandwidth utilization and allows the remote router to use less CPU and memory, which means that a smaller and less-expensive remote router can be used at that site.

If the remote site can see all routes, the router can select a path that is best to reach a given network. However, depending on the number of routes in the internetwork and the amount of bandwidth connecting the remote site to the distribution sites, this approach can mean that higher-bandwidth links or large routers are needed to handle the additional overhead.

After you determine the minimum routing requirements, you can make EIGRP more scalable. Two of the best options are the following:

- Configure route summarization using the **ip summary-address eigrp** command on the outbound interfaces of the appropriate routers.
- Configure the remote routers as stub EIGRP routers.

These methods are described in the next two sections.

Other methods to limit query range include route filtering and interface packet filtering. For example, if specific EIGRP routing updates are filtered to a router and that router receives a query about those filtered (blocked) networks, the router indicates that the network is unreachable and does not extend the query any further.

Limiting Query Range with Summarization

Route summarization is most effective with a sound address allocation. Having a two- or three-layer hierarchical network design, with routers summarizing between the layers greatly assists traffic flow and route distribution.

Note

For a full discussion of internetwork design, see *Authorized Self-Study Guide: Designing for Cisco Internetwork Solutions (DESGN)*, Second Edition (Cisco Press, 2008).

Figure 2-45 shows the topology of a nonscalable internetwork in which addresses (subnets) are either randomly assigned or assigned by historical requirements. In this example, multiple subnets from different major networks are located in each cloud, requiring many subnet routes to be injected into the core. In addition, because of the random assignment of addresses, query traffic cannot be localized to any portion of the network, thus increasing convergence time. Administration and troubleshooting are also more complex in this scenario.

Figure 2-45. Nonscalable Internetwork.

[View full size image]

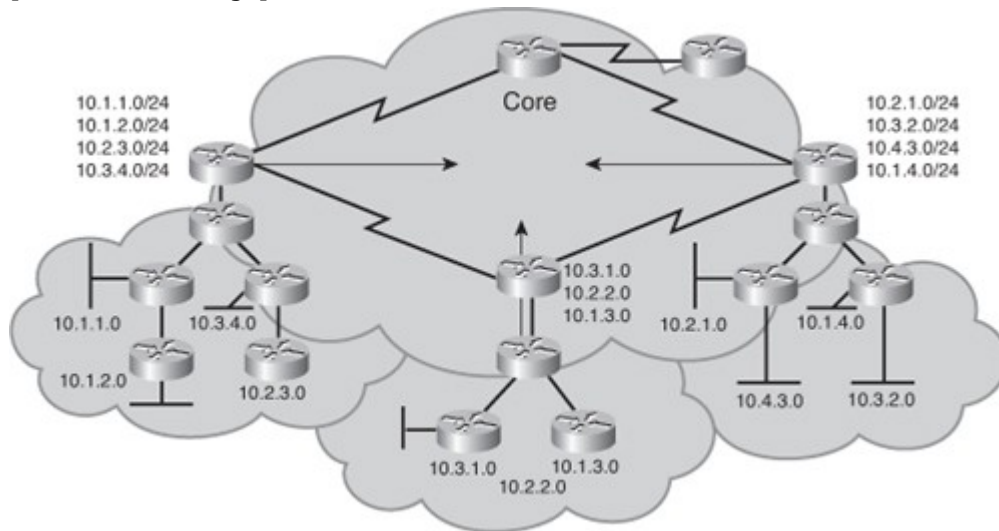


Figure 2-46 illustrates a better-designed network. Subnet addresses from individual major networks are localized within each cloud, allowing summary routes to be injected into the core. As an added benefit, the summary routes act as a boundary for the queries generated by a topology change.

Figure 2-46. Scalable Internetwork.

[View full size image]

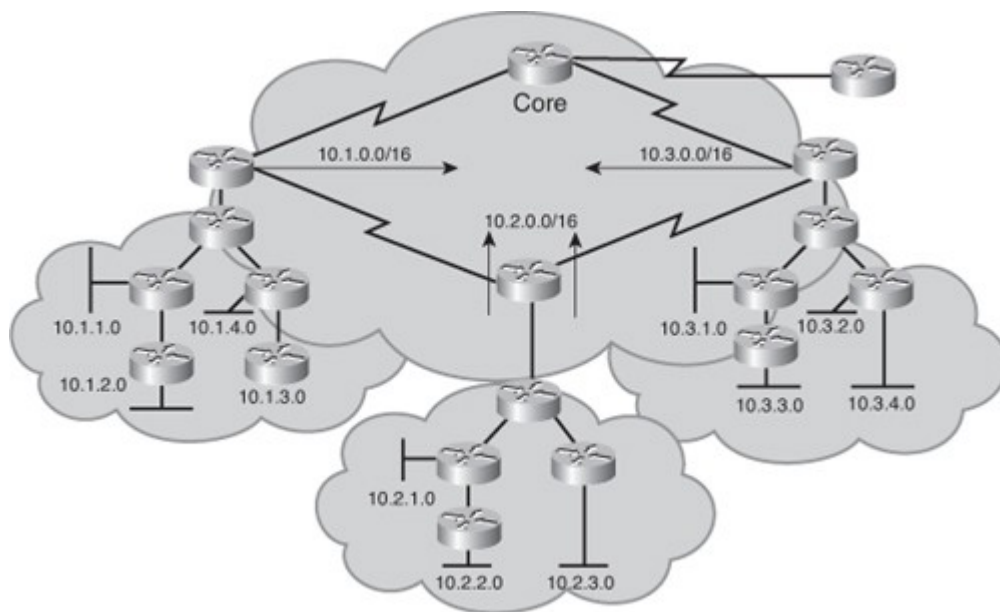
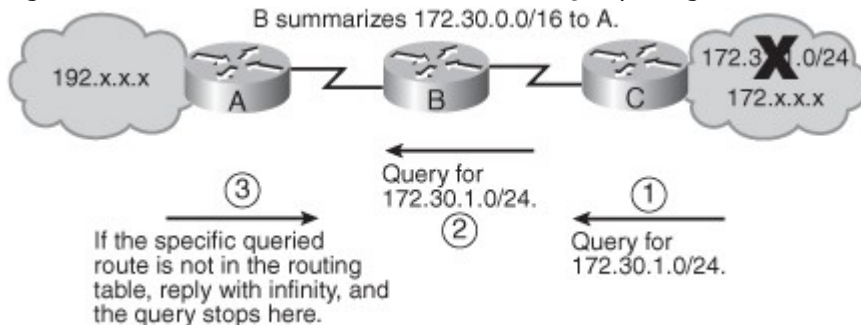


Figure 2-47 shows an example network to illustrate how EIGRP summarization can limit the query range. Router B sends a summary route of 172.30.0.0/16 to Router A. When network 172.30.1.0/24 goes down, Router A receives a query from Router B about that network. Because Router A has received only a summary route, that specific network is not in its routing table and so Router A replies to the query with a "network 172.30.1.0/24 unreachable" message and does not extend the query any further. Notice that the query stops at the router that receives the summary route (Router A in this example), not at the router that sends the summary route (Router B in this example). A remote router extends the query about a network only if it has an exact match for the network in its routing table.

Figure 2-47. EIGRP Summarization Can Limit Query Range.

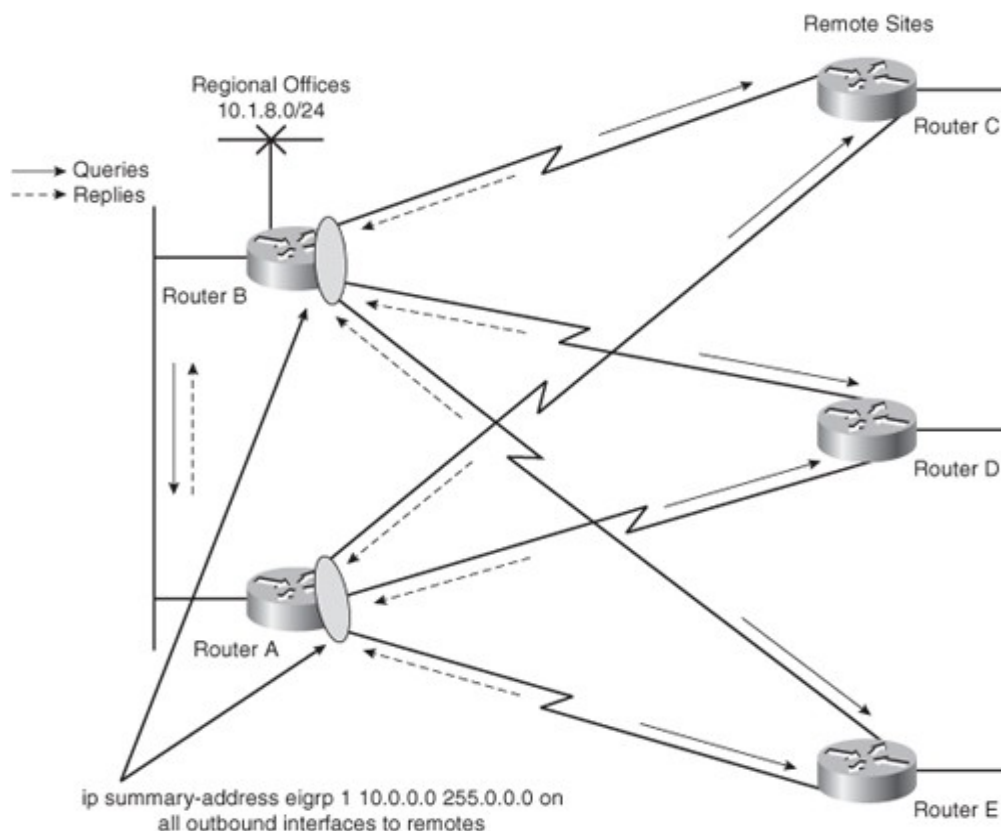


Summarization minimizes the size of the routing table, which means less CPU and memory usage to manage it and less bandwidth to transmit the information. Summarization also reduces the chance of networks becoming SIA because it reduces the number of routers that see each query, so the chance of a query encountering one of these issues is also reduced.

Figure 2-48 illustrates how route summarization can affect the network shown earlier in Figure 2-44. The ip summary-address eigrp command is configured on the outbound interfaces of Routers A and B so that Routers A and B advertise the 10.0.0.0/8 summary route to the remote Routers C, D, and E.

Figure 2-48. Limiting Updates and Queries Using Summarization.

[View full size image]



The 10.1.8.0/24 network is not advertised to the remote routers. Therefore, the remote routers (C, D, and E) do not extend the queries about the 10.1.8.0/24 network back to the other regional routers, reducing the convergence traffic (queries and replies) caused by the redundant topology. When Routers A and B send the query for 10.1.8.0/24 to Routers C, D, and E, these routers immediately reply that the destination is unreachable. Queries for the lost 10.1.8.0/24 networks are not propagated beyond the remote sites, preventing Routers A and B from becoming SIA waiting for the query process to receive all the replies.

Limiting Query Range Using a Stub

Hub-and-spoke network topologies commonly use stub routing. In this topology, the remote router forwards all traffic that is not local to a hub router, so the remote router does not need to retain a complete routing table. Generally, the hub router needs to send only a default route to the remote routers.

In a hub-and-spoke topology, having a full routing table on the remote routers serves no functional purpose because the path to the corporate network and the Internet is always through the hub router. In addition, having a full routing table at the spoke routers increases the amount of memory required.

Traffic from a hub router should not use a remote router as a transit path. A typical connection from a hub router to a remote router has significantly less bandwidth than a connection at the network core. Attempting to use the connection to a remote router as a transit path typically results in excessive congestion. The EIGRP stub routing feature can prevent this problem by restricting the remote router from advertising the hub router's routes back to other hub routers. For example, in Figure 2-48, routes recognized by the remote router from hub Router A are not advertised to hub Router B. Because the remote router does not advertise the hub routes back to the hub routers, the hub routers do not use the remote routers as a transit path. Using the EIGRP stub routing feature improves network stability, reduces resource utilization, and simplifies stub router configuration.

The EIGRP stub feature was first introduced in Cisco IOS Release 12.0(7)T.

Only the remote routers are configured as stubs. The stub feature does not prevent routes from being advertised to the remote router.

A stub router indicates in the hello packet to all neighboring routers its status as a stub router. Any router that receives a packet informing it of its neighbor's stub status does not query the stub router for any routes. Therefore, a router that has a stub peer does not query that peer.

Thus, it is important to note that stub routers are not queried. Instead, hub routers connected to the stub router answer the query on behalf of the stub router.

The EIGRP stub routing feature also simplifies the configuration and maintenance of hub-and-spoke networks, improves network stability, and reduces resource utilization. When stub routing is enabled in dual-homed remote configurations, you do not have to configure filtering on remote routers to prevent them from appearing as transit paths to the hub routers.

Caution

EIGRP stub routing should be used on stub routers only. A stub router is defined as a router connected to the network core or hub layer, and through which core transit traffic should not flow. A stub router should have only hub routers for EIGRP neighbors. Ignoring this restriction causes undesirable behavior.

When planning for EIGRP stub configuration, complete the following steps:

- Examine the topology and the existing EIGRP configuration.
- Define requirements, including which routers will be configured as stub routers, and whether redistribution and summarization will be deployed.
- Design the network.
- Create an implementation plan.
- Configure and verify the configuration.

To configure a router as an EIGRP stub, use the `eigrp stub [receive-only | connected | static | summary | redistributed]` router configuration command. A router configured as a stub with the `eigrp stub` command shares information about connected and summary routes with all neighbor routers by default. Table 2-13 describes the four optional keywords that can be used with the `eigrp stub` command to modify this behavior.

Table 2-13. *eigrp stub* Command Parameters

Parameter	Description
receive-only	The receive-only keyword restricts the router from sharing any of its routes with any other router within an EIGRP autonomous system. This keyword does not permit any other keyword to be specified, because it prevents any type of route from being sent. Use this option if there is a single interface on the router.
connected	The connected keyword permits the EIGRP stub routing feature to send connected routes. If a network command does not include the connected routes, it might be necessary to redistribute connected routes with the redistribute connected command under the EIGRP process. This option is enabled by default and is the most widely practical stub option.
static	The static keyword permits the EIGRP stub routing feature to send static routes. Redistributing static routes with the <code>redistribute static</code> command is still necessary.
summary	The summary keyword permits the EIGRP stub routing feature to send summary routes. You can create summary routes manually with the ip summary-address eigrp command or automatically at a major network border router with the <code>auto-summary</code> command enabled. This option is enabled by default.
redistributed	The redistribute option permits the EIGRP stub routing feature to send redistributed routes. Redistributing routes with the redistribute command is still necessary.

The optional parameters in this command can be used in any combination, with the exception of the **receive-only** keyword. If any of the keywords (except **receive-only**) is used individually, the connected and summary routes are not sent automatically.

In Example 2-59, the `eigrp stub` command is used to configure the router as a stub that advertises connected and summary routes.

Example 2-59. *eigrp stub* Command to Advertise Connected and Summary Routes

```
Router(config)#router eigrp 1  
Router(config-router)#network 10.0.0.0  
Router(config-router)#eigrp stub
```

In Example 2-60, the `eigrp stub receive-only` command is used to configure the router as a stub for which connected, summary, and static routes are not sent.

Example 2-60. *eigrp stub* Command to Receive Only Routes

```
Router(config)#router eigrp 1  
Router(config-router)#network 10.0.0.0 eigrp  
Router(config-router)#eigrp stub receive-only
```

The EIGRP stub feature does not enable route summarization on the hub router. The network administrator should configure route summarization on the hub routers if desired. Cisco highly recommends using both EIGRP route summarization and EIGRP stub features to provide the best scalability.

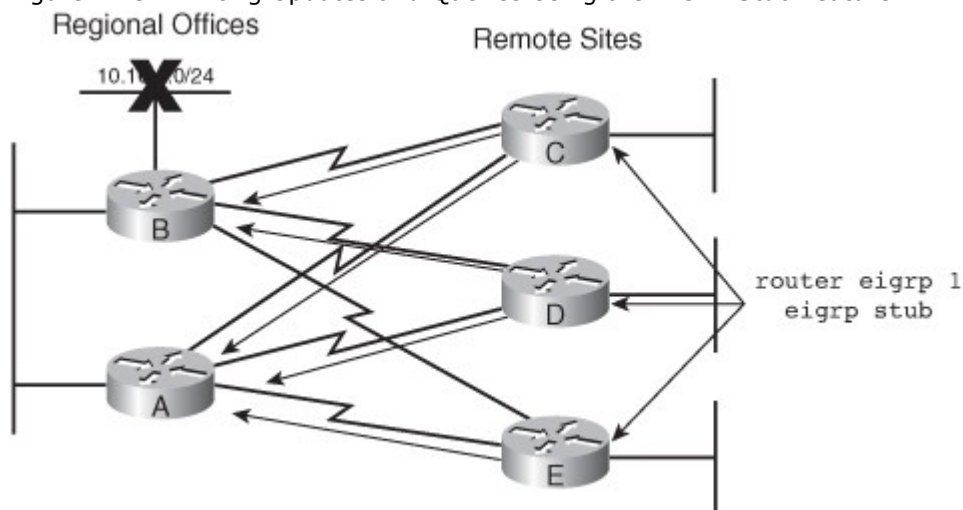
If a true stub network is required, the hub router can be configured to send a default route to the spoke routers. This approach is the simplest and conserves the most bandwidth and memory on the spoke routers.

Note

Although EIGRP is a classless routing protocol, it has classful behavior by default, such as having automatic summarization on by default. When you configure the hub router to send a default route to the remote router, ensure that the **ip classless** command on the remote router. By default, the **ip classless** command is enabled in all Cisco IOS images that support the EIGRP stub routing feature.

Figure 2-49 illustrates how using the EIGRP stub feature affects the network shown earlier in Figure 2-44. Each of the remote routers is configured as a stub. Queries for network 10.1.8.0/24 are not sent to Routers C, D, or E, thus reducing the bandwidth used and the chance of the routes being SIA.

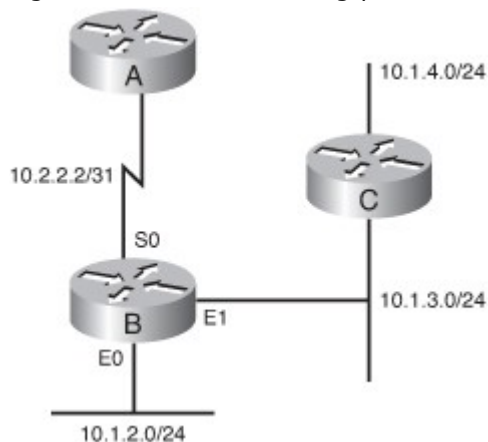
Figure 2-49. Limiting Updates and Queries Using the EIGRP Stub Feature.



Using the EIGRP stub feature at the remote sites allows the hub (regional offices) sites to immediately answer queries without propagating the queries to the remote sites, saving CPU cycles and bandwidth, and lessening convergence time even when the remote sites are dual-homed to two or more hub sites.

Figure 2-50 illustrates another example network; Example 2-61 shows part of the configuration for Router B.

Figure 2-50. Network for *eigrp stub* Command Example.



Example 2-61. Configuration for Router B in Figure 2-50

```

RouterB#show running-config
<output omitted>
ip route 10.1.4.0 255.255.255.0 10.1.3.10
!
interface ethernet 0
ip address 10.1.2.1 255.255.255.0
!
interface ethernet 1
ip address 10.1.3.1 255.255.255.0
!
interface serial 0
ip address 10.2.2.3 255.255.255.254
ip summary-address eigrp 100 10.1.2.0 255.255.254.0
!
router eigrp 100
 redistribute static 1000 1 255 1 1500
 network 10.2.2.2 0.0.0.1
 network 10.1.2.0 0.0.0.255
<output omitted>
  
```

Using this example network and configuration, consider which networks Router B will advertise to Router A when the various options of the **eigrp stub** command are also configured on Router B:

- With the **eigrp stub connected** command, Router B will advertise only 10.1.2.0/24 to Router A. Notice that although 10.1.3.0/24 is also a connected network, it is not advertised to Router A because it is not advertised in a **network** command, and connected routes are not redistributed.
- With the **eigrp stub summary** command, Router B will advertise only 10.1.2.0/23, the summary route that is configured on the router, to Router A.
- With the **eigrp stub static** command, Router B will advertise only 10.1.4.0/24, the static route that is configured on the router, to Router A. (Note that the **redistribute static** command is configured on Router B.)

Note

Without the **static** option, EIGRP will not send any static routes, including internal static routes that normally would be automatically redistributed.

- With the **eigrp stub receive-only** command, Router B will not advertise anything to Router A.

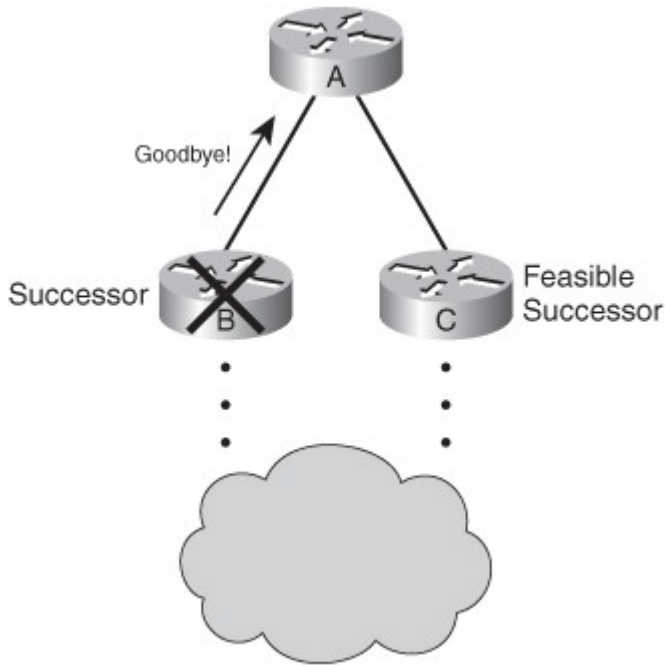
- With the **eigrp stub redistributed** command, Router B will advertise only 10.1.4.0/24, the redistributed static route, to Router A.

Graceful Shutdown

Graceful shutdown, implemented with the *goodbye message* feature, is designed to improve EIGRP network convergence.

In Figure 2-51, Router A is using Router B as the successor for several routes. Router C is the feasible successor for the same routes. Router B normally would not tell Router A if the EIGRP process on Router B was going down (for example, if Router B was being reconfigured). Router A would have to wait for its hold timer to expire before it would discover the change and react to it. Packets sent during this time would be lost.

Figure 2-51. Graceful Shutdown Causes the Router to Say Goodbye.



With graceful shutdown, a goodbye message is broadcast when an EIGRP routing process is shut down, to inform adjacent peers about the impending topology change. This feature allows supporting EIGRP peers to synchronize and recalculate neighbor relationships more efficiently than would occur if the peers discovered the topology change after the hold timer expired.

The goodbye message is supported in Cisco IOS Software Release 12.3(2), 12.3(3)B, and 12.3(2)T and later. Interestingly, goodbye messages are sent in hello packets. EIGRP sends an interface goodbye message with all K values set to 255 when taking down all peers on an interface.

The following message is displayed by routers that support goodbye messages when one is received:

```
*Apr 26 13:48:42.523: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1
(Ethernet0/0) is down: Interface Goodbye received
```

A Cisco router that runs a software release that does not support the goodbye message will misinterpret the message as a K value mismatch and therefore display the following message:

```
*Apr 26 13:48:41.811: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1
(Ethernet0/0) is down: K-value mismatch
```

Note

The receipt of a goodbye message by a nonsupporting peer does not disrupt normal network operation. The nonsupporting peer will terminate the session when the hold timer expires. The sending and receiving routers will reconverge normally after the sender reloads.

Note

An EIGRP router will send a goodbye message on an interface if the **network** command (under the EIGRP process) that encompasses the network on that interface is removed (with the **no network** command). An EIGRP router sends a goodbye message on all interfaces if the EIGRP process is shut down (with the **no router eigrp** command). An EIGRP router will not, however, send a goodbye message if an interface is shut down or the router is reloaded.

Summary

In this chapter, you learned about Cisco's EIGRP, an advanced distance vector routing protocol. The chapter focused on the following topics:

- Features of EIGRP, including fast convergence, use of partial updates, multiple network layer support, use of multicast and unicast, VLSM support, seamless connectivity across all data link layer protocols and topologies, and sophisticated metrics.
- EIGRP's underlying processes and technologies—neighbor discovery/recovery mechanism, RTP, DUAL finite-state machine, and protocol-dependent modules.
- EIGRP's tables—neighbor table (containing all routers with which it has a neighbor relationship), topology table (containing all routes from all neighbors), and routing table (containing the best routes to each destination and used for forwarding packets).
- EIGRP terminology
 - The advertised distance (the metric for an EIGRP *neighbor router* to reach the destination, which is the metric between the next-hop router and the destination), the feasible distance (the sum of the AD from the next-hop neighbor plus the cost between the local router and the next-hop router), the successor (a neighboring router that has a least-cost loop-free path to a destination, the lowest FD), and the feasible successor (a neighboring router that has a loop-free backup path to a destination).
 - Passive routes, those not undergoing recomputation; active routes, those undergoing recomputation. Passive is the operational, stable state.
- The five EIGRP packet types: hello, update, query, reply, and acknowledgment. Updates, queries, and replies are sent reliably.
- EIGRP initial route discovery process, started by a router sending hello packets. Neighboring routers reply with update packets, which populate the router's topology table. The router chooses the successor routes and offers them to the routing table.
- The DUAL process including selecting FSs. To qualify as an FS, a next-hop router must have an AD less than the FD of the current successor route for the particular network, to ensure a loop-free network.
- The EIGRP metric calculation, which defaults to bandwidth (the slowest bandwidth between the source and destination) plus delay (the cumulative interface delay along the path).
- Planning EIGRP implementations, including the IP addressing, network topology, and EIGRP traffic engineering. The list of tasks for each router in the network includes enabling the EIGRP routing protocol (with the correct autonomous system number), configuring the proper network statements, and optionally configuring the metric to appropriate interfaces.
- Basic EIGRP configuration commands
 - **router eigrp** *autonomous-system-number* global configuration command
 - **network** *network-number* [*wildcard-mask*] interface configuration command
 - **bandwidth** *kilobits* interface configuration command
- Commands for verifying EIGRP operation
 - show ip eigrp neighbors
 - show ip route
 - show ip route eigrp
 - show ip protocols
 - show ip eigrp interfaces
 - show ip eigrp topology
 - show ip eigrp traffic
 - debug eigrp packets
 - debug ip eigrp
 - debug ip eigrp summary

- debug eigrp neighbors
- Using the **passive-interface** *type number* [**default**] router configuration command to prevent a routing protocol's routing updates from being sent through the specified router interface. With EIGRP, hello messages are not sent out of the specified interface, so neighborships are not formed on that interface. However, the router will still advertise the network to its EIGRP neighbors.
- Creating a statically configured default route with the **ip route** *0.0.0.0 0.0.0.0 next-hop | interface* global configuration command. Alternatively, any major network residing in the local routing table can become an EIGRP default route when used in the **ip default-network** *network-number* global configuration command.
- EIGRP summarization: EIGRP automatically summarizes on the major network boundary by default (in some IOS versions). To turn off automatic summarization, use the **no auto-summary** router configuration command. Use the **ip summary-address eigrp** *as-number address mask [admin-distance]* interface configuration command to manually create a summary route at an arbitrary bit boundary, as long as a more specific route exists in the routing table.
- EIGRP over Frame Relay, including
 - An overview of Frame Relay, where PVCs are created by an SP and are identified by DLCIs, locally significant numbers between the router and the Frame Relay switch.
 - EIGRP over a physical interface using Inverse ARP dynamic mapping, the default
 - EIGRP over the physical interface using static maps with the **frame-relay map** *protocol protocol-address dlci* [**broadcast**] [**ietf | cisco**] [**payload-compress {packet-by-packet | frf9 stac}**] interface configuration command.
 - Configuring subinterfaces with the **interface serial** *number.subinterface-number* {**point-to-point | multipoint**} command.
 - Frame Relay multipoint subinterfaces, on which the IP address-to-DLCI mapping is done by either specifying the local DLCI value (using the **frame-relay interface-dlci** *dlci* command) and relying on Inverse ARP, or using manual IP address-to-DLCI mapping.
 - Frame Relay point-to-point subinterfaces, on which the IP address-to-DLCI mapping is done by specifying the local DLCI value.
 - Split horizon, which is disabled by default on most interfaces. Split horizon is enabled by default on Frame Relay multipoint interfaces. It can be disabled with the **no ip split-horizon eigrp** *as-number* command.
 - The **neighbor** {*ip-address | ipv6-address*} *interface-type interface-number* router configuration command, used to define a neighboring EIGRP router. Instead of using multicast packets, EIGRP exchanges routing information with the specified neighbor using unicast packets.
- EIGRP over MPLS, including
 - MPLS, which uses labels assigned to each packet at the edge of the network; MPLS nodes use the label to determine where to send the packet.
 - A Layer 2 MPLS VPN, which provides a Layer 2 service across the backbone, where customer routers are connected together on the same IP subnet. A type of ATOM, called EoMPLS may be used. Customer routers may also be connected to MPLS edge routers via VLAN subinterfaces; different subinterfaces in the PE routers are used to connect to different VLANs.
 - A Layer 3 MPLS VPN, which provides a Layer 3 service across the backbone, where customer routers are connected to ISP edge routers, and on each side, a separate IP subnet is used. The CE routers see the PE routers as another customer router in the path. PE routers maintain separate routing tables for each customer.
- EIGRP load-balancing, including
 - EIGRP equal-cost load balancing that distributes traffic over all interfaces with the same metric for the destination address.
 - The **maximum-paths** *maximum-path* router configuration command, to specify up to 16 equally good routes be kept in the routing table. Set *maximum-path* to 1 to disable load balancing.
 - EIGRP unequal-cost load balancing, across routes with different metrics, using the **variance** *multiplier* router configuration command and the **traffic-share** [**balanced | min across-interfaces**] router configuration command.
- EIGRP operation in WAN environments

- By default, EIGRP uses up to 50 percent of the bandwidth declared on an interface or subinterface. EIGRP uses the bandwidth of the link set by the **bandwidth** command, or the link's default bandwidth if none is configured, when calculating how much bandwidth to use. The **ip bandwidth-percent eigrp as-number percent** interface configuration command changes the percentage.
- Configuring multipoint interfaces: Configure the bandwidth to represent the minimum CIR times the number of circuits. For a small number of very low-speed circuits, define the interfaces as point-to-point so that their bandwidth can be set to match the provisioned CIR.
- Configuring, verifying, and troubleshooting EIGRP MD5 authentication, including the following commands
 - **ip authentication mode eigrp autonomous-system md5** interface configuration command
 - **key chain name-of-chain** global configuration command
 - **key key-id** key-chain configuration command
 - **key-string key** key-chain-key configuration command
 - **accept-lifetime start-time {infinite | end-time | duration seconds}** key-chain-key configuration command
 - **send-lifetime start-time {infinite | end-time | duration seconds}** key-chain-key configuration command
 - **ip authentication key-chain eigrp autonomous-system name-of-chain** interface configuration command
 - **show ip eigrp neighbors**
 - **show ip route**
 - **show key chain**
 - **debug eigrp packets**
- EIGRP scalability factors, including the amount of information exchanged, the number of routers, the depth of the topology, and the number of alternative paths through the network.
- The SIA state: If an EIGRP router does not receive a reply to all the outstanding queries within 3 minutes (the default time) it goes into SIA state, and the router resets the neighbor relationships for the neighbors that failed to reply. Reasons for going into SIA include: router is too busy, the link between the routers is not good, or the link has a unidirectional failure.
- The Active Process Enhancement feature that adds SIA-Query and SIA-Reply to ensure that only appropriate neighbor relationships are reset.
- Limiting the query range to help reduce SIAs, which can be accomplished by
 - Only announcing default routes to remote routers, which are configured to announce only their directly connected networks.
 - Configuring route summarization using the **ip summary-address eigrp** command on the outbound interfaces of the appropriate routers
 - Route and interface packet filtering.
 - Configuring the remote routers as stub EIGRP routers so that they will not be queried, using the **eigrp stub [receive-only | connected | static | summary | redistributed]** router configuration command.
 - Using both EIGRP route summarization and EIGRP stub features is recommended, to provide the best scalability.
- Graceful shutdown, which broadcasts a goodbye message (in a hello packet, with all K values set to 255) when an EIGRP routing process is shut down, to inform neighbors.

References

For additional information, see the following resources:

- The Cisco EIGRP protocol home page: <http://www.cisco.com/go/eigrp>
- The Cisco IGRP Metric document: http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a008009405c.shtml (a good reference for the "EIGRP Metric Calculation" section)
- The Cisco IOS IP Routing Protocols Command Reference: http://www.cisco.com/en/US/docs/ios/iproute/command/reference/irp_book.html

Review Questions

Answer the following questions and then see Appendix A, "Answers to Review Questions," for the answers.

1. What are some features of EIGRP?
2. Is EIGRP operational traffic multicast or broadcast?
3. What are the four key technologies employed by EIGRP?
4. Which of the following best describes the EIGRP topology table?
 - a. It is populated as a result of receiving hello packets.
 - b. It contains all learned routes to a destination.
 - c. It contains only the best routes to a destination.
5. Describe the five types of EIGRP packets.
6. How often are EIGRP hello packets sent on LAN links?
7. What is the difference between the hold time and the hello interval?
8. Which of the following statements are true? (Choose three.)
 - a. A route is considered passive when the router is not performing recomputation on that route.
 - b. A route is passive when it is undergoing recomputation.
 - c. A route is active when it is undergoing recomputation.
 - d. A route is considered active when the router is not performing recomputation on that route.
 - e. Passive is the operational state for a route.
 - f. Active is the operational state for a route.
9. Which command is used to see the RTO and hold time?
 - a. show ip eigrp traffic
 - b. show ip eigrp timers
 - c. show ip eigrp route
 - d. show ip eigrp neighbors
10. Why are EIGRP routing updates described as reliable?
11. Which of the following statements are true about advertised distance (AD) and feasible distance (FD)? (Choose two.)
 - a. The AD is the EIGRP metric for a *neighbor router* to reach a particular network.
 - b. The AD is the EIGRP metric for *this router* to reach a particular network.
 - c. The FD is the EIGRP metric for *this router* to reach a particular network.
 - d. The FD is the EIGRP metric for *the neighbor router* to reach a particular network.
12. What does it mean when a route is marked as an FS?
13. In the following table, place the letter of the description next to the term the description describes. The descriptions may be used more than once.

Descriptions:

- a. A network protocol that EIGRP supports.
- b. A table that contains FS information.
- c. The administrative distance determines routing information that is included in this table.
- d. A neighbor router that has the best path to a destination.
- e. A neighbor router that has a loop-free alternative path to a destination.
- f. An algorithm used by EIGRP that ensures fast convergence.
- g. A multicast packet used to discover neighbors.
- h. A packet sent by EIGRP routers when a new neighbor is discovered and when a change occurs.

Term	Description Letter
Successor	
Feasible successor	
Hello	
Topology table	
IP	

Term	Description Letter
Update	
IPv6	
Routing table	
DUAL	
IPX	

14. The following is part of the output of the **show ip eigrp topology** command:
P 10.1.3.0/24, 1 successors, FD is 10514432
 via 10.1.2.2 (10514432/28160), Serial0/0/0
- What are the two numbers in parentheses?
15. Answer true or false to the following statements.
EIGRP performs autosummarization.
EIGRP autosummarization cannot be turned off.
EIGRP supports VLSM.
EIGRP can maintain independent routing tables.
The EIGRP hello interval is an unchangeable fixed value.
16. How do IGRP and EIGRP differ in their metric calculation?
17. What units are the bandwidth and delay parameters in the EIGRP metric calculation?
18. What are some of the tasks that should be listed in the implementation plan for EIGRP?
19. Router A has three interfaces with IP addresses 172.16.1.1/24, 172.16.2.3/24, and 172.16.5.1/24. What commands would be used to configure EIGRP to run in autonomous system 100 on only the interfaces with addresses 172.16.2.3/24 and 172.16.5.1/24?
20. What does the **passive-interface** command do when configured with EIGRP?
21. Router R1 is configured with the **ip default-network 172.17.0.0** command, and the **network 172.17.0.0** command under the EIGRP process. Router R2 is an EIGRP neighbor of R1's and learns about the 172.17.0.0 network from R1. How are the routing tables on both routers affected by the **ip default-network 172.17.0.0** command?
22. Routers A and B are connected and are running EIGRP on all their interfaces. Router A has four interfaces, with IP addresses 172.16.1.1/24, 172.16.2.3/24, 172.16.5.1/24, and 10.1.1.1/24. Router B has two interfaces, with IP addresses 172.16.1.2/24 and 192.168.1.1/24. There are other routers in the network that are connected on each of the interfaces of these two routers that are also running EIGRP. Which summary routes does Router A generate automatically (assuming autosummarization is enabled)? (Choose two.)
- 172.16.0.0/16
 - 192.168.1.0/24
 - 10.0.0.0/8
 - 172.16.1.0/22
 - 10.1.1.0/24
23. Which of the following are true?
- For Frame Relay point-to-point interfaces, set the **bandwidth** to the CIR.
 - For Frame Relay point-to-point interfaces set the **bandwidth** to the sum of all CIRs.
 - For Frame Relay multipoint connections, set the **bandwidth** to the sum of all CIRs.
 - For generic serial interfaces such as PPP and HDLC, set the **bandwidth** to match the line speed.
 - For Frame Relay multipoint connections, set the **bandwidth** to the CIR.
24. Router R1 with IP address 172.16.1.1 can reach Router R2 with IP address 172.16.1.2 over a Frame Relay connection. The DLCI of the connection on R1 is 100 and the DLCI of the connection on R2 is 200. Write the command that configures a static map on Router R1 so that Router R1 can send EIGRP updates to Router R2.

25. On which type of interfaces is split horizon enabled for EIGRP by default? On which type of interfaces is split horizon disabled for EIGRP by default?
26. What does the EIGRP **neighbor** command do?
27. Describe the differences between a Layer 2 MPLS VPN and a Layer 3 MPLS VPN.
28. You are deploying EIGRP over EoMPLS. Your main office has Router R1, connected to a provider edge Router PE1. A branch office has Router R2, connected to provider edge Router PE2. Between which routers is an EIGRP neighbor relationship established?
29. Router A has four EIGRP paths to a destination with the following EIGRP metrics. Assuming no potential routing loops exist and the command **variance 3** is configured on Router A, which paths are included for load balancing?
- a. Path 1: 1100
 - b. Path 2: 1200
 - c. Path 3: 2000
 - d. Path 4: 4000
30. Router A has the following configuration:
- ```
interface s0
 ip bandwidth-percent eigrp 100 40
 bandwidth 256
 router eigrp 100
 network 10.0.0.0
```

What is the maximum bandwidth (in kbps) that EIGRP uses on the S0 interface?

- a. 100
  - b. 40
  - c. 256
  - d. 102
  - e. 10
  - f. 47
31. What is the default EIGRP authentication?
- a. Simple password
  - b. MD5
  - c. None
  - d. IPsec
32. When configuring EIGRP authentication between two routers on a link, does each router have a unique password?
33. What does the **accept-lifetime** command do for EIGRP authentication?
34. What command is used to troubleshoot EIGRP authentication?
- a. debug eigrp authentication
  - b. debug ip eigrp packets
  - c. debug eigrp packets
  - d. debug ip eigrp authentication
35. What is the default EIGRP stuck-in-active timer?
36. With the EIGRP active process enhancement, when does the SIA-Query get sent?
37. How does EIGRP summarization limit the query range?
38. How does the EIGRP stub feature limit the query range?
39. What does the **eigrp stub receive-only** command do?
40. In which EIGRP packet type are goodbye messages sent?.

This chapter introduces the Open Shortest Path First (OSPF) routing protocol. It covers the following topics:

- Understanding OSPF Terminology and Operation
- OSPF Packets
- Configuring and Verifying Basic OSPF Routing
- Understanding OSPF Network Types
- Understanding OSPF LSAs
- Interpreting the OSPF LSDB and Routing Table
- Configuring and Verifying Advanced OSPF Features
- Configuring and Verifying OSPF Authentication

This chapter examines the Open Shortest Path First (OSPF) routing protocol, which is one of the most commonly used interior gateway protocols in IP networking. OSPF is an open-standard protocol based primarily on RFC 2328. OSPF is a fairly complex protocol made up of several protocol handshakes, database advertisements, and packet types. This chapter describes basic configuration and verification of OSPF, in both single and multiple areas, and explores OSPF configuration over specific network types. The chapter also covers configuration and verification of advanced OSPF features, including passive interface, default routes, summarization, virtual links, changing the cost metric, and special area types. The chapter concludes with a discussion of OSPF authentication configuration and verification.

### Understanding OSPF Terminology and Operation

This section introduces the major characteristics of the OSPF routing protocol, including a description of link-state routing protocols, area structures, link-state adjacencies, shortest path first (SPF) metric calculations, and link-state data structures.

### Link-State Routing Protocols

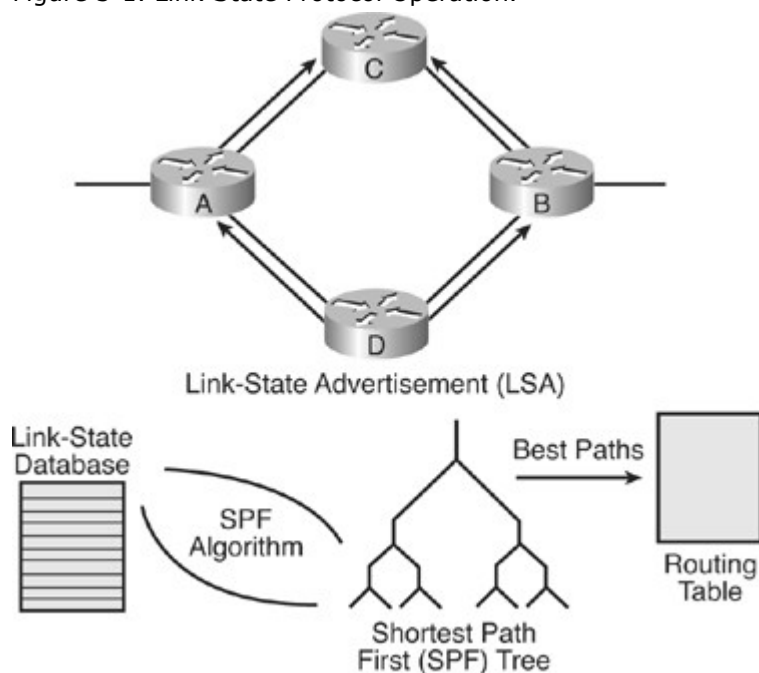
The need to overcome the limitations of distance vector routing protocols led to the development of link-state routing protocols. Link-state routing protocols have the following characteristics:

- They respond quickly to network changes.
- They send triggered updates when a network change occurs.
- They send periodic updates, known as link-state refresh, at long time intervals, such as every 30 minutes.

Link-state routing protocols generate routing updates only when a change occurs in the network topology. When a link changes state, the device that detected the change creates a link-state advertisement (LSA) concerning that link, as shown in Figure 3-1. The LSA propagates to all neighboring devices using a special multicast address. Each routing device stores the LSA, forwards the LSA to all neighboring devices (within an area; areas are described in the next section, "OSPF Area Structure") and updates its link-state database (LSDB). This flooding of the LSA ensures that all routing devices can update their databases and then update their routing tables to reflect the new topology. As shown in Figure 3-1, the LSDB is used to calculate the best paths through the network. Link-state routers find the best paths to a destination by applying Dijkstra's algorithm, also known as SPF, against the LSDB to build the SPF tree. Each router selects the best paths from their SPF tree and places them in their routing table.



Figure 3-1. Link-State Protocol Operation.



#### Link-State Routing Analogy

You can think of the LSDB as being like a map in a shopping mall—every map in the mall is the same, just as the LSDB is the same in all routers within an area.

The one difference between all the maps in a shopping mall is the “you are here” dot. By looking at this dot, you can determine the best way to get to every store from your current location. The best path to a specific store will be different from each location in the mall. Link-state routers function similarly—they each calculate the best way to every network within the area, from their own perspective, using the LSDB.

OSPF and Integrated Intermediate System-to-Intermediate System (IS-IS) are classified as link-state routing protocols because of the manner in which they distribute routing information and calculate routes.

Routers running link-state routing protocols collect routing information from all other routers in the network (or from within a defined area of the network), and then each router independently calculates its best paths to all destinations in the network, using Dijkstra’s (SPF) algorithm. Incorrect information from any particular router is less likely to cause confusion because each router maintains its own view of the network.

For all the routers in the network to make consistent routing decisions, each link-state router must keep a record of the following information:

- **Its immediate neighbor routers**— If the router loses contact with a neighbor router, within a few seconds it invalidates all paths through that router and recalculates its paths through the network. For OSPF, adjacency information about neighbors is stored in the OSPF neighbor table, also known as an adjacency database.
- **All the other routers in the network, or in its area of the network, and their attached networks**— The router recognizes other routers and networks through LSAs, which are flooded through the network. LSAs are stored in a topology table or database (which is also called an LSDB).
- **The best paths to each destination**— Each router independently calculates the best paths to each destination in the network using Dijkstra’s (SPF) algorithm. All paths are kept in the LSDB. The best paths are then offered to the routing table (also called the forwarding database). Packets arriving at the router are forwarded based on the information held in the routing table.

#### Note

You might encounter different terminology for the various OSPF tables, as follows:

- OSPF neighbor table = adjacency database
- OSPF topology table = OSPF topology database = LSDB

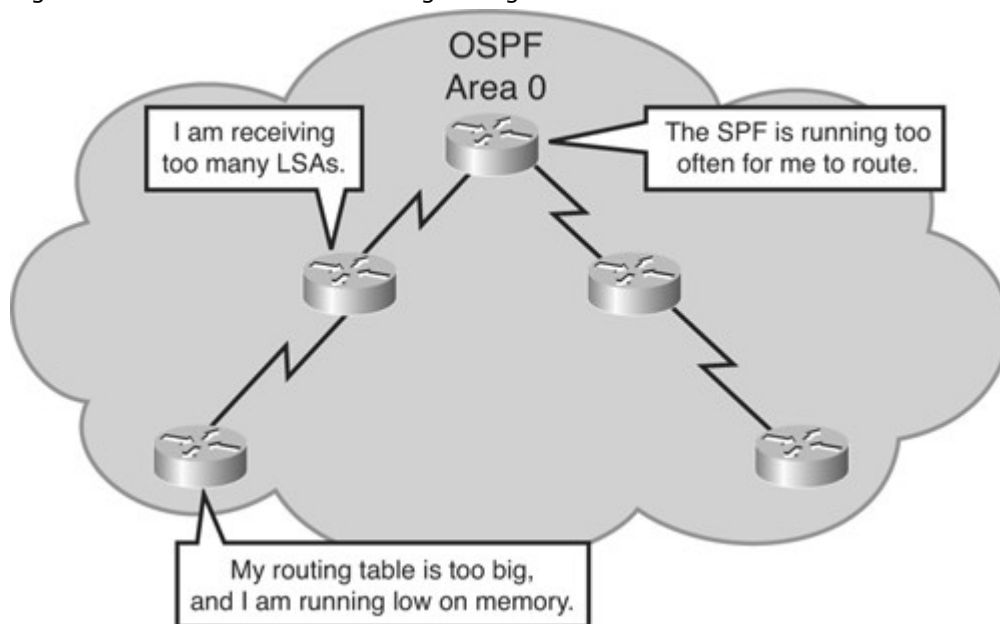
- Routing table = forwarding database

The memory resources required to maintain these tables can be a drawback to link-state protocols. However, because the topology table (LSDB) is identical in all OSPF routers in an area and contains full information about all the routers and links in an area, each router can independently select a loop-free and efficient path, based on cost (as described in the “OSPF Metric Calculation” section, later in this chapter), to reach every network in the area. This benefit overcomes the “routing by rumor” limitations of distance vector routing. Recall that routers running distance vector routing protocols rely on routing decisions from their neighbors. Individual routers do not have the full picture of the network topology. In comparison, each router running a link-state routing protocol has the full picture of the network topology and can independently make a decision based on an accurate picture of the network topology.

#### OSPF Area Structure

In small networks, the web of router links is not complex, and paths to individual destinations are easily deduced. However, in large networks, the resulting web is highly complex, and the number of potential paths to each destination is large. Therefore, the Dijkstra calculations comparing all of these possible routes can be very complex and can take significant time, resulting in issues such as those shown in Figure 3-2.

Figure 3-2. Issues with Maintaining a Large OSPF Network.



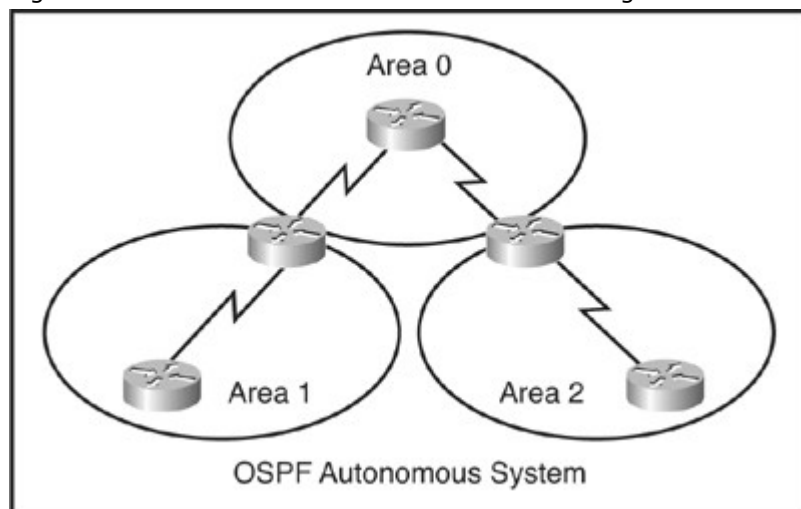
If an area becomes too big, the following issues need to be addressed:

- **Frequent SPF algorithm calculations**— In a large network, changes are inevitable, so the routers spend many CPU cycles recalculating the SPF algorithm and updating the routing table.
- **Large routing table**— OSPF does not perform route summarization by default. If the routes are not summarized, the routing table can become very large, depending on the size of the network.
- **Large LSDB**— Because the LSDB covers the topology of the entire network, each router must maintain an entry for every network in the area, even if not every route is selected for the routing table.

Link-state routing protocols usually reduce the size of the Dijkstra calculations by partitioning the network into areas, as shown in Figure 3-3. The number of routers in an area and the number of LSAs that flood only within the area are smaller, which means that the LSDB or topology database for an area is small.

Consequently, the Dijkstra calculation is easier and takes less time. Interarea routing still occurs, but many of the internal routing operations, such as SPF calculations, can remain within individual areas. For example, if area 1 is having problems with a link going up and down, routers in other areas do not need to continually run their SPF calculation, because they can be isolated from the problem in area 1.

Figure 3-3. The Solution: OSPF Hierarchical Routing.



Assuming a proper IP addressing hierarchy and proper OSPF configuration is in place, using multiple OSPF areas has several important advantages:

- **Reduced frequency of SPF calculations**— Because detailed route information exists within each area, it is not necessary to flood all link-state changes to all other areas. Therefore, only routers that are affected by the change need to recalculate the SPF algorithm and the impact of the change is localized within the area.
- **Smaller routing tables**— With multiple areas, detailed route entries for specific networks within an area can remain in the area. Instead of advertising these explicit routes outside the area, routers can be configured to summarize the routes into one or more summary addresses. Advertising these summaries reduces the number of LSAs propagated between areas but keeps all networks reachable.
- **Reduced LSU overhead**— LSUs contain a variety of LSA types, including link-state and summary information. Rather than send an LSU about each network within an area, a router can advertise a single summarized route or a small number of routes between areas, thereby reducing the overhead associated with LSUs when they cross areas.

OSPF uses a two-layer area hierarchy:

- **Backbone area**— An OSPF area whose primary function is the fast and efficient movement of IP packets. Backbone areas interconnect with other OSPF area types. Generally, end users are not found within a backbone area. The backbone area is also called OSPF area 0. Hierarchical networking defines area 0 as the core to which all other areas directly connect.

Note

In some Cisco documentation, you will see the backbone area referred to as a transit area. However, in the OSPF RFC, a transit area is related to virtual links, as described in detail in the "OSPF Virtual Links" section, later in this chapter.

- **Regular (nonbackbone) area**— An OSPF area whose primary function is to connect users and resources. Regular areas (also called normal areas) are usually set up along functional or geographic groupings. By default, a regular area does not allow traffic from another area to use its links to reach other areas. By default, all traffic from other areas must cross backbone area 0. Regular areas can have several subtypes, including standard area, stub area, totally stubby area, not-so-stubby area (NSSA), and totally stubby NSSA. The "Configuring OSPF Special Area Types" section, later in this chapter, discusses area types in further detail.

Note

"Totally stubby NSSA" are also sometimes referred to as "totally NSSA" or "totally stub NSSA." Consider all of these terms to be synonyms.

OSPF forces this rigid two-layer area hierarchy. The network's underlying physical connectivity must map to the two-layer area structure, with all nonbackbone areas attaching directly to area 0.

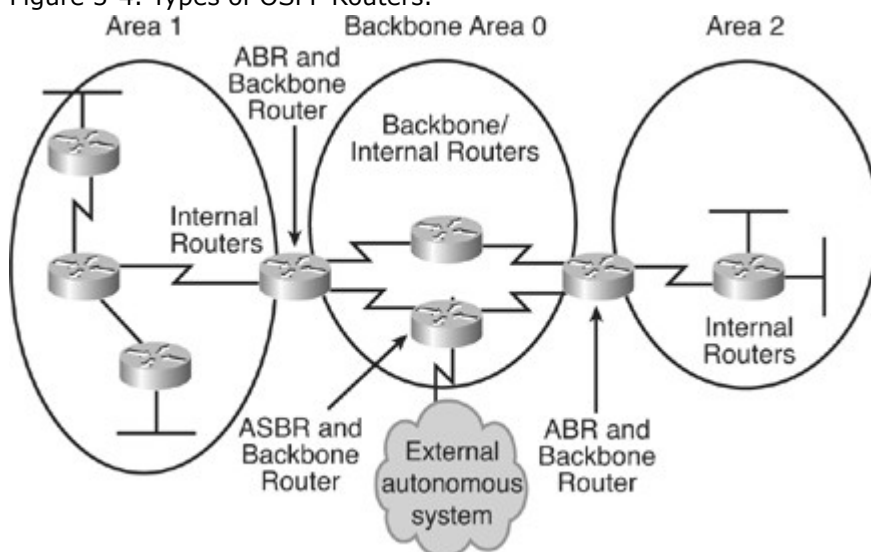
## OSPF Areas

In link-state routing protocols, all routers must keep a copy of the LSDB. The more OSPF routers that exist (and the more links on those routers), the larger the LSDB. It can be advantageous to have all information in all routers, but this approach does not scale to large network sizes. The area concept is a compromise. Routers inside an area maintain detailed information about the links and routers located within that area. OSPF can be configured so that only general or summary information about routers and links in other areas is maintained.

When OSPF is configured properly and a router or link fails, that information is flooded along adjacencies to only routers in the local area. Routers outside the area do not receive this information. By maintaining a hierarchical structure and limiting the number of routers in an area, an OSPF autonomous system (AS) can scale to very large sizes.

As mentioned, OSPF areas require a hierarchical structure, meaning that all areas must connect directly to area 0. In Figure 3-4, notice that links between area 1 routers and area 2 routers are not allowed. All interarea traffic must pass through the backbone area 0. The optimal number of routers per area varies based on factors such as network stability, but in the "Designing Large Scale IP Internetworks" document, Cisco recommends the following guidelines:

Figure 3-4. Types of OSPF Routers.



- An area should have no more than 50 routers.
- A router should not be in more than three areas.

These values are recommended to ensure that OSPF calculations do not overwhelm the routers. Of course, the network design and link stability can also affect the load on the routers.

### Note

The "Designing Large Scale IP Internetworks" document is part of the *Internetworking Design Guide* written in the 1990s. There are large networks today in which the design exceeds these guidelines. However, this document is referenced by Cisco and therefore is included here.

## Area Terminology

OSPF routers of different types control the traffic that goes in and out of areas. The following are the four router types, as shown in Figure 3-4:

- **Internal router**— Routers that have all of their interfaces in the same area. All routers within the same area have identical LSDBs.
- **Backbone router**— Routers that sit in the perimeter of the backbone area 0 and that have at least one interface connected to area 0. Backbone routers maintain OSPF routing information using the same procedures and algorithms as internal routers.
- **Area Border Router (ABR)**— Routers that have interfaces attached to multiple areas, maintain separate LSDBs for each area to which they connect, and route traffic destined for or arriving from other areas. ABRs connect area 0 to a nonbackbone area and are exit points for the area, which means that routing information destined for another area can get there only via the ABR of the local

area. ABRs distribute this routing information into the backbone. The backbone routers then forward the information to the other ABRs. ABRs are the only point where area address summarization can be configured (to summarize the routing information from the LSDBs of their attached areas). ABRs separate LSA flooding zones, and may function as the source of default routes. An area can have one or more ABRs.

The ideal design is to have each ABR connected to two areas only, the backbone and another area. As mentioned, the recommended upper limit is three areas.

- **Autonomous System Boundary Router (ASBR)**— Routers that have at least one interface attached to a different routing domain (such as another OSPF autonomous system or a domain using the Enhanced Interior Gateway Protocol [EIGRP]). An OSPF autonomous system consists of all the OSPF areas and the routers within them. ASBRs can redistribute external routes into the OSPF domain and vice versa.

#### Note

Redistribution is defined as the capability of boundary routers connecting different routing domains to exchange and advertise routing information between those routing domains. Redistribution is covered in Chapter 4, “Manipulating Routing Updates.”

A router can be more than one router type. For example, if a router interconnects to area 0 and area 1, and to a non-OSPF network, it is both an ABR and an ASBR.

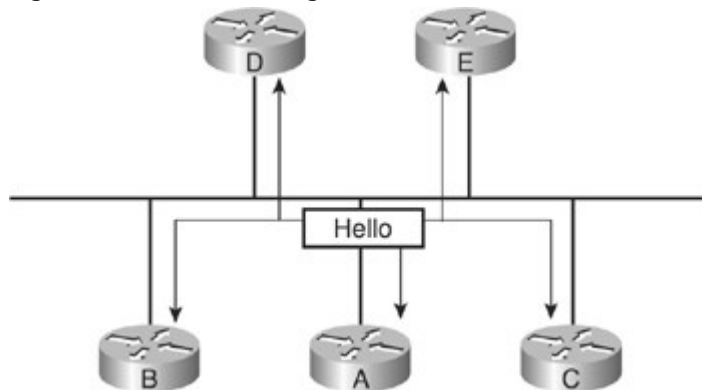
A router has a separate LSDB for each area to which it connects. Therefore, an ABR will have one LSDB for area 0 and another LSDB for the other area in which it participates. Two routers belonging to the same area maintain identical LSDBs for that area.

An LSDB is synchronized between pairs of adjacent routers, as described in the next section.

#### OSPF Adjacencies

A router running a link-state routing protocol must first establish neighbor adjacencies, by exchanging hello packets with the neighboring routers, as shown in Figure 3-5. In general, routers establish adjacencies as follows:

Figure 3-5. Hello Exchange on a Broadcast Network.



- The router sends and receives hello packets to and from its neighboring routers. The destination address is typically a multicast address.
- The routers exchange hello packets subject to protocol-specific parameters, such as checking whether the neighbor is in the same area, using the same hello interval, and so on. Routers declare the neighbor up when the exchange is complete.
- After two routers establish neighbor adjacency using hello packets, they synchronize their LSDBs by exchanging LSAs and confirming the receipt of LSAs from the adjacent router. The two neighbor routers now recognize that they have synchronized their LSDBs with each other. For OSPF, this means that the routers are now in full adjacency state with each other.
- If necessary, the routers forward any new LSAs to other neighboring routers, ensuring complete synchronization of link-state information inside the area.

The two OSPF routers on a point-to-point serial link, usually encapsulated in High-Level Data Link Control (HDLC) or Point-to-Point Protocol (PPP), form a full adjacency with each other.

However, OSPF routers on broadcast networks, such as LAN links, elect one router as the designated router (DR) and another as the backup designated router (BDR). All other routers on the LAN form full adjacencies

with these two routers and pass LSAs only to them. The DR forwards updates received from one neighbor on the LAN to all other neighbors on that same LAN. One of the main functions of a DR is to ensure that all the routers on the same LAN have an identical LSDB. Thus, on broadcast networks, an LSDB is synchronized between a DROTHER (a router that is not a DR or a BDR) and its DR and BDR.

The DR passes its LSDB to any new routers that join that LAN. Having all the routers on that LAN pass the same information to the new router is inefficient, so the one DR router represents the other routers to a new router on the LAN or to other routers in the area.

Routers on the LAN also maintain a partial-neighbor relationship, called a two-way adjacency state, with the other routers on the LAN that are not the DR or BDR, the DROTHERs. For example, if in Figure 3-5, Router A is the DR and Router B is the BDR, Router C will form a full adjacency with both Routers A and B, and a two-way adjacency with Routers D and E.

It is important to note that the DR concept is at the link level. In a multiaccess broadcast environment, each network segment has its own DR and BDR. For example, a router connected to multiple multiaccess broadcast networks can be a DR on one segment and a DROTHER on another segment.

#### Note

LSAs are also called link-state protocol data units (PDUs).

LSAs report the state of routers and the links between routers—hence the term *link state*. Thus, link-state information must be synchronized between routers. To accomplish this, LSAs have the following characteristics:

- LSAs are reliable. There is a method for acknowledging their delivery.
- LSAs are flooded throughout the area (or throughout the domain if there is only one area).
- LSAs have a sequence number and a set lifetime, so each router recognizes that it has the most current version of the LSA.
- LSAs are periodically refreshed to confirm topology information before they age out of the LSDB.

Only by reliably flooding link-state information can every router in the area or domain ensure that it has the latest, most accurate view of the network. Only then can the router make reliable routing decisions that are consistent with the decisions of other routers in the network.

#### OSPF Metric Calculation

Edsger Dijkstra designed the mathematical algorithm used by link-state routing protocols for calculating the best paths through complex networks. By assigning a cost to each link in the network, and by placing the specific node at the root of a tree and summing the costs toward each given destination, the branches of the tree can be calculated to determine the best path to each destination. The best paths are offered to the forwarding database (the routing table).

For OSPF, the default behavior on Cisco routers is that the interface cost is calculated based on its configured bandwidth. The higher the bandwidth, the lower the cost. The default OSPF cost on Cisco routers is calculated using the formula  $(100) / (\text{bandwidth in megabits per second [Mbps]})$ . This formula can also be written as  $(10^8) / (\text{bandwidth in bps})$ .

#### Note

The OSPF RFC 2328 does not specify what the link cost should be, but on Cisco routers it defaults to being inversely proportional to the link's bandwidth.

If the link bandwidth is changed, the OSPF cost will change. Only one cost can be assigned per interface. It is advertised as the link cost in the router link advertisements. Default OSPF costs are as follows:

- 56-kbps serial link—Default cost is 1785.
- 64-kbps serial link—Default cost is 1562.
- T1 (1.544-Mbps serial link)—Default cost is 64.
- E1 (2.048-Mbps serial link)—Default cost is 48.
- Ethernet—Default cost is 10.
- Fast Ethernet—Default cost is 1.
- FDDI—Default cost is 1.
- ATM—Default cost is 1.

You can also manually define an OSPF cost for each interface, which overrides the default cost value (as described in more detail in the "Changing the Cost Metric" section, later in this chapter).

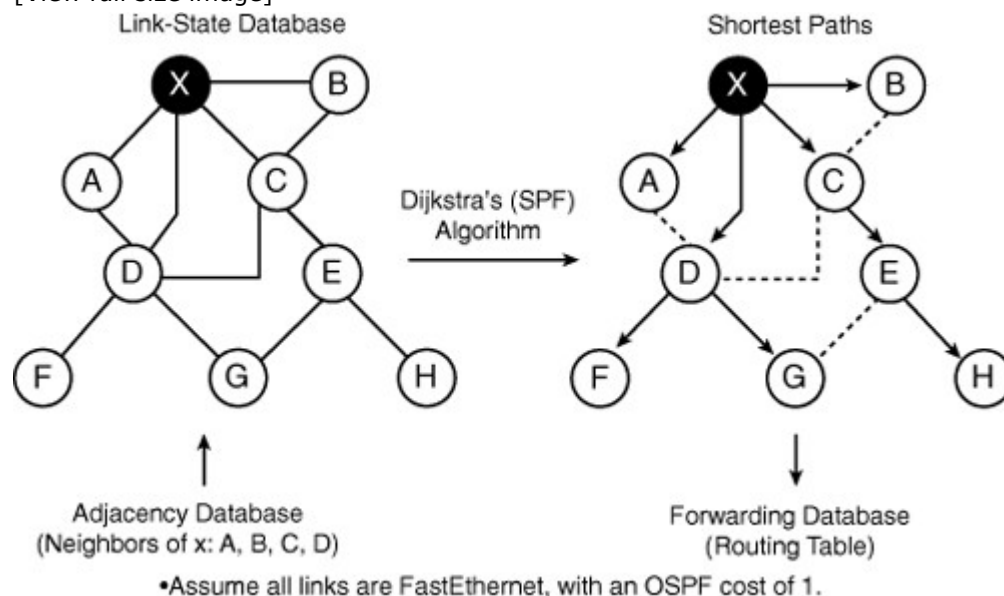
Figure 3-6 is an example of a Dijkstra calculation. The following occurs:



- Router H advertises its presence to Router E. Router E passes Router H's and its own advertisements to its neighbors (Routers C and G). Router G passes these and its own advertisements to D, and so on.
- These LSAs follow the split-horizon rule, which dictates that a router should never advertise an LSA to the router from which it came. In this example, Router E does not advertise Router H's LSAs back to Router H.
- Router X has four neighbor routers: A, B, C, and D. From these routers, it receives the LSAs from all other routers in the network. From these LSAs, it can also deduce the links between all routers and draw the web of routers shown in Figure 3-6.
- Each Fast Ethernet link in Figure 3-6 is assigned an OSPF cost of 1. By summing the costs to each destination, the router can deduce the best path to each destination.
- The right side of Figure 3-6 shows the resulting best paths (the SPF tree) from Router X. From these best paths, shown with solid lines, routes to destination networks attached to each router are offered to the routing table; for each route, the next-hop address is the appropriate neighboring router (A, B, C, or D).

Figure 3-6. SPF Calculations.

[View full size image]



### Link-State Data Structures

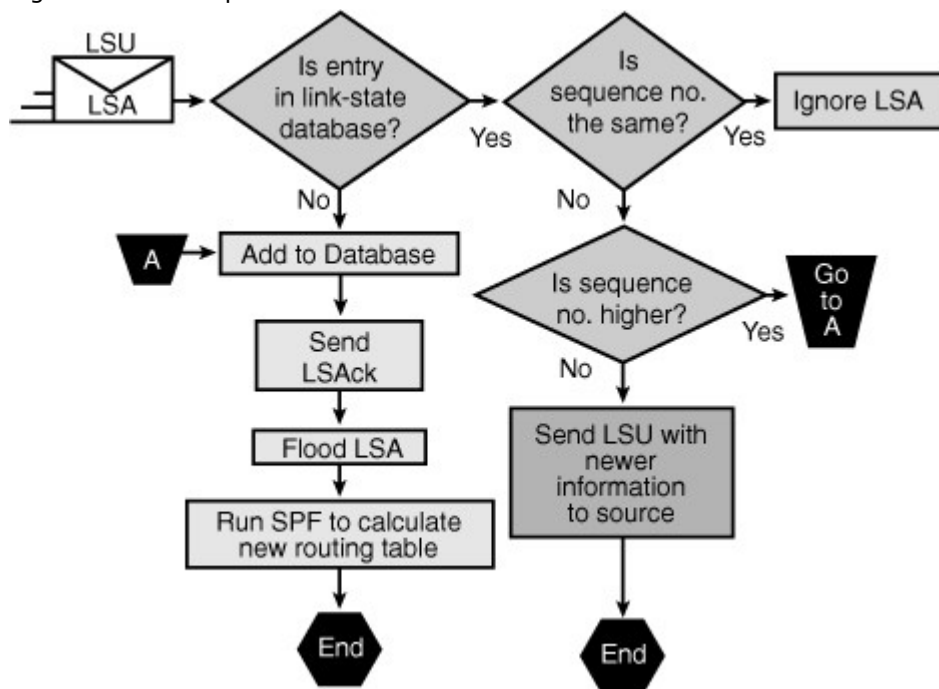
Each LSA entry has its own aging timer, which the Link-State Age field carries. After a default of 30 minutes (expressed in seconds in the Link-State Age field), the router that originated the entry resends the LSA, with a higher sequence number, in a link-state update (LSU), to verify that the link is still active. If the LSA were to reach its maximum age (maxage) of 60 minutes, it would be discarded. An LSU can contain one or more LSAs. This LSA validation method saves on bandwidth compared to distance vector routers, which send their entire routing table at short, periodic intervals.

As shown in Figure 3-7, when each router receives the LSU, it does the following:

- If the LSA entry does not already exist, the router adds the entry to its LSDB, sends back a link-state acknowledgment (LSAck), floods the information to other routers, runs SPF, and updates its routing table.
- If the entry already exists and the received LSA has the same sequence number, the router ignores the LSA entry.
- If the entry already exists but the LSA includes newer information (it has a higher sequence number), the router adds the entry to its LSDB, sends back an LSAck, floods the information to other routers, runs SPF, and updates its routing table.
- If the entry already exists but the LSA includes older information, it sends an LSU to the sender with its newer information.



Figure 3-7. LSA Operations.



#### OSPF Packets

OSPF performs several functions, including the following:

- Neighbor discovery, to form adjacencies
- Flooding link-state information, to facilitate LSDBs being built in each router
- Running SPF to calculate the shortest path to all known destinations
- Populating the routing table with the best routes to all known destinations

Changes might occur in the state of the links after the routing table is initially populated. OSPF detects these changes and responds by flooding the information about the changes within areas, and possibly to other areas, to maintain the LSDBs in all neighboring routers.

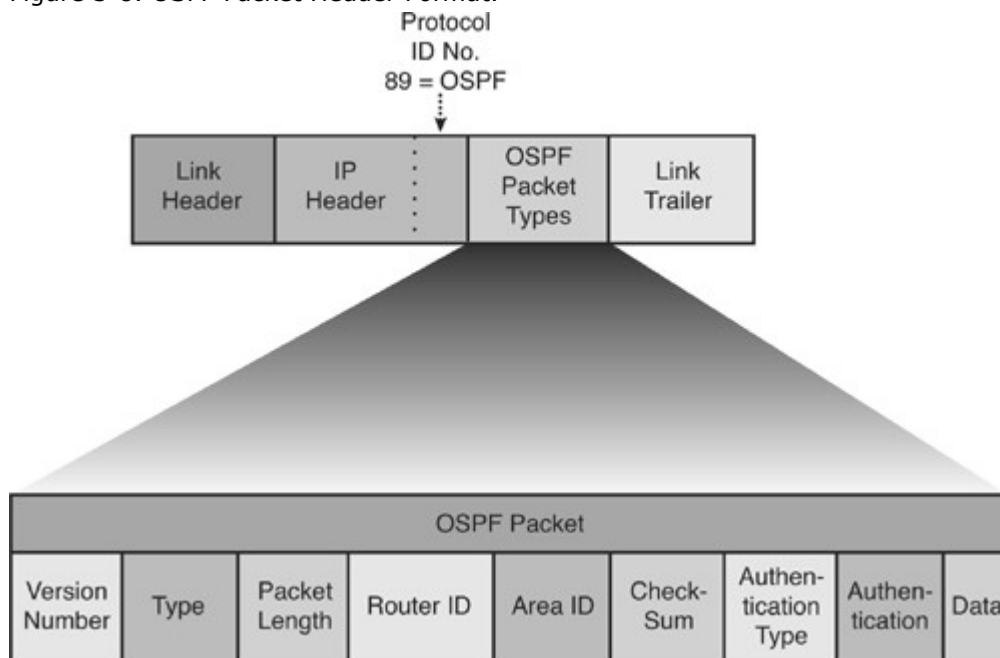
This section describes the five OSPF packet types, listed in Table 3-1, and explains where and how these packets interact to build OSPF neighbor adjacencies and maintain the OSPF topology database.

Table 3-1. OSPF Packets

| Type | Packet Name                | Description                                              |
|------|----------------------------|----------------------------------------------------------|
| 1    | Hello                      | Discovers neighbors and builds adjacencies between them  |
| 2    | Database description (DBD) | Checks for database synchronization between routers      |
| 3    | Link-state request (LSR)   | Requests specific link-state records from another router |
| 4    | LSU                        | Sends specifically requested link-state records          |
| 5    | LSAck                      | Acknowledges the other packet types                      |

All five OSPF packets are encapsulated directly in an IP payload, as shown in Figure 3-8. The OSPF packet does not use Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). OSPF requires a reliable packet transport scheme, and because TCP is not used, OSPF defines its own acknowledgment routine using an acknowledgment packet (OSPF packet type 5).

Figure 3-8. OSPF Packet Header Format.



As shown in Figure 3-8, a protocol identifier of 89 in the IP header indicates an OSPF packet. Each OSPF packet begins with a header with the following fields:

- **Version Number**— Set to 2 for OSPF Version 2, the current IPv4 version of OSPF. (OSPF Version 3 is used for IP version 6 [IPv6], as described in Chapter 8, “Implementing IPv6 in an Enterprise Network.”)
- **Type**— Differentiates the five OSPF packet types.
- **Packet Length**— The length of the OSPF packet in bytes.
- **Router ID**— Defines which router is the packet’s source.
- **Area ID**— Defines the area in which the packet originated.
- **Checksum**— Used for packet header error detection to ensure that the OSPF packet was not corrupted during transmission.
- **Authentication Type**— An option in OSPF that describes either no authentication, clear-text passwords, or encrypted message digest 5 (MD5) for router authentication.
- **Authentication**— Used with authentication type.

Note

OSPF authentication is covered in the “Configuring and Verifying OSPF Authentication” section, later in this chapter.

- **Data**— Contains different information, depending on the OSPF packet type:
  - **For the hello packet**—Contains a list of known neighbors.
  - **For the DBD packet**—Contains a summary of the LSDB, which includes all known router IDs and their last sequence number, among several other fields.
  - **For the LSR packet**—Contains the type of LSU needed and the router ID of the router that has the needed LSU.
  - **For the LSU packet**—Contains the full LSA entries. Multiple LSA entries can fit in one OSPF update packet.
  - **For the LSAck packet**—This data field is empty.

Establishing OSPF Neighbor Adjacencies: Hello

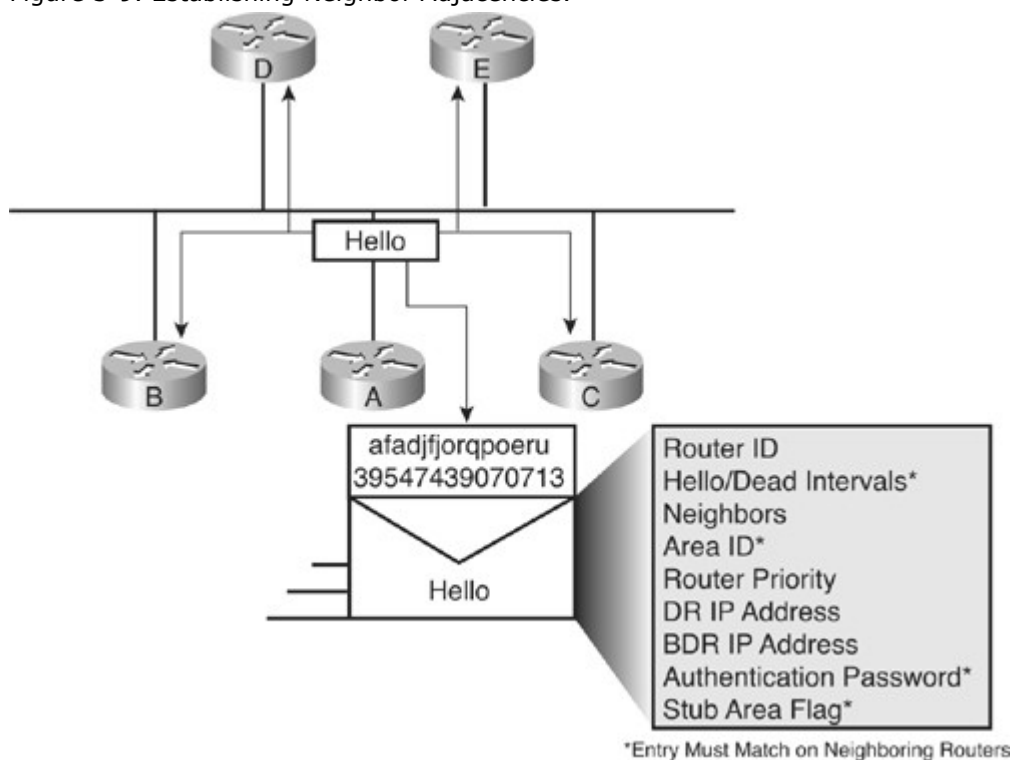
OSPF routing depends on the status of a link between two routers. The Hello protocol establishes and maintains neighbor relationships by ensuring bidirectional (two-way) communication between neighbors. Bidirectional communication occurs when a router sees itself listed in the hello packet received from a

neighbor. Neighbor OSPF routers must recognize each other on the network before they can share routing information.

Each interface participating in OSPF uses the IP multicast address 224.0.0.5 to periodically send hello packets. As shown in Figure 3-9, a hello packet contains the following information:

- **Router ID**— A 32-bit number that uniquely identifies the router. The highest IP address on an active interface is chosen by default unless a loopback interface address exists or the router ID is manually configured (this process is described later, in the “OSPF Router ID” section). For example, IP address 172.16.12.1 would be chosen over 172.16.1.1. This router ID is important in establishing neighbor relationships and coordinating LSU exchanges. The router ID is also used to break ties during the DR and BDR selection processes if the OSPF priority values are equal (as described in the “Electing a DR and BDR and Setting Priority” section, later in this chapter).
- **Hello and dead intervals**— The hello interval specifies how often, in seconds, a router sends hello packets (10 seconds is the default on multiaccess networks). The dead interval is the amount of time in seconds that a router waits to hear from a neighbor before declaring the neighbor router out of service (the dead interval is four times the hello interval by default). These timers must be the same on neighboring routers; otherwise an adjacency will not be established.
- **Neighbors**— The Neighbors field lists the adjacent routers with which this router has established bidirectional communication. Bidirectional communication is indicated when the router sees itself listed in the Neighbors field of the hello packet from the neighbor.
- **Area ID**— To communicate, two routers must share a common segment, and their interfaces must belong to the same OSPF area on that segment. These routers will all have the same link-state information for that area.
- **Router priority**— An 8-bit number that indicates a router’s priority. Priority is used when selecting a DR and BDR.
- **DR and BDR IP addresses**— If known, the IP addresses of the DR and BDR for the specific multiaccess network.
- **Authentication password**— If router authentication is enabled, two routers must exchange the same password. Authentication is not required, but if it is enabled, all peer routers must have the same password.
- **Stub area flag**— A stub area is a special area. The stub area technique reduces routing updates by replacing them with a default route. Two neighboring routers must agree on the stub area flag in the hello packets. The “Configuring OSPF Special Area Types” section, later in this chapter, describes stub areas in greater detail.

Figure 3-9. Establishing Neighbor Adjacencies.



The following hello packet fields must match on neighboring routers for them to establish an adjacency:

- Hello Interval
- Dead Interval
- Area ID
- Authentication Password
- Stub Area Flag

#### Note

For routers to establish an adjacency on an interface, the primary IP addresses on the routers' interfaces must also be on the same subnet with the same mask, and the interface maximum transmission unit (MTU) must match.

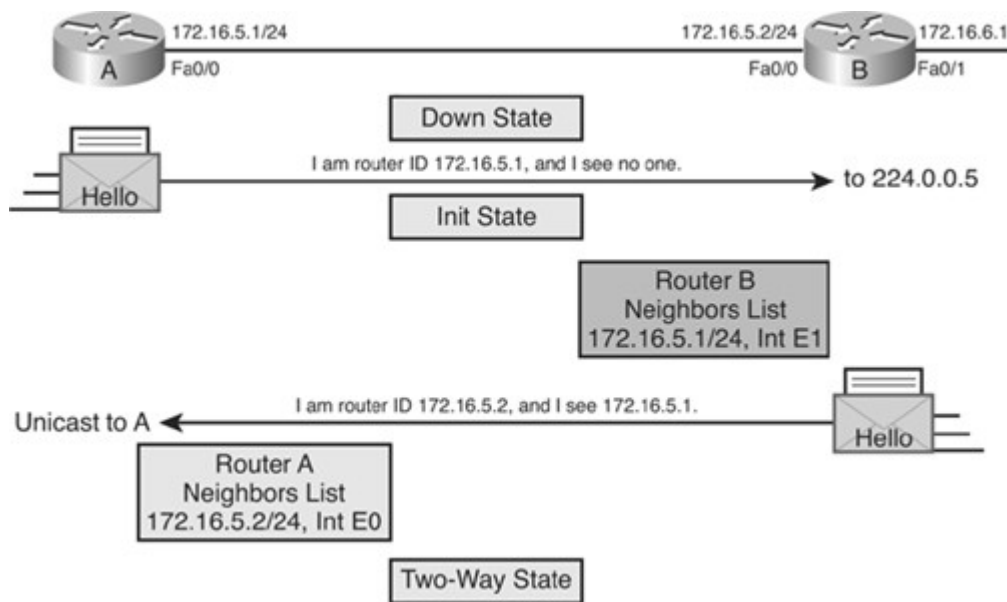
#### Exchange Process and OSPF Neighbor Adjacency States

When routers running OSPF initialize, they first go through an exchange process, using the Hello protocol. In this process, the routers go through various states (detailed in the upcoming "OSPF Neighbor States" section), as shown in Figure 3-10 and described as follows:

1. Router A is enabled on the LAN; OSPF on the interface is in a *down* state because it has not exchanged information with any other router. It begins by sending a hello packet through each of its interfaces participating in OSPF, even though it does not know the identity of the DR or of any other routers. The hello packet is sent out using the multicast address 224.0.0.5.
2. All directly connected routers running OSPF receive the hello packet from Router A and add Router A to their list of neighbors. These routers are now in the neighbor *init* state.
3. All routers that received the hello packet send a unicast reply packet to Router A with their corresponding information. The Neighbor field in the hello packet includes all other neighboring routers, including Router A.
4. When Router A receives these hello packets, it adds all the routers that have its router ID in their hello packets to its own neighbor relationship database. Router A is now in the neighbor *two-way* state with these routers. At this point, all routers that have each other in their lists of neighbors have established bidirectional communication.

Figure 3-10. Establishing Bidirectional Communication.

[View full size image]



Periodically (every 10 seconds by default on broadcast networks) the routers in a network exchange hello packets to ensure that communication is still working.

If the link is a broadcast network, such as an Ethernet LAN link, a DR and BDR must be selected for the link. The DR and BDR form adjacencies with all other routers on the LAN link. This process must occur before the routers can begin exchanging link-state information.

#### Note

If a router joins a broadcast network in which there is already a DR and BDR, it will get to the neighbor *two-way* state with all routers, including the DR and BDR, and those that are DROTHER (not DR or BDR). The joining router will continue to form *full* bidirectional adjacencies only with the DR and BDR.

After the DR and BDR have been selected, the routers are considered to be in the neighbor *exstart* state and they are ready to discover the link-state information about the internetwork and create their LSDBs. The process used to discover the network routes is the exchange protocol, which gets the neighbors to a *full* state of communication.

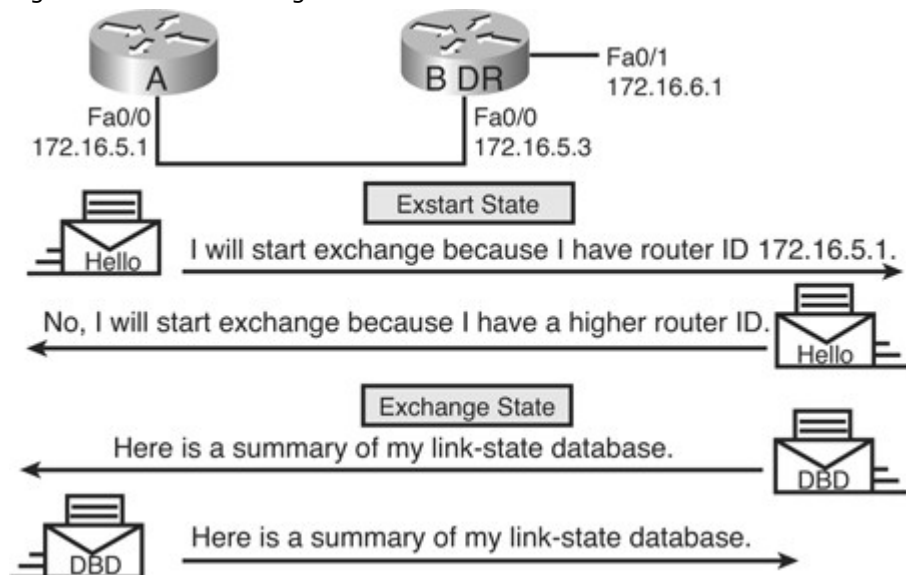
Although the hello packets (OSPF packet type 1) are used in the hello protocol, the other four types of OSPF packets are used during the process of exchanging and synchronizing LSDBs, as follows:

- **OSPF packet type 2 DBD**— Used to describe the LSAs available in the LSDB
- **OSPF packet type 3 LSR**— Used to request missing LSA information
- **OSPF packet type 4 LSU**— Used to send complete LSAs
- **OSPF packet type 5 LSAck**— Used to acknowledge LSUs, to ensure reliable transport and information exchange

OSPF type 4 and type 5 packets are sent to the OSPF multicast IP address, except when retransmitting, when sent across a virtual link, and on nonbroadcast networks. All other packets are sent to a unicast IP address.

The exchange protocol process is shown in Figure 3-11. The first step in this process is for the DR and BDR to establish adjacencies with each of the other routers. In this figure, assume that Router B has been selected as the DR for the link (the DR selection process is described in the “Electing a DR and BDR and Setting Priority” section, later in this chapter). The exchange protocol operates as follows:

Figure 3-11. Discovering the Network Routes.



1. In the neighbor *exstart* state, a master and slave relationship is created between each router and its adjacent DR and BDR. The router with the higher router ID acts as the master during the exchange process.

Note

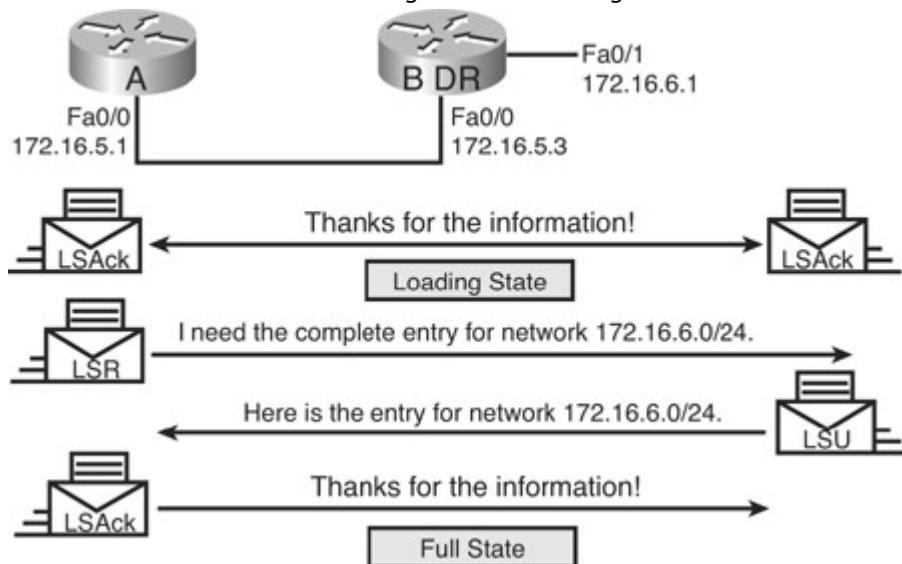
Recall that only the DR exchanges and synchronizes link-state information with the routers to which it has established adjacencies. Having the DR represent the network in this capacity reduces the amount of routing update traffic.

2. The master and slave routers exchange one or more DBD packets (also called DDPs). The routers are in the neighbor *exchange* state.

A DBD includes information about the LSA entry header that appears in the router's LSDB. The entries can be about a link or about a network. Each LSA entry header includes information about the link-state type, the address of the advertising router, the link's cost, and the sequence number. The router uses the sequence number to determine the "newness" of the received link-state information.

3. When the router receives the DBD, it performs the following actions, as shown in Figure 3-12:

Figure 3-12. Adding Link-State Entries.



- It acknowledges the receipt of the DBD using the LSAck packet.

- It compares the information it received with the information it has in its own LSDB. If the DBD has a more current link-state entry, the router sends an LSR to the other router. When routers are sending LSRs, they are in the neighbor *loading* state.
- The other router responds with the complete information about the requested entry in an LSU packet. Again, when the router receives an LSU, it sends an LSAck.

4. The router adds the new link-state entries to its LSDB.

After all LSRs have been satisfied for a given router, the adjacent routers are considered synchronized and in a neighbor *full* state. When adjacent routers are in a full state, they do not repeat the exchange protocol unless the neighbor state changes from full.

The routers must be in a full state with a neighbor before they can route traffic on an interface. Remember, on a broadcast link, routers reach full neighbor state only with the DR and BDR, but they reach two-way state with other routers on the link. At this point, all the routers in the area should have identical LSDBs.

#### OSPF Neighbor States

The following is a brief summary of the states OSPF may pass through before becoming adjacent to (neighbors with) another router:

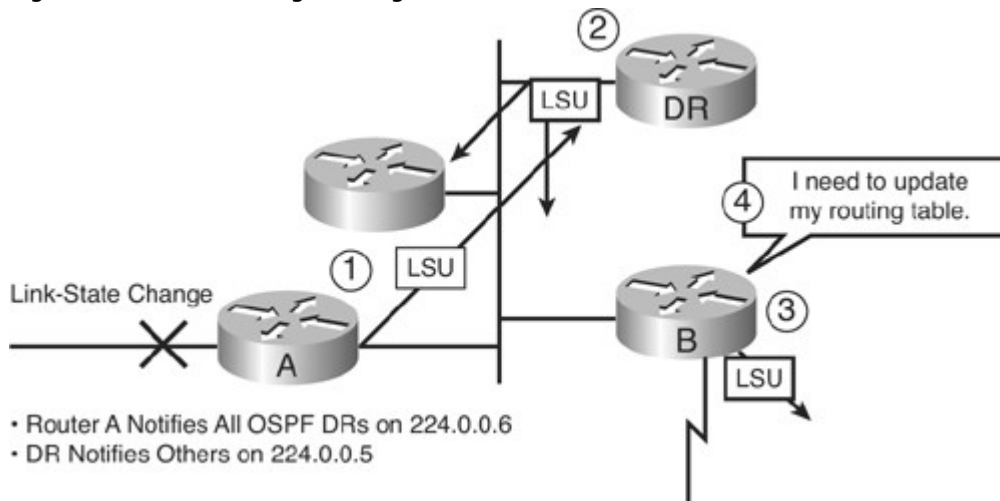
- **Down:** No active neighbor detected.
- **Init:** Hello packet received.
- **Two-way:** Router sees its own router ID in a received hello packet.
- **ExStart:** Master/slave roles determined.
- **Exchange:** DBDs (summary of LSDB) sent.
- **Loading:** Exchange of LSRs and LSUs, to populate LSDBs.
- **Full:** Neighbors fully adjacent.

With neighbors on NBMA interfaces (such as Frame Relay), OSPF may also go into the Attempt state before the init state. The Attempt state indicates that no recent information has been received from the neighbor and that an effort will be made to contact the neighbor by sending hello packets at a reduced poll interval.

#### Maintaining Routing Information

In a link-state routing environment, it is very important for the link-state databases of all routers to stay synchronized. When a change in a link state occurs, as shown in Figure 3-13, the routers use a flooding process to notify the other routers in the network of the change. LSUs provide the mechanism for flooding LSAs. OSPF simplifies the synchronization issue by requiring only adjacent routers to remain synchronized.

Figure 3-13. Maintaining Routing Information.



#### Note

Although it is not shown in Figure 3-13, all LSUs are acknowledged.

In general, the following are the flooding process steps in a multiaccess network:

1. A router notices a change in a link state and multicasts an LSU packet, which includes the updated LSA entry with the sequence number incremented, to 224.0.0.6. This address goes to all OSPF DRs and



BDRs. (On point-to-point links, the LSU is multicast to 224.0.0.5.) An LSU packet might contain several distinct LSAs.

2. The DR receives the LSU, processes it (as described earlier in Figure 3-7), acknowledges the receipt of the change and floods the LSU to other routers on the network using the OSPF multicast address 224.0.0.5. After receiving the LSU, each router responds to the DR with an LSAck. To make the flooding procedure reliable, each LSA must be acknowledged separately.
3. If a router is connected to other networks, it floods the LSU to those other networks by forwarding the LSU to the DR of the multiaccess network (or to the adjacent router if in a point-to-point network). That DR, in turn, multicasts the LSU to the other routers in the network.
4. The router updates its LSDB using the LSU that includes the changed LSA. It then recomputes the SPF algorithm against the updated database after a short delay and updates the routing table as necessary.

#### Note

The **timers throttle spf** router configuration command, introduced in Cisco IOS Software Release 12.2(14)S, enables the OSPF throttling feature so that the SPF calculations can be potentially delayed during network instability. The "OSPF Shortest Path First Throttling" section of the *Cisco IOS IP Routing Protocols Configuration Guide* provides details about this command. This command replaces the **timers spf** command in earlier Cisco IOS Software releases.

Notice in this process, OSPF uses two multicast addresses:

- 224.0.0.5 goes to all OSPF routers on the link.
- 224.0.0.6 goes to the DR and BDR on the link.

Summaries of individual link-state entries, not the complete link-state entries, are sent every 30 minutes to ensure LSDB synchronization. Each link-state entry has a timer to determine when the LSA refresh update must be sent. Each link-state entry also has a maximum age (maxage) of 60 minutes. As mentioned, if a link-state entry is not refreshed within 60 minutes, it is removed from the LSDB.

A change in the topology database is a necessary but not sufficient condition for SPF recalculation. SPF is triggered if any of the following occur:

- The LSA's Options field has changed.
- The LSA's LS age is set to maxage.
- The Length field in the LSA header has changed.
- The contents of the LSA (excluding the LSA header) have changed.

An SPF calculation is performed separately for each area in the topology database.

#### Note

In a Cisco router, if a route already exists, the routing table is used at the same time the SPF algorithm is calculating. However, if the SPF is calculating a new route, the new route is used only after the SPF calculation is complete.

### OSPF Link-State Sequence Numbers

This section describes the combination of the maxage timer, the refresh timer, and the link-state sequence numbers, which help OSPF maintain a database of only the most recent link-state records.

An LSA is considered to be more recent if it has the following:

- A higher sequence number
- A higher checksum number (if the sequence numbers are equal)
- An age equal to maxage (indicating the LSA is poisoned)
- A significantly smaller (younger) LS age

The link-state sequence number field in an LSA header is 32 bits long. Beginning with the leftmost bit set, the first legal sequence number is 0x80000001, and the last number is 0x7FFFFFFF. The sequence number is used to detect old or redundant LSA records. The larger the number, the more recent the LSA.

To ensure an accurate database, OSPF floods (refreshes) each LSA every 30 minutes. This interval is called the *LSRefreshTime*. Each time a record is flooded, the sequence number is incremented by 1. When a router receives a new LSA update it resets the LSA record's age. An LSA never remains in the database for longer than the maximum age of 1 hour without a refresh.

It is possible for an LSA to exist in the database for long periods of time, being refreshed every 30 minutes. At some point the sequence number needs to wrap back to the starting sequence number. When this process

occurs, the existing LSA is prematurely aged out (the maxage timer is immediately set to 1 hour) and flushed. The LSA then restarts its sequencing at 0x80000001.

The partial output of the `show ip ospf database` command in Example 3-1 demonstrates how the LS age and LS sequence numbers are kept in the database. Every OSPF router announces a router LSA for those interfaces that it owns in an area. The link ID in the output is the router ID of the router that created the router LSA. The advertising router (shown as ADV Router in the output) is the router ID of the OSPF router that announced the router LSA. Generally, the link ID and advertising router for a router LSA are the same. The first router LSA entry in the OSPF database shown in Example 3-1 indicates that the router LSA with link ID 192.168.1.67 has been updated eight times (because the sequence number is 0x80000008) and that the last update occurred 48 seconds ago (as indicated in the Age column).

Example 3-1. LSA Database Sequence Numbers and Maxage

```
RouterA#show ip ospf database
OSPF Router with ID (192.168.1.67) (Process ID 10)
 Router Link States (Area 1)
Link ID ADV Router Age Seq# Checksum Link count
192.168.1.67 192.168.1.67 48 0x80000008 0xB112 2
192.168.2.130 192.168.2.130 212 0x80000006 0x3F44 2
<output omitted>
```

#### Verifying Packet Flow

The `debug ip ospf packet` command is used to troubleshoot and verify that OSPF packets are flowing properly between two routers; Example 3-2 demonstrates output from this command. Notice that the output shows the fields in the OSPF header, but they are not described in any detail. Table 3-2 describes the OSPF packet header fields represented in this output.

Table 3-2. *debug ip ospf packet* Command

| Field  | Description                                                                                                                               |
|--------|-------------------------------------------------------------------------------------------------------------------------------------------|
| v:     | Identifies the version of OSPF; OSPFv2 in this example.                                                                                   |
| t:     | Specifies the OSPF packet type:<br>1—Hello<br>2—DBD<br>3—LSR<br>4—LSU<br>5—LSAck<br>This example has a Type 1 packet, a hello packet.     |
| l:     | Specifies the OSPF packet length in bytes; 48 in this example.                                                                            |
| rid:   | Displays the OSPF router ID; 10.0.0.12 in this example.                                                                                   |
| aid:   | Shows the OSPF area ID; 0.0.0.1 in this example.                                                                                          |
| chk:   | Displays the OSPF checksum; D882 in this example.                                                                                         |
| aut:   | Provides the OSPF authentication type:<br>0—No authentication<br>1—Simple password<br>2—MD5<br>No authentication is used in this example. |
| auk:   | Specifies the OSPF authentication key, if used. It is not used in this example.                                                           |
| keyid: | Displays the MD5 key ID; only used for MD5 authentication. It is not used in this example.                                                |

| Field | Description                                                                                     |
|-------|-------------------------------------------------------------------------------------------------|
| seq:  | Provides the sequence number; only used for MD5 authentication. It is not used in this example. |
| from: | Interface from which this packet was received, S0/0/0.2 in this example.                        |

#### Example 3-2. Debug of a Single Packet

```
R1#debug ip ospf packet
OSPF packet debugging is on
R1#
*Apr 16 11:03:51.206: OSPF: rcv. v:2 t:1 l:48 rid:10.0.0.12
aid:0.0.0.1 chk:D882 aut:0 auk: from Serial0/0/0.2
```

### Configuring and Verifying Basic OSPF Routing

This section discusses the basic OSPF configuration and verification of single-area and multiarea OSPF networks.

The following topics are discussed:

- Planning and configuring OSPF
- OSPF router ID
- Verifying OSPF operations

#### Planning and Configuring OSPF

This section describes how to plan and configure basic OSPF and provides some detailed examples.

#### Planning OSPF Routing Implementations

This section describes how to plan, implement, and document an OSPF deployment.

When preparing to deploy OSPF routing in a network, the existing network state and requirements first need to be gathered, and deployment options considered. Considerations for OSPF include the following:

- **IP addressing plan**— The IP addressing plan governs how OSPF can be deployed and how well the OSPF deployment will scale. A detailed hierarchical IP subnet and addressing plan must be produced, to enable OSPF summarization, allow the network to scale more easily, and to optimize OSPF behavior.
- **Network topology**— The topology consists of the devices (routers, switches, and so on) and the links connecting them. A detailed network topology should be created to assess OSPF scalability requirements and to determine which OSPF features might be required (for example, multiple areas, OSPF summarization, stub areas, and redistribution). The topology should include backup links where necessary.
- **OSPF areas**— Dividing an OSPF network into areas decreases the LSDB size and limits the propagation of link-state updates when the topology changes. The routers that are to be ABRs and ASBRs must be identified, as are those that are to perform any summarization or redistribution.

After the requirements have been assessed, the implementation plan can be created. The implementation plan should include the following steps:

- Define the network requirements
- Gather the required parameters
- Define the OSPF routing parameters
- Configure OSPF
- Verify the OSPF configuration

The information necessary to implement OSPF routing includes the following:

- The IP addresses to be configured on individual router interfaces.
- A list of routers on which OSPF is to be enabled, along with the OSPF process number to use and the connected networks that are to run OSPF and that need to be advertised (per individual router).
- The area in which each interface is to be configured.

- Metrics that need to be applied to specific interfaces, to influence the OSPF best path selection. The required metric and the interface where the metric needs to be applied should be specified.

In the implementation plan, the list of tasks for each router in the network must be defined. For OSPF, the tasks include the following:

- Enabling the OSPF routing protocol, directly on an interface or by using the correct **network** command under the OSPF routing process configuration mode.
- Assigning the correct area id to the interface, via the OSPF configuration on the interface or under the OSPF routing process configuration mode.
- Optionally configuring the metric to appropriate interfaces.

After implementing OSPF, proper deployment on each router should be verified. Verification tasks include the following:

- Verifying that the appropriate OSPF neighbor relationships and adjacencies are established
- Verifying that the OSPF LSDB is populated with the necessary information
- Verifying that IP routing table is populated with the necessary information
- Verifying that there is connectivity in the network between routers and to other devices
- Verifying that OSPF behaves as expected in a case of a topology change, by testing link failure and router failure events

After a successful OSPF deployment, the solution and verification process and results should be documented for future reference. Documentation should include a topology map, the IP addressing plan, the area hierarchy, the networks and interfaces included in OSPF on each router, the default and any special metrics configured, and the verification results.

#### Configuring Basic OSPF

This section describes how you can configure OSPF using the **router ospf** command with **network area** commands under it, or directly on router interfaces with the **ip ospf area** command.

To configure the OSPF process, do the following:

- Step 1. Enable the OSPF process on the router using the `router ospf process-id [vrf vpn-name]` global configuration command. Table 3-3 describes the parameters of the `router ospf` command.

Table 3-3. *router ospf* Command

| Parameter                 | Description                                                                                                                                                                                                 |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>process-id</code>   | An internally used number that identifies the OSPF routing process. The <i>process-id</i> does not need to match process IDs on other routers. It can be any positive integer in the range from 1 to 65535. |
| <code>vrf vpn-name</code> | (Optional) Specifies the name of the virtual private network (VPN) routing and forwarding (VRF) instance to associate with OSPF VRF processes.                                                              |

- Step 2. Identify which interfaces on the router are part of the OSPF process, and identify the OSPF area to which the network belongs, using the `network ip-address wildcard-mask area area-id router` configuration command. Table 3-4 describes the parameters of the `network` command in the context of OSPF.

Table 3-4. *network* Command Parameters with OSPF

| Parameter                  | Description                                                                                                                                                                                                                                                   |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ip-address</code>    | Either the network address, subnet address, or the interface's address. This address instructs the router to determine which links to advertise to, which links to check for advertisements, and which networks to advertise.                                 |
| <code>wildcard-mask</code> | Determines how to interpret the <i>ip-address</i> . The mask has wildcard bits, in which 0 is a match and 1 is "don't care." For example, 0.0.255.255 indicates a match in the first 2 bytes. To specify the interface address, use the mask 0.0.0.0 to match |

| Parameter | Description                                                                                                                                                                                                             |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|           | all 4 bytes of the address.<br>An address and wildcard mask combination of 0.0.0.0 255.255.255.255 matches all interfaces on the router.                                                                                |
| area-id   | Specifies the OSPF area to be associated with the address. This parameter can be a decimal number in the range from 0 to 4294967295, or it can be in dotted-decimal notation similar to an IP address, such as A.B.C.D. |

Alternatively, starting with Cisco IOS Software Release 12.3(11)T (and some specific versions of earlier releases), OSPF can be enabled directly on the interface using the `ip ospf process-id area area-id [secondaries none]` interface configuration command. This command simplifies the configuration of unnumbered interfaces. Because this command is configured explicitly for the interface, it takes precedence over the network area command. Table 3-5 describes the parameters of the `ip ospf area` command.

Table 3-5. *ip ospf area* Command

| Parameter         | Description                                                                                                                                                                                                                         |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| process-id        | An internally used number that identifies the OSPF routing process. The process ID is entered as a decimal number in the range from 1 to 65535.                                                                                     |
| area-id           | Specifies the OSPF area to be associated with the interface. The area ID is entered either as a decimal value in the range from 0 to 4294967295, or it can be in dotted-decimal notation similar to an IP address, such as A.B.C.D. |
| second-aries none | (Optional) Prevents secondary IP addresses on the interface from being advertised.                                                                                                                                                  |

#### Note

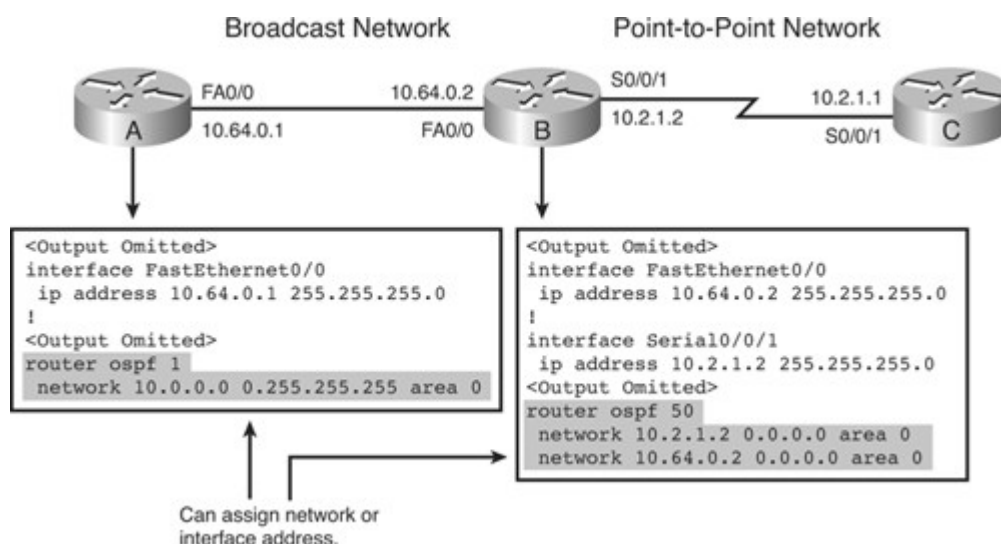
To configure an unnumbered interface, use the **`ip unnumbered interface-type interface-number`** interface configuration command. This command enables IP processing on an interface without assigning an explicit IP address to the interface. The interface will use the IP address of the interface specified by the *interface-type interface-number* parameters as the source address of traffic from the configured interface. The interface specified in the command must be in the “up” state.

#### Single-Area OSPF Configuration Example

Figure 3-14 shows the OSPF configuration for Fast Ethernet broadcast networks and serial point-to-point links, on two of the three routers. All three routers in Figure 3-14 are assigned to area 0 and are configured for network 10.0.0.0.

Figure 3-14. Configuring OSPF on Internal Routers of a Single Area.

[View full size image]



Router A uses a general **network 10.0.0.0 0.255.255.255** statement. This technique assigns all interfaces defined in the 10.0.0.0 network to OSPF process 1.

Router B uses a specific host address technique. The wildcard mask of 0.0.0.0 matches all 4 bytes of the address. This technique allows the administrator to define which specific interfaces will run OSPF.

Although the two examples shown in Figure 3-14 are a commonly used combination of a network statement and a wildcard mask, others could also work. For instance, a range of subnets could be specified.

Note

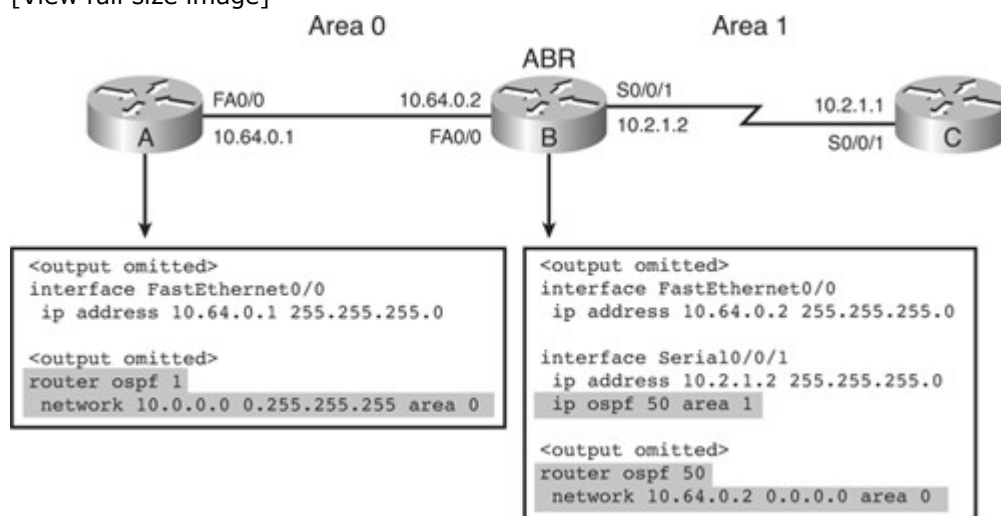
The network command for OSPF is used strictly to enable OSPF for a single interface or for multiple interfaces. The network command and its wildcard mask are not used for route summarization purposes.

#### Multiarea OSPF Configuration Example

Figure 3-15 shows an example of multiarea OSPF configuration. Router A is in area 0, Router C is in area 1, and Router B is the ABR between the two areas.

Figure 3-15. Configuring OSPF for Multiple Areas.

[View full size image]



The configuration for Router A is the same as it was in Figure 3-14.

Router B has a **network** statement for area 0. The configuration for area 1 in this example uses the **ip ospf 50 area 1** interface configuration command. Alternatively a separate **network** router configuration command, such as **network 10.2.1.2 0.0.0.0 area 1**, could have been used.

## OSPF Router ID

An OSPF router ID uniquely identifies each OSPF router in the network. The OSPF routing process chooses a router ID for itself when it starts up. The router ID is a unique number in IP address format that can be assigned in the following ways:

- By default, the highest IP address of any active physical interface when OSPF starts is chosen as the router ID. The interface does not have to be part of the OSPF process, but it has to be up. There must be at least one "up" IP interface on the router for OSPF to use as the router ID. If no up interface with an IP address is available when the OSPF process starts, the following error message occurs:  
R1(config)#**router ospf 1**  
2w1d: %OSPF-4-NORTRID: OSPF process 1 cannot start.
- Alternatively, if a loopback interface exists, its IP address will always be preferred as the router ID instead of the IP address of a physical interface, because a loopback interface never goes down. If there is more than one loopback interface, the highest IP address on any active loopback interface becomes the router ID.
- Alternatively, if the **router-id ip-address** OSPF router configuration command is used, it will override the use of the address of a physical or loopback interface as the router ID. Using the **router-id** command is the preferred procedure for setting the router ID.

The OSPF database uses the router ID to uniquely describe each router in the network. Remember that every router keeps a complete topology database of all routers and links in an area and network. Therefore, router IDs should be unique throughout the OSPF autonomous system, no matter how they are configured.

After the router ID has been set, it does not change, even if the interface that the router is using for the router ID goes down. The OSPF router ID changes only if the router reloads or if the OSPF routing process restarts.

### OSPF Router ID Stability

If a physical interface address is being used as the router ID, and that physical interface fails, and the router (or OSPF process) is restarted, the router ID will change. This change in router ID makes it more difficult for network administrators to troubleshoot and manage OSPF. The stability provided by using a loopback interface for the router ID or by using the **router-id** command comes from the router ID staying the same, regardless of the state of the physical interfaces.

The following sections describe how loopback interfaces are configured, and how the router ID is configured and verified.

### Loopback Interfaces

To allow OSPF to use a loopback address as the router ID, first define a loopback interface with the **interface loopback number** global configuration command, and then configure an IP address on the loopback interface.

As mentioned, configuring an IP address on a loopback interface overrides the highest IP address on any active physical interface being used as the router ID. OSPF is more stable if a loopback interface address is used, rather than a physical interface address, because the loopback interface is always active and cannot fail, whereas a real interface could go down. For this reason, you should use a loopback address on all key routers (unless the **router-id** command is used).

If the loopback address is advertised with the **network** command, then this address can be pinged for testing purposes, and used for management of the router and for the OSPF router ID. The IP address chosen for the loopback address can be either a private or public IP address, dictated by the network requirements (of course, if a private IP address is used it saves public IP address space).

### Note

Using a loopback address requires a different subnet for each router, unless the host address itself is advertised. By default, OSPF advertises loopback interface addresses as /32 host routes.

### OSPF router-id Command

The **router-id ip-address** OSPF router configuration command ensures that OSPF selects a specific planned router ID. The *ip-address* parameter can be any unique arbitrary 32-bit value in an IP address format (dotted decimal).



After configuring the router-id command, use the clear ip ospf process EXEC command to restart the OSPF routing process, so the router reselects the new IP address as its router ID. For example, the commands shown in Example 3-3 ensure that OSPF selects the preconfigured router ID 172.16.1.1.

**Caution**

The **clear ip ospf process** command temporarily disrupts an operational network.

**Example 3-3. router-id Command**

```
Router(config)#router ospf 1
Router(config-router)#router-id 172.16.1.1

Router#clear ip ospf process
```

**Note**

Changing a router ID of a router whose router ID was set with the **router-id** command requires only that the OSPF process be cleared. However, changing the OSPF router ID of a router whose router ID was set by configuring a loopback interface address may require you to either reboot the router or to disable and then enable OSPF. (The IOS documentation is unclear on this point; we have seen both of these ways, and clearing the OSPF process, work in testing.)

**Verifying the OSPF Router ID**

Use the **show ip ospf** command to verify the OSPF router ID. This command also displays OSPF timer settings and other statistics, including the number of times the SPF algorithm has been executed. Optional parameters allow you to specify other information to be displayed.

Example 3-4 shows example output from this command when executed on Router B in Figure 3-15.

**Example 3-4. show ip ospf Command from Router B in Figure 3-15**

```
Code View: Scroll / Show All
RouterB#show ip ospf
Routing Process "ospf 50" with ID 10.64.0.2
Supports only single TOS(TOS0) routes
Supports opaque LSA
Supports Link-local Signaling (LLS)
Supports area transit capability
It is an area border router
Initial SPF schedule delay 5000 msec
Minimum hold time between two consecutive SPF's 10000 msec
Maximum wait time between two consecutive SPF's 10000 msec
Incremental-SPF disabled
Minimum LSA interval 5 secs
Minimum LSA arrival 1000 msec
LSA group pacing timer 240 secs
Interface flood pacing timer 33 msec
Retransmission pacing timer 66 msec
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 2. 2 normal 0 stub 0 nssa
Number of areas transit capable is 0
External flood list length 0
Area BACKBONE(0)
Area BACKBONE(0)
```

```
Area has no authentication
SPF algorithm last executed 00:01:25.028 ago
SPF algorithm executed 7 times
Area ranges are
Number of LSA 6. Checksum Sum 0x01FE3E
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
```

#### Area 1

```
Number of interfaces in this area is 1
Area has no authentication
SPF algorithm last executed 00:00:54.636 ago
SPF algorithm executed 3 times
Area ranges are
Number of LSA 4. Checksum Sum 0x01228A
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
```

RouterB#

### Verifying OSPF Operations

To verify that OSPF has been properly configured, use the following **show** commands:

- The **show ip ospf** command displays the OSPF router ID, OSPF timers, the number of times the SPF algorithm has been executed, and LSA information.
- The **show ip ospf interface** *[type number]* [**brief**] command verifies that interfaces are configured in the intended areas. In addition, this command displays the timer intervals (including the hello interval) and shows the neighbor adjacencies.
- The **show ip ospf neighbor** *[type number]* *[neighbor-id]* [**detail**] command displays a list of neighbors, including their OSPF router ID, their OSPF priority, their neighbor adjacency state (such as init, exstart, or full), and the dead timer.
- The **show ip route ospf** command displays the OSPF routes known to the router. This command is one of the best ways to determine connectivity between the local router and the rest of the internetwork. This command also has optional parameters so that you can further specify the information to be displayed, including the OSPF *process-id*.
- The **show ip protocols** command displays IP routing protocol parameters, including timers, filters, metrics, networks, and other information for the entire router.

The **debug ip ospf events** command displays OSPF-related events, such as adjacencies, flooding information, DR selection, and SPF calculations. The **debug ip ospf adj** command tracks adjacencies as they go up and down. The **debug ip ospf packet** command verifies that OSPF packets are flowing.

#### Note

The **log-adjacency-changes** router configuration command configure the router to send a Syslog message when an OSPF neighbor goes up or down.

The following sections illustrate example output from some of these commands, and the syntax of the command parameters.

## Note

Sample output from the show ip ospf command was provided in the “Verifying the OSPF Router ID” section, earlier in this chapter.

Sample output from the debug ip ospf packet command was provided in the “OSPF Link-State Sequence Numbers” section, earlier in this chapter.

Sample output from the debug ip ospf adj command is provided in the “Displaying OSPF Adjacency Activity” section, later in this chapter.

## The show ip ospf interface Command

Table 3-6 describes the parameters of the show ip ospf interface [type number] [brief] command.

Table 3-6. *show ip ospf interface* Command

| Parameter | Description                                                                                                              |
|-----------|--------------------------------------------------------------------------------------------------------------------------|
| type      | (Optional) Specifies the interface type                                                                                  |
| number    | (Optional) Specifies the interface number                                                                                |
| brief     | (Optional) Displays brief overview information for OSPF interfaces, states, addresses and masks, and areas on the router |

The show ip ospf interface command output in Example 3-5 is from Router A in the configuration example in Figure 3-15 and details the OSPF status of the Fast Ethernet 0/0 interface. This command verifies that OSPF is running on that particular interface and shows the OSPF area it is in. The command also displays other information, such as the OSPF process ID, the router ID, the OSPF network type, the DR and BDR, timers, and neighbor adjacency information.

Example 3-5. *show ip ospf interface* Command on Router A in Figure 3-15

```
RouterA#show ip ospf interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
Internet Address 10.64.0.1/24, Area 0
Process ID 1, Router ID 10.64.0.1, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DROTHER, Priority 0
Designated Router (ID) 10.64.0.2, Interface address 10.64.0.2
No backup designated router on this network
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
 oob-resync timeout 40
 Hello due in 00:00:04
Supports Link-local Signaling (LLS)
Index 1/1, flood queue length 0
Next 0x0(0)/0x0(0)
Last flood scan length is 1, maximum is 4
Last flood scan time is 0 msec, maximum is 4 msec
Neighbor Count is 1, Adjacent neighbor count is 1
 Adjacent with neighbor 10.64.0.2 (Designated Router)
Suppress hello for 0 neighbor(s)
```

## The show ip ospf neighbor Command

One of the most important OSPF troubleshooting commands is the **show ip ospf neighbor** [type number] [neighbor-id] [detail] command. OSPF does not send or receive updates without having full adjacencies between neighbors. This command displays OSPF neighbor information for each interface.

Table 3-7 contains information about the parameters of this command.

Table 3-7. *show ip ospf neighbor* Command

| Parameter   | Description                                  |
|-------------|----------------------------------------------|
| type        | (Optional) Specifies the interface type      |
| number      | (Optional) Specifies the interface number    |
| neighbor-id | (Optional) Specifies the neighbor ID         |
| detail      | (Optional) Displays details of all neighbors |

Example 3-6 illustrates output from Router B in Figure 3-15. Router B has two neighbors. The first entry in the table represents the adjacency formed on the Fast Ethernet interface. A FULL state means that the LSDB has been exchanged successfully. The DROTHER entry means that a router other than this neighboring router (Router A) is the designated router. (Note that the OSPF priority on Router A's Fast Ethernet 0/0 interface has been set to 0, indicating that it cannot be the DR or BDR on that interface.)

Example 3-6. *show ip ospf neighbor* Command from Router B in Figure 3-15

RouterB#**show ip ospf neighbor**

| Neighbor ID | Pri | State        | Dead Time | Address   | Interface       |
|-------------|-----|--------------|-----------|-----------|-----------------|
| 10.64.0.1   | 0   | FULL/DROTHER | 00:00:30  | 10.64.0.1 | FastEthernet0/0 |
| 10.2.1.1    | 0   | FULL/ -      | 00:00:34  | 10.2.1.1  | Serial0/0/1     |

The second line of output in Example 3-6 represents Router C, Router B's neighbor on the serial interface. DR and BDR are not used on point-to-point interfaces (as indicated by a dash [-]).

Example 3-7 shows further output from Router B in Figure 3-15, providing details of Router B's neighbors.

Example 3-7. *show ip ospf neighbor detail* Command from Router B in Figure 3-15

Code View: Scroll / Show All

RouterB#**show ip ospf neighbor detail**

**Neighbor 10.64.0.1, interface address 10.64.0.1**

In the area 0 via interface FastEthernet0/0  
 Neighbor priority is 0, State is FULL, 16 state changes  
 DR is 10.64.0.2 BDR is 0.0.0.0  
 Options is 0x52  
 LLS Options is 0x1 (LR)  
 Dead timer due in 00:00:35  
 Neighbor is up for 00:07:14  
 Index 2/2, retransmission queue length 0, number of retransmission 0  
 First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)  
 Last retransmission scan length is 0, maximum is 0  
 Last retransmission scan time is 0 msec, maximum is 0 msec

**Neighbor 10.2.1.1, interface address 10.2.1.1**

In the area 1 via interface Serial0/0/1  
**Neighbor priority is 0, State is FULL, 6 state changes**  
 DR is 0.0.0.0 BDR is 0.0.0.0  
 Options is 0x52  
 LLS Options is 0x1 (LR)  
 Dead timer due in 00:00:39  
 Neighbor is up for 00:01:50  
 Index 1/1, retransmission queue length 0, number of retransmission 1  
 First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)  
 Last retransmission scan length is 1, maximum is 1

Last retransmission scan time is 0 msec, maximum is 0 msec

The show ip route ospf Command

As illustrated in Example 3-8, the show ip route ospf command is used to verify the OSPF routes in the IP routing table.

The O code indicates that the route was learned from OSPF. The IA code indicates that the learned route is in another area (interarea). In Example 3-8, the 10.2.1.0 subnet is learned on Fast Ethernet 0/0 via neighbor 10.64.0.2. The [110/782] in the routing table represents the administrative distance assigned to OSPF (110) and the total cost of the route to subnet 10.2.1.0 (cost of 782).

Example 3-8. show ip route ospf Command

```
RouterA#show ip route ospf
 10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
O IA 10.2.1.0/24 [110/782] via 10.64.0.2, 00:03:05, FastEthernet0/0
RouterA#
```

The show ip protocols Command

The **show ip protocols** command verifies that OSPF is running and provides OSPF routing protocol parameters, including timers, filters, metrics, networks, and other information for the entire router.

The command output in Example 3-9 shows that the OSPF routing protocol with process number 1 is configured. The router ID of the router is 10.64.0.1 and it belongs to area 0.

Example 3-9. show ip protocols Command

```
RouterA#show ip protocols
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 10.64.0.1
 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
 Reference bandwidth unit is 100 mbps
<output omitted>
```

The debug ip ospf events Command

As illustrated in Example 3-10, the debug ip ospf events command is used to display OSPF-related events. This sample output shows that the router received a hello packet on its Serial 0/0/1 interface.

Example 3-10. debug ip ospf events Command

```
Code View: Scroll / Show All
R1#debug ip ospf events
OSPF events debugging is on
*Apr 27 11:47:00.942: OSPF: Rcv hello from 10.0.0.12 area 1 from Serial0/0/1
10.1.0.2
*Apr 27 11:47:00.942: OSPF: End of hello processing
```

## Understanding OSPF Network Types

Understanding that an OSPF area is made up of different types of network links is important because the adjacency behavior is different for each network type, and OSPF must be properly configured to function correctly over certain network types.

### Types of OSPF Networks

OSPF defines distinct types of networks, based on their physical link type. OSPF operation on each type is different, including how adjacencies are established and the configuration required.

OSPF defines three types of networks:

- **Point-to-point**— A network that joins a single pair of routers.
- **Broadcast**— A multiaccess broadcast network, such as Ethernet.
- **Nonbroadcast multiaccess (NBMA)**— A network that interconnects more than two routers but that has no broadcast capability. Frame Relay, ATM, and X.25 are examples of NBMA networks. There are five modes of OSPF operation available for NBMA networks, as described later in the “OSPF over NBMA Topology Modes of Operation” section of this chapter.

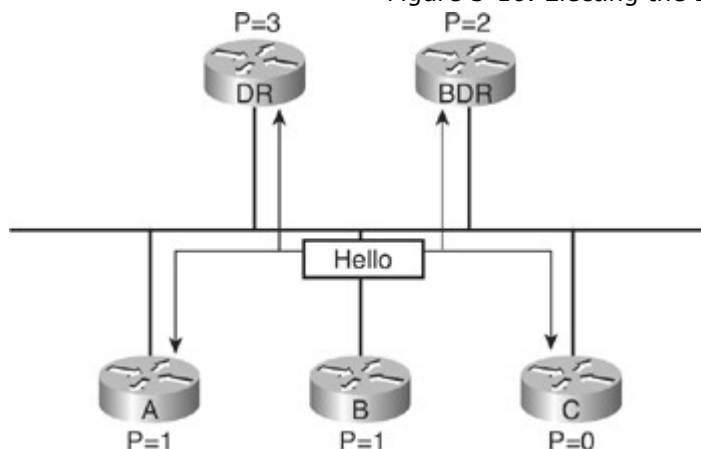
OSPF operation on each of these network types is described in the following sections. Because DR and BDR play a role in some network types, the election of the DR and BDR on these network types is also described first. OSPF operation over both Layer 2 and Layer 3 Multiprotocol Label Switching (MPLS) virtual private networks (VPNs) is also explored.

### Electing a DR and BDR and Setting Priority

To elect a DR and BDR, the routers view the OSPF priority value of the other routers during the hello packet exchange process and then use the following conditions to determine which router to select:

- The router with the highest priority value is the DR, as shown in Figure 3-16.

Figure 3-16. Electing the DR and BDR.



- The router with the second-highest priority value is the BDR.
- The default for the interface OSPF priority is 1. In case of a tie, the router ID is used. The router with the highest router ID becomes the DR. The router with the second-highest router ID becomes the BDR.
- A router with a priority of 0 cannot become the DR or BDR. A router that is not the DR or BDR is a DROTHER.
- If a router with a higher priority value gets added to the network, it does not preempt the DR and BDR. The only time a DR or BDR changes is if one of them goes out of service. If the DR is out of service, the BDR becomes the DR, and a new BDR is selected. If the BDR is out of service, a new BDR is elected.

The BDR does not perform any DR functions when the DR is operating. Instead, the BDR receives all the information, but the DR performs the LSA forwarding and LSDB synchronization tasks. The BDR performs the DR tasks only if the DR fails. To determine whether the DR is out of service, the BDR uses the wait timer. This timer is a reliability feature. If the BDR does not confirm that the DR is forwarding LSAs before the wait timer expires, the BDR assumes that the DR is out of service.

It is important to remember that the DR concept is at the link level. In a multiaccess broadcast environment each network segment has its own DR and BDR. For example, a router connected to multiple multiaccess broadcast networks can be a DR on one segment and a regular (DROTHER) router on another segment.

Use the **ip ospf priority** *number* interface configuration command to affect which router interfaces on a multiaccess link are the DR and the BDR. The default priority is 1, and the range is from 0 to 255. The highest priority interface becomes the DR, and the second-highest priority interface becomes the BDR. Any interfaces set to 0 priority cannot be involved in the DR or BDR election process.

Example 3-11 illustrates an example of configuring the Fast Ethernet interface on a router with an OSPF priority of 10.

Example 3-11. *ip ospf priority* Command

```
Router(config)#interface FastEthernet 0/0
Router(config-if)#ip ospf priority 10
```

#### Note

An interface's priority usually takes effect only when the existing DR goes down. A DR does not relinquish its status just because a new interface reports a higher priority in its hello packet.

Setting an interface's OSPF priority to 0—indicating that it should not be the DR or the BDR—takes effect immediately, however. A new election takes place, and the interface in question will not be elected for either the DR or BDR role.

#### Adjacency Behavior for a Point-to-Point Link

A point-to-point network joins a single pair of routers. A T1 serial line configured with a link-layer protocol such as PPP or HDLC is an example of a point-to-point network.

On point-to-point networks, the router dynamically detects its neighboring routers by multicasting its hello packets to all OSPF routers using the address 224.0.0.5. On point-to-point networks, neighboring routers become adjacent whenever they can communicate directly. No DR or BDR election is performed, because a point-to-point link can have only two routers, so there is no need for a DR or BDR.

Usually, the IP source address of an OSPF packet is set to the address of the outgoing interface on the router. If an unnumbered interface is used, the IP source address is set to the IP address of another interface on the router.

The default OSPF hello and dead intervals on point-to-point links are 10 seconds and 40 seconds, respectively. (The hello and dead timers can be changed with the **ip ospf hello-interval** *seconds* and **ip ospf dead-interval** *seconds* interface configuration commands.)

#### Adjacency Behavior for a Broadcast Network

An OSPF router on a multiaccess broadcast network such as Ethernet forms an adjacency with its DR and BDR. Adjacent routers have synchronized LSDBs. A common media segment is the basis for adjacency, such as two routers connected on the same Ethernet segment. When routers first come up on the segment, they perform the hello process and then elect the DR and BDR to represent the multiaccess broadcast network. The routers then attempt to form adjacencies with the DR and BDR.

The DR and BDR add value to the network in the following ways:

- **Reducing routing update traffic**— The DR and BDR act as a central point of contact for link-state information exchange on a given multiaccess broadcast network. Therefore, each router must establish a full adjacency with the DR and the BDR only. Instead of each router exchanging link-state information with every other router on the segment, each router sends the link-state information to the DR and BDR only. The DR represents the multiaccess broadcast network in the sense that it sends link-state information from each router to all other routers in the network. This flooding process significantly reduces the router-related traffic on a segment.
- **Managing link-state synchronization**— The DR and BDR ensure that the other routers on the network have the same link-state information about the internetwork. In this way, the DR and BDR reduce the number of routing errors.

Remember that after a DR and BDR have been selected, any router added to the broadcast network establishes full adjacencies with the DR and BDR only.

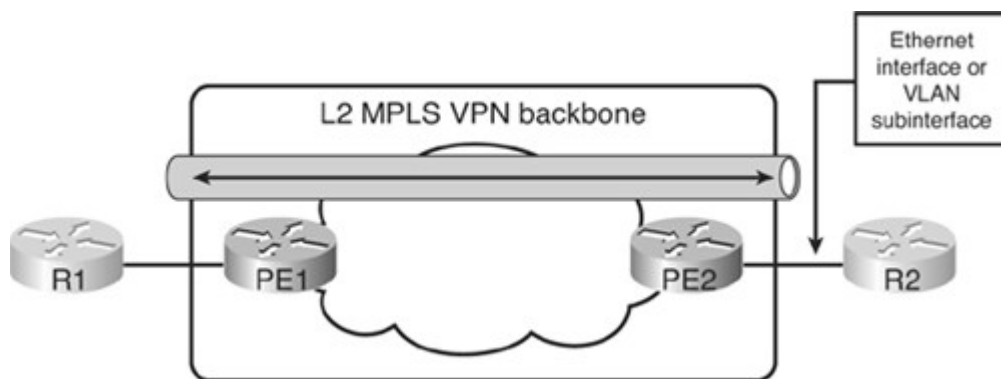
#### Adjacency Behavior over a Layer 2 MPLS VPN

As described in Chapter 2, "Configuring the Enhanced Interior Gateway Routing Protocol," Ethernet over MPLS (EoMPLS) is used to deploy Layer 2 MPLS VPNs, in which an MPLS backbone provides a Layer 2 Ethernet port-to-port connection, as illustrated in Figure 3-17. (EoMPLS is also known as a type of Metro Ethernet service.)

Figure 3-17. EoMPLS Provides Layer 2 MPLS VPN Connectivity.

[View full size image]





In Figure 3-17, Routers R1 and R2 are exchanging Ethernet frames transparently across the MPLS backbone. They are connected to provider edge (PE) routers. The PE1 router takes the Ethernet frame received from the directly connected router R1, encapsulates it into an MPLS packet and forwards it across the backbone to the PE2 router. The PE2 router decapsulates the MPLS packet and reproduces the Ethernet frame on its Ethernet link to router R2.

EoMPLS does not include any MAC layer address learning and filtering. Therefore routers PE1 and PE2 do not filter any frames based on MAC addresses. EoMPLS also does not use the Spanning Tree Protocol (STP). Bridge protocol data units (BPDUs) are propagated transparently and not processed, so LAN loop detection must be performed by other devices or avoided by design. A service provider can use LAN switches in conjunction with EoMPLS to provide these features.

The two customer routers R1 and R2 in Figure 3-17 could also be connected to the MPLS edge routers PE1 and PE2 via virtual LAN (VLAN) subinterfaces, using different subinterfaces in the PE1 and PE2 routers to connect to different VLANs. In this case, the PE1 subinterface to the VLAN where the router R1 is connected is used for EoMPLS forwarding. The Ethernet frame that arrives from router R1 on the specific VLAN subinterface is encapsulated into MPLS and forwarded across the backbone to router PE2. The PE2 router decapsulates the packet and reproduces the Ethernet frame on its outgoing VLAN subinterface to router R2.

When you are deploying OSPF over EoMPLS, there are no changes to the OSPF configuration from the customer perspective. For example, the OSPF process needs to be enabled, and **network** commands must be configured to include all the interfaces that will run in the OSPF process, in the appropriate OSPF areas. These interfaces include the link toward the PE routers (Routers PE1 and PE2) over which the Routers R1 and R2 will form their neighbor relationship.

From the OSPF perspective, the Layer 2 MPLS VPN backbone and the PE1 and PE2 routers are not visible. A neighbor relationship is established directly between Routers R1 and R2 over the MPLS backbone, and behaves in the same way as on an Ethernet broadcast network. The OSPF network type is a multiaccess broadcast network, so DR and BDR routers are elected. Other routers form full adjacencies only with the DR and BDR.

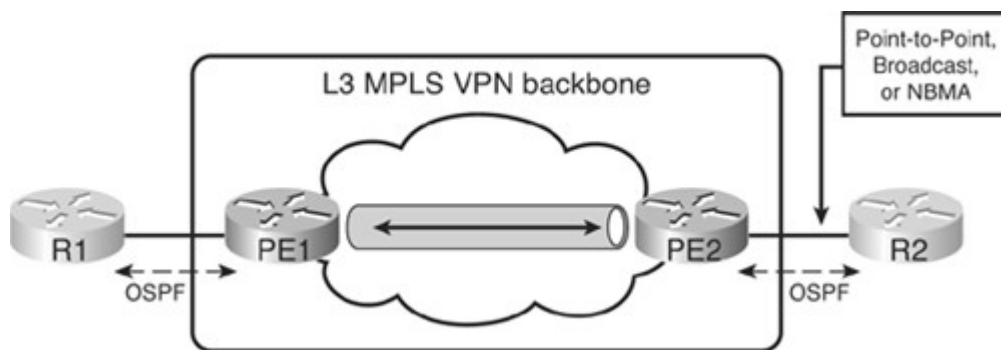
#### Adjacency Behavior over a Layer 3 MPLS VPN

As described in Chapter 2, a Layer 3 MPLS VPN architecture provides a peer-to-peer VPN where PE routers participate in customer routing, providing optimum routing between customer sites. Figure 3-18 illustrates such a network. The PE routers carry a separate set of routes for each customer, isolating customers from each other. In Layer 3 MPLS VPN technology (even when running OSPF as a PE-CE routing protocol), the following requirements must be met:

- The customer routers (also called the C-routers, R1 and R2 in the figure) should not be MPLS VPN-aware. They should run standard IP routing software.
- The provider core routers (also called the P-routers, within the cloud in the figure) must not carry customer (VPN) routes, to make the MPLS VPN solution scalable.
- The provider edge routers (the PE routers) must support MPLS VPN services and traditional IP services.

Figure 3-18. OSPF over Layer 3 MPLS VPN Connectivity.

[View full size image]



To the customer routers running OSPF (Routers R1 and R2 in Figure 3-18), the Layer 3 MPLS VPN backbone looks like a standard corporate backbone. The customer routers run standard IP routing software and exchange routing updates with the PE routers that appear to them as normal routers in the customer's network.

OSPF is enabled on these routers on proper interfaces using the **network** command. The customer has to agree upon OSPF parameters (such as authentication) with the service provider (SP) to ensure connectivity. These parameters are often governed by the SP.

The P-routers are hidden from the customer's routers, and the customer edge (CE) routers are unaware of MPLS VPN. The internal topology of the Layer 3 MPLS backbone is therefore totally transparent to the customer. The PE routers receive IPv4 routing updates from the CE routers and install the updates in the appropriate VRF table. This part of the configuration and operation is the SP's responsibility.

The OSPF network type of the CE-PE link can be point to point, broadcast or NBMA.

#### Adjacency Behavior for an NBMA Network

When a single interface interconnects multiple sites over an NBMA network, the network's nonbroadcast nature can create reachability issues. NBMA networks can support more than two routers, but they have no inherent broadcast capability. To implement broadcasting or multicasting, the router replicates the packets to be broadcast or multicast and sends them individually on each permanent virtual circuit (PVC) to all destinations. This process is CPU and bandwidth intensive. If the NBMA topology is not fully meshed, a broadcast or multicast sent by one router does not reach all the other routers. Frame Relay, ATM, and X.25 are examples of NBMA networks.

The default OSPF hello and dead intervals on NBMA interfaces are 30 seconds and 120 seconds, respectively.

The following sections detail issues with DR elections in an NBMA topology, OSPF topology options in a Frame Relay network, OSPF modes for NBMA topologies, and how to configure OSPF in each of these modes.

#### DR Election in an NBMA Topology

By default, OSPF cannot automatically build adjacencies with neighbor routers over NBMA interfaces.

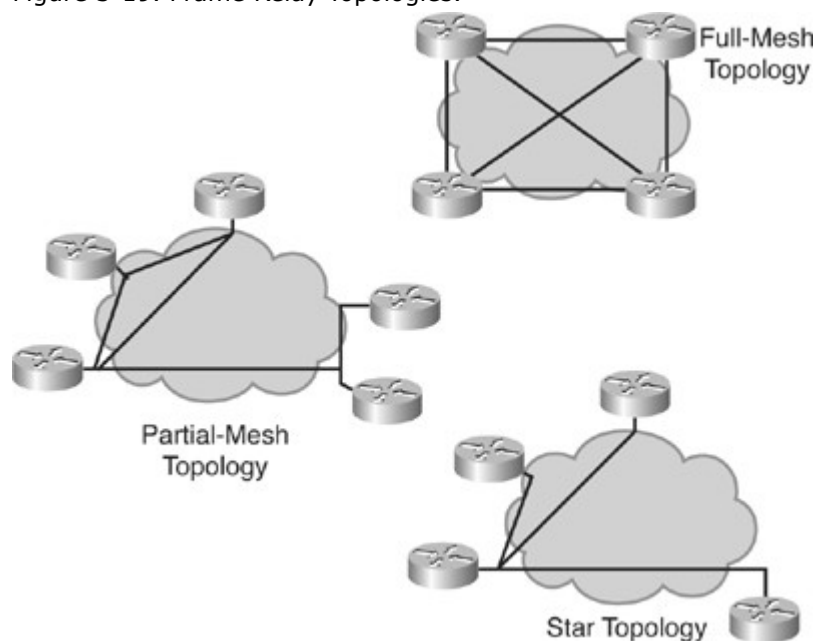
OSPF considers the NBMA environment to function similarly to other multiaccess media such as Ethernet. However, NBMA networks are usually hub-and-spoke (star) topologies using PVCs or switched virtual circuits (SVCs). In these cases, the physical topology does not provide the multiaccess capability on which OSPF relies.

The election of the DR becomes an issue in NBMA topologies because the DR and BDR need to have full Layer 2 connectivity with all routers in the NBMA network. The DR and BDR also need to have a list of all the other routers so that they can establish adjacencies.

#### OSPF over Frame Relay Topology Options

Depending on the network topology, several OSPF configuration choices are available for a Frame Relay network. By default, interfaces that support Frame Relay are multipoint connection types. With Frame Relay, remote sites interconnect in a variety of ways, as shown in Figure 3-19. The following examples are types of Frame Relay topologies:

Figure 3-19. Frame Relay Topologies.



- **Star topology**— A star topology, also known as a hub-and-spoke configuration, is the most common Frame Relay network topology. In this topology, remote sites connect to a central site that generally provides a service or application. The star topology is the least expensive topology because it requires the fewest PVCs. The central router provides a multipoint connection because it typically uses a single interface to interconnect multiple PVCs.
- **Full-mesh topology**— In a full-mesh topology, all routers have virtual circuits (VCs) to all other destinations. This method, although costly, provides direct connections from each site to all other sites and allows for redundancy. As the number of nodes in the full-mesh topology increases, the topology becomes increasingly expensive. To calculate how many VCs are needed to implement a full-mesh topology, use the formula  $n(n - 1)/2$ , where  $n$  is the number of nodes in the network.
- **Partial-mesh topology**— In a partial-mesh topology, not all sites have direct access to a central site. This method reduces the cost compared to implementing a full-mesh topology.

#### OSPF over NBMA Topology Modes of Operation

OSPF on Cisco routers can operate in various modes over NBMA networks. The choice of modes available depends on the topology of the NBMA network. Each mode has unique configuration requirements.

As described in RFC 2328, the following are the two official modes for OSPF in NBMA topologies:

- **Nonbroadcast**— The nonbroadcast (NBMA) mode simulates the operation of OSPF in broadcast networks. Neighbors must be configured manually, and DR and BDR election is required. This configuration is typically used with full-mesh networks.
- **Point-to-multipoint**— Point-to-multipoint mode treats the nonbroadcast network as a collection of point-to-point links. In this environment, the routers automatically identify their neighboring routers but do not elect a DR and BDR. This configuration is typically used with partial-mesh networks.

The choice between nonbroadcast and point-to-multipoint modes determines how the Hello protocol and flooding work over the nonbroadcast network.

The main advantage of point-to-multipoint mode is that it requires less manual configuration. The main advantage of nonbroadcast mode is that there is less overhead traffic compared to point-to-multipoint mode.

In addition, Cisco offers the following modes for OSPF operation in an NBMA network:

- Point-to-multipoint nonbroadcast
- Broadcast
- Point-to-point

#### Selecting the OSPF Network Type for NBMA Networks

Use the `ip ospf network {broadcast | non-broadcast | point-to-multipoint [non-broadcast] | point-to-point}` interface configuration command to select the OSPF mode. Table 3-8 describes the parameters of this command, and the modes, in greater detail.

Table 3-8. *ip ospf network* Command

| Command Options                                       | Description                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>broadcast</b> (Cisco mode)                         | Makes the WAN interface appear to be a LAN.<br>One IP subnet.<br>Uses a multicast OSPF hello packet to automatically discover the neighbors.<br>DR and BDR are elected.<br>Full- or partial-mesh topology.                                                                                                                                               |
| <b>non-broadcast</b> (RFC-compliant mode)             | One IP subnet.<br>Neighbors must be manually configured.<br>DR and BDR are elected.<br>DR and BDR need to have full connectivity with all other routers.<br>Typically used in a full- or partial-mesh topology.                                                                                                                                          |
| <b>point-to-multipoint</b> (RFC-compliant mode)       | One IP subnet.<br>Uses a multicast OSPF hello packet to automatically discover the neighbors.<br>DR and BDR are not required. The router sends additional LSAs with more information about neighboring routers.<br>Typically used in a partial-mesh or star topology.                                                                                    |
| <b>point-to-multipoint non-broadcast</b> (Cisco mode) | If multicast and broadcast are not enabled on the VCs, the RFC-compliant point-to-multipoint mode cannot be used, because the router cannot dynamically discover its neighboring routers using the hello multicast packets, so this Cisco mode should be used instead.<br>Neighbors must be manually configured.<br>DR and BDR election is not required. |
| <b>point-to-point</b> (Cisco mode)                    | Different IP subnet on each subinterface.<br>No DR or BDR election.<br>Used when only two routers need to form an adjacency on a pair of interfaces.<br>Interfaces can be either LAN or WAN.                                                                                                                                                             |

The default OSPF modes are as follows:

- The default OSPF mode on a point-to-point Frame Relay subinterface is the point-to-point mode.
- The default OSPF mode on a Frame Relay multipoint subinterface is the nonbroadcast mode.
- The default OSPF mode on a main Frame Relay interface is also the nonbroadcast mode.

The following sections detail the configuration of OSPF in each of these modes.

#### OSPF Configuration in Cisco Broadcast Mode

Broadcast mode is a workaround for statically listing all existing neighboring routers. The interface is set to broadcast and behaves as though the router connects to a LAN. DR and BDR election is still performed. Therefore, take special care to ensure either a full-mesh topology or a static election of the DR based on the interface priority to ensure that the selected DR and BDR have full connectivity to all other neighbor routers. Example 3-12 shows a sample configuration of a Frame Relay router in a full-mesh topology, with the broadcast mode of operation defined.

#### Example 3-12. Frame Relay Router in OSPF Broadcast Mode with Full-Mesh Topology

```
Router(config)#interface serial 0/0/0
Router(config-if)#encapsulation frame-relay
Router(config-if)#ip ospf network broadcast
```

## OSPF Nonbroadcast Mode Configuration

In nonbroadcast mode, OSPF emulates operation over a broadcast network. A DR and BDR are elected for the NBMA network, and the DR originates an LSA for the network. In this environment, the routers are usually fully meshed to facilitate the establishment of adjacencies among them. If the routers are not fully meshed, the DR and BDR should be selected manually to ensure that the selected DR and BDR have full connectivity to all other neighbor routers. Neighboring routers are statically defined to start the DR/BDR election process. When using nonbroadcast mode, all routers are on one IP subnet.

For flooding over a nonbroadcast interface, the LSU packet must be replicated for each PVC. The updates are sent to each of the interface's neighboring routers, as defined in the neighbor table.

When few neighbors exist in the network, nonbroadcast mode is the most efficient way to run OSPF over NBMA networks because it has less overhead than point-to-multipoint mode.

After you enable the OSPF process for specific interfaces, you configure nonbroadcast mode by

- Manually configuring OSPF neighbors
- Defining the OSPF network type as nonbroadcast (unless it is the default)

Use the neighbor ip-address [priority number] [poll-interval number] [cost number] [database-filter all] router configuration command to statically define adjacent relationships in NBMA networks using the nonbroadcast mode. Table 3-9 describes the parameters of this command in detail.

Table 3-9. *neighbor* Command

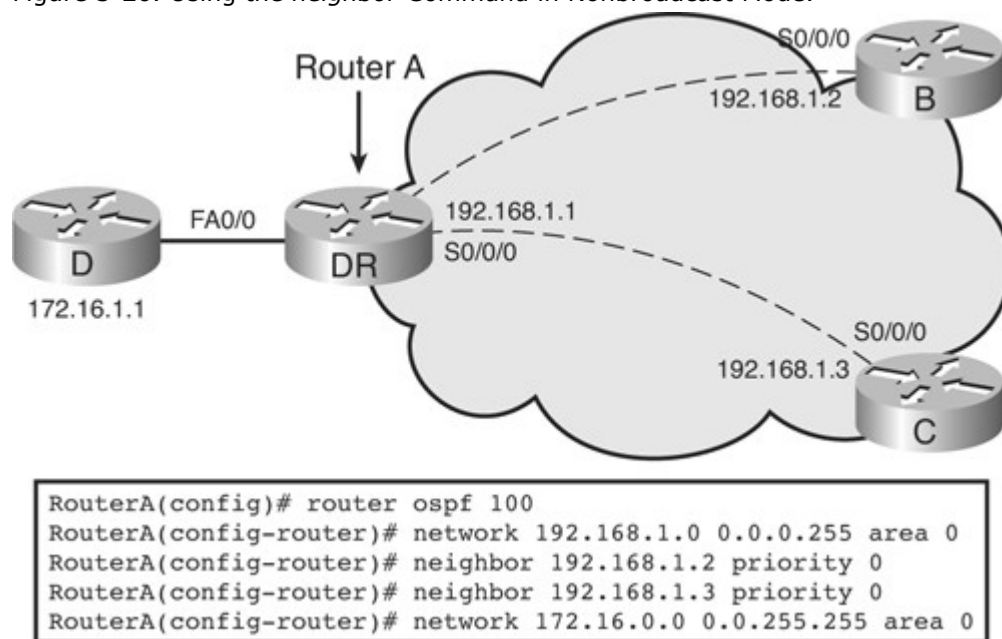
| Parameter                          | Description                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ip-address                         | Specifies the IP address of the neighboring router.                                                                                                                                                                                                                                                                                                                                        |
| <b>priority</b> <i>number</i>      | (Optional) Specifies priority of neighbor. The default is 0, which means that the neighboring router does not become the DR or BDR.                                                                                                                                                                                                                                                        |
| <b>poll-interval</b> <i>number</i> | (Optional) Specifies how long an NBMA interface waits before sending hellos to the neighbors even if the neighbor is inactive. The poll interval is defined in seconds.                                                                                                                                                                                                                    |
| <b>cost</b> <i>number</i>          | (Optional) Assigns a cost to the neighbor in the form of an integer from 1 to 65535. Neighbors with no specific cost configured assume the cost of the interface based on the <b>ip ospf cost</b> command. For point-to-multipoint interfaces, the <b>cost</b> keyword and the number argument are the only options that are applicable. This keyword does not apply to nonbroadcast mode. |
| database-filter all                | (Optional) Filters outgoing LSAs to an OSPF neighbor.                                                                                                                                                                                                                                                                                                                                      |

### Note

You cannot use the OSPF **neighbor** command to specify an OSPF neighbor on nonbroadcast networks within an OSPF VPN routing instance.

Figure 3-20 shows an example of statically defining adjacencies. All three routers are using the default nonbroadcast mode on their Frame Relay interfaces, so neighboring routers must be manually configured on each. The priority parameter is set to 0 for Routers B and C to ensure that Router A becomes the DR. Only Router A has full connectivity to the other two routers because the topology is not a full-mesh. No BDR will be elected in this case.

Figure 3-20. Using the *neighbor* Command in Nonbroadcast Mode.



#### Note

The default priority on the **neighbor** command is supposed to be 0. However, during testing it was noted that configuring the priority in this way for nonbroadcast mode interfaces actually resulted in a priority of 1, not 0. Setting the OSPF priority to 0 at the interface level (with the **ip ospf priority** command) on Routers B and C did result in a priority of 0 and the routers not being elected as DR or BDR.

In nonbroadcast mode, **neighbor** statements are required only on the DR and BDR. In a hub-and-spoke topology, **neighbor** statements must be placed on the hub, which must be configured to become the DR by being assigned a higher priority. **Neighbor** statements are not mandatory on the spoke routers. In a full-mesh NBMA topology, you might need **neighbor** statements on all routers unless the DR and BDR are statically configured using the **ip ospf priority** command.

The `show ip ospf neighbor` command displays OSPF neighbor information on a per-interface basis, as described earlier in the “Verifying OSPF Operations” section. Example 3-13 demonstrates sample output from the `show ip ospf neighbor` command for Router A in Figure 3-20 (with the OSPF priorities of Routers B and C set to 0 at the interface level, as noted). Router A has a serial Frame Relay interface and a Fast Ethernet interface. The Serial 0/0/0 interface on this router has two neighbors; both have a state of FULL/DROTHER. DROTHER means that the neighboring router is not a DR or BDR (because Router A is the DR and there is no BDR in this network). The neighbor learned on Fast Ethernet 0/0 has a state of FULL/BDR, which means that it has successfully exchanged LSDB information with the router issuing the show command, and that it is the BDR.

Example 3-13. `show ip ospf neighbor` Output for Router A in Figure 3-20

```
RouterA#show ip ospf neighbor
```

| Neighbor ID | Pri | State        | Dead Time | Address     | Interface       |
|-------------|-----|--------------|-----------|-------------|-----------------|
| 192.168.1.3 | 0   | FULL/DROTHER | 00:01:57  | 192.168.1.3 | Serial0/0/0     |
| 192.168.1.2 | 0   | FULL/DROTHER | 00:01:33  | 192.168.1.2 | Serial0/0/0     |
| 172.16.1.1  | 1   | FULL/BDR     | 00:00:34  | 172.16.1.1  | FastEthernet0/0 |

#### OSPF Configuration in Point-to-Multipoint Mode

Networks in RFC 2328-compliant point-to-multipoint mode are designed to work with partial-mesh or star topologies. In point-to-multipoint mode, OSPF treats all router-to-router connections over the nonbroadcast network as if they are point-to-point links.



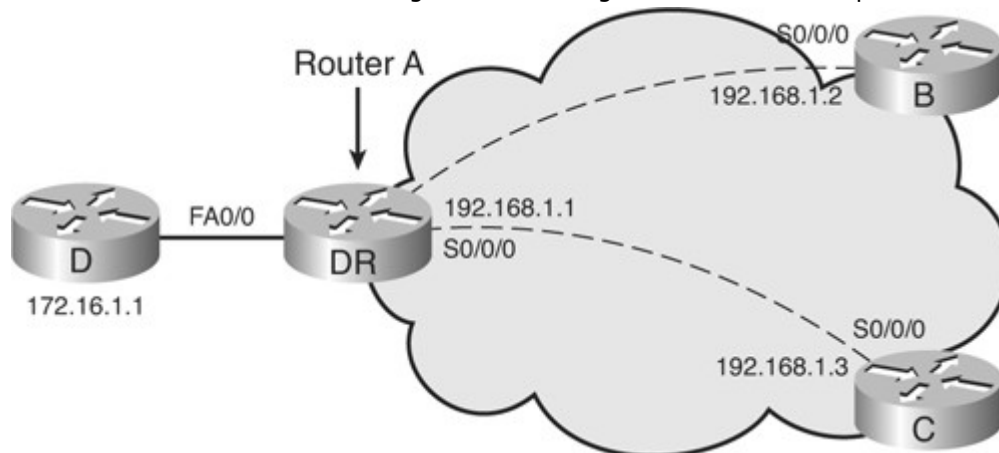
In point-to-multipoint mode, DRs are not used, and a type 2 network LSA is not flooded to adjacent routers. (The “Understanding OSPF LSAs” section, later in this chapter, covers the different types of LSAs in greater detail.) Instead, OSPF point-to-multipoint works by exchanging additional LSUs that are designed to automatically discover neighboring routers and add them to the neighbor table.

In large networks, using point-to-multipoint mode reduces the number of PVCs required for complete connectivity, because you are not required to have a full-mesh topology. In addition, not having a full-mesh topology reduces the number of neighbor entries in the neighborship table.

Point-to-multipoint mode has the following properties:

- **Does not require a full-mesh network**— This environment allows routing to occur between two routers that are not directly connected but that are connected through a router that has VCs to each of the two routers. All three routers connected to the Frame Relay network in Figure 3-21 could be configured for point-to-multipoint mode.

Figure 3-21. Using OSPF Point-to-Multipoint Mode.



- **Does not require a static neighbor configuration**— In nonbroadcast mode, neighboring routers are statically defined to start the DR election process, and allow the exchange of routing updates. However, because point-to-multipoint mode treats the network as a collection of point-to-point links, multicast hello packets discover neighboring routers dynamically. Statically configuring neighboring routers is not necessary.
- **Uses one IP subnet**— As in nonbroadcast mode, when using point-to-multipoint mode, all routers are on one IP subnet.
- **Duplicates LSA packets**— Also as in nonbroadcast mode, when flooding out a nonbroadcast interface in point-to-multipoint mode, the router must replicate the LSU. The LSU packet is sent to each of the interface’s neighboring routers, as defined in the neighbor table.

Example 3-14 shows partial configurations of routers A and C in Figure 3-21 in point-to-multipoint mode. This configuration does not require subinterfaces and uses only a single subnet. In point-to-multipoint mode, a DR or BDR is not required, so DR and BDR election and priorities are not a concern.

Example 3-14. Point-to-Multipoint Configuration for Routers A and C in Figure 3-21

```
RouterA(config)#interface Serial0/0/0
RouterA(config-if)#ip address 192.168.1.1 255.255.255.0
RouterA(config-if)#encapsulation frame-relay
RouterA(config-if)#ip ospf network point-to-multipoint
<output omitted>
RouterA(config)#router ospf 100
RouterA(config-router)#log-adjacency-changes
RouterA(config-router)#network 172.16.0.0 0.0.255.255 area 0
RouterA(config-router)#network 192.168.1.0 0.0.0.255 area 0

RouterC(config)#interface Serial0/0/0
RouterC(config-if)#ip address 192.168.1.3 255.255.255.0
```



```
RouterC(config-if)#encapsulation frame-relay
RouterC(config-if)#ip ospf network point-to-multipoint
<output omitted>
RouterC(config)#router ospf 100
RouterC(config-router)#log-adjacency-changes
RouterC(config-router)#network 192.168.1.0 0.0.0.255 area 0
```

Example 3-15 demonstrates output from the show ip ospf interface command, which displays key OSPF details for each interface. This sample output is from Router A in Figure 3-21.

Example 3-15. show ip ospf interface Output from Router A in Figure 3-21

```
Code View: Scroll / Show All
RouterA#show ip ospf interface s0/0/0
Serial0/0/0 is up, line protocol is up
 Internet Address 192.168.1.1/24, Area 0
 Process ID 100, Router ID 192.168.1.1, Network Type POINT_TO_MULTIPOINT, Cost:
781
 Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT
 Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
 oob-resync timeout 120
 Hello due in 00:00:26
 Supports Link-local Signaling (LLS)
 Index 2/2, flood queue length 0
 Next 0x0(0)/0x0(0)
 Last flood scan length is 1, maximum is 1
 Last flood scan time is 0 msec, maximum is 4 msec
 Neighbor Count is 2, Adjacent neighbor count is 2
 Adjacent with neighbor 192.168.1.3
 Adjacent with neighbor 192.168.1.2
 Suppress hello for 0 neighbor(s)
RouterA#
```

The OSPF network type, area number, cost, and state of the interface are all displayed. The hello interval for a point-to-multipoint interface is 30 seconds and the dead interval is 120 seconds. Point-to-multipoint mode, point-to-multipoint nonbroadcast mode, and nonbroadcast mode default to a 30-second hello timer, while point-to-point mode and broadcast mode default to a 10-second hello timer. Recall that the hello and dead timers on the neighboring interfaces must match for the neighbors to form successful adjacencies.

The listed adjacent neighboring routers are all dynamically learned.

OSPF Configuration in Cisco Point-to-Multipoint Nonbroadcast Mode

Point-to-multipoint nonbroadcast mode is a Cisco extension of the RFC-compliant point-to-multipoint mode. You must statically define neighbors, and you can modify the cost of the link to the neighboring router to reflect the different bandwidths of each link. The RFC point-to-multipoint mode was developed to support underlying point-to-multipoint VCs that support multicast and broadcast. If multicast and broadcast are not enabled on the VCs, RFC-compliant point-to-multipoint mode cannot be used because the router cannot dynamically discover its neighboring routers using the hello multicast packets. In this case, the Cisco point-to-multipoint nonbroadcast mode should be used instead.

In point-to-multipoint nonbroadcast mode, you must statically define neighbors, like in nonbroadcast mode. As in point-to-multipoint mode, DRs and BDRs are not elected.

Using Subinterfaces in OSPF over Frame Relay Configuration

OSPF can also be run over subinterfaces. This section first describes subinterfaces and how to configure them, and then describes OSPF operation over subinterfaces.

A physical interface can be split into multiple logical interfaces called subinterfaces. Subinterfaces were originally created to better handle issues caused by split horizon over NBMA for distance vector-based routing protocols. Each subinterface requires an IP subnet.

Each subinterface is defined as either a point-to-point or multipoint interface. A point-to-point subinterface has similar properties to a physical point-to-point interface.

Define subinterfaces using the interface serial.number.subinterface-number {multipoint | point-to-point} global configuration command. Table 3-10 lists the parameters of this command.

Table 3-10. *interface serial* Command Parameters

| Parameter                  | Description                                                                                                                                                                                                                       |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| number.subinterface-number | Specifies the interface number and subinterface number. The subinterface number is in the range of 1 to 4294967293. The interface number that precedes the period (.) is the interface number to which this subinterface belongs. |
| multipoint                 | Specifies that the subinterface is multipoint. On multipoint subinterfaces routing IP, all routers are in the same subnet.                                                                                                        |
| point-to-point             | Specifies that the subinterface is point-to-point. On point-to-point subinterfaces routing IP, each pair of point-to-point routers is in its own subnet.                                                                          |

The choice of **point-to-point** or **multipoint** keywords affects the OSPF operation, as described in the sections that follow.

It is important to remember that OSPF defaults to point-to-point mode on point-to-point subinterfaces, and to nonbroadcast mode on the multipoint subinterfaces.

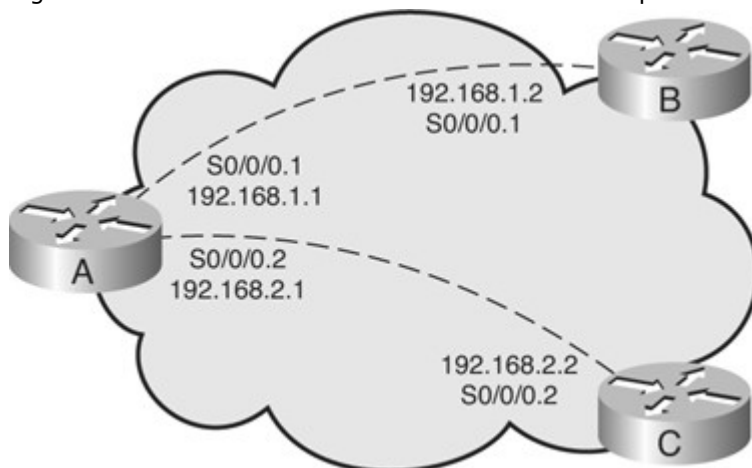
#### OSPF over Point-to-Point Subinterfaces

When point-to-point subinterfaces are configured, each virtual circuit (PVC and SVC) gets its own subinterface. A point-to-point subinterface has the properties of any physical point-to-point interface: there is no DR or BDR and neighbor discovery is automatic, so neighbors do not need to be configured.

Point-to-point mode is used when only two nodes exist on the NBMA network. This mode is typically used only with point-to-point subinterfaces. Each point-to-point connection is one IP subnet. An adjacency forms over the point-to-point network with no DR or BDR election.

In the example in Figure 3-22 and the configurations in Example 3-16, Router A's serial 0/0/0 interface is configured with point-to-point subinterfaces. Although all three routers on the Frame Relay network have only one physical serial port, Router A appears to have two logical ports. Each logical port (subinterface) has its own IP address and operates in OSPF point-to-point mode. Each subinterface is on its own IP subnet. Router B's configuration is also presented in Example 3-16.

Figure 3-22. OSPF Point-to-Point Subinterface Example.



Example 3-16. Point-to-Point Subinterface Configuration for Routers A and B in Figure 3-22

```
RouterA#
interface Serial0/0/0
 no ip address
 encapsulation frame-relay
 !
interface Serial0/0/0.1 point-to-point
 ip address 192.168.1.1 255.255.255.0
 frame-relay interface-dlci 121
interface Serial0/0/0.2 point-to-point
 ip address 192.168.2.1 255.255.255.0
 frame-relay interface-dlci 132

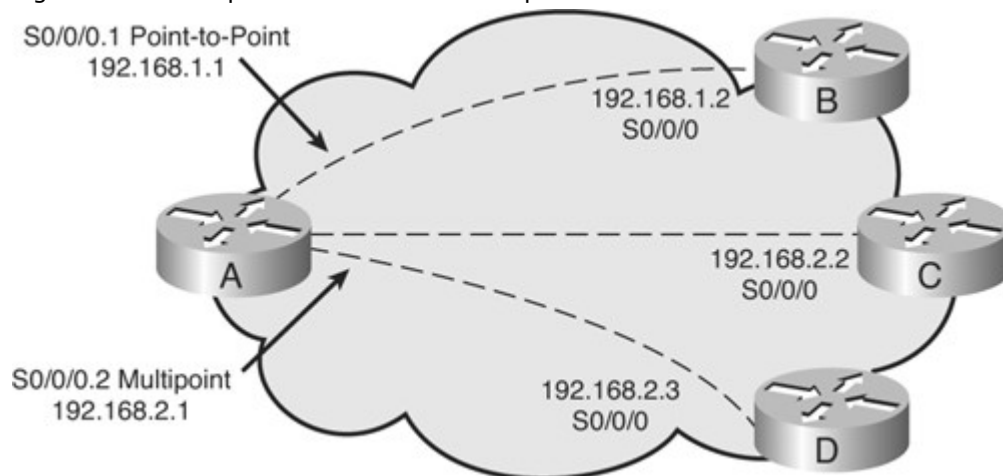
RouterB#
interface Serial0/0/0
 no ip address
 encapsulation frame-relay
 !
interface Serial0/0/0.1 point-to-point
 ip address 192.168.1.2 255.255.255.0
 frame-relay interface-dlci 122
```

#### OSPF over Multipoint Subinterfaces

When multipoint subinterfaces are configured, multiple virtual circuits (PVCs or SVCs) exist on a single subinterface. Multipoint Frame Relay subinterfaces default to OSPF nonbroadcast mode, which requires neighbors to be statically configured and a DR and BDR election.

In the example in Figure 3-23 and partial configuration in Example 3-17, Router A has one point-to-point subinterface and one multipoint subinterface. The multipoint subinterface supports two other routers in a single subnet. The configuration for Routers B and C are also presented in Example 3-17. Router D's configuration is similar to Router C's configuration.

Figure 3-23. Multipoint Subinterface Example.



Example 3-17. Point-to-Point Subinterface Configuration for Routers A, B, and C in Figure 3-23

```
Code View: Scroll / Show All
RouterA#
interface Serial0/0/0.1 point-to-point
```

```
ip address 192.168.1.1 255.255.255.0
<output omitted>
interface Serial0/0/0.2 multipoint
ip address 192.168.2.1 255.255.255.0
<output omitted>
router ospf 100
network 192.168.0.0 0.0.255.255 area 0
neighbor 192.168.2.2 priority 0
neighbor 192.168.2.3 priority 0
```

```
RouterB#
interface Serial0/0/0
ip address 192.168.1.2 255.255.255.0
<output omitted>
```

```
RouterC#
interface Serial0/0/0
ip address 192.168.2.2 255.255.255.0
ip ospf priority 3
<output omitted>
router ospf 100
network 192.168.0.0 0.0.255.255 area 0
```

#### OSPF Configuration in Cisco Point-to-Point Mode

Although the Cisco point-to-point mode is typically used only with point-to-point subinterfaces, it can also be used when a single PVC exists on a serial interface and point-to-point behavior is desired. In this mode, the PVC emulates a leased line. Because they are treated as point-to-point links, DR and BDR are not used, and only a single subnet is used on each point-to-point link.

#### Note

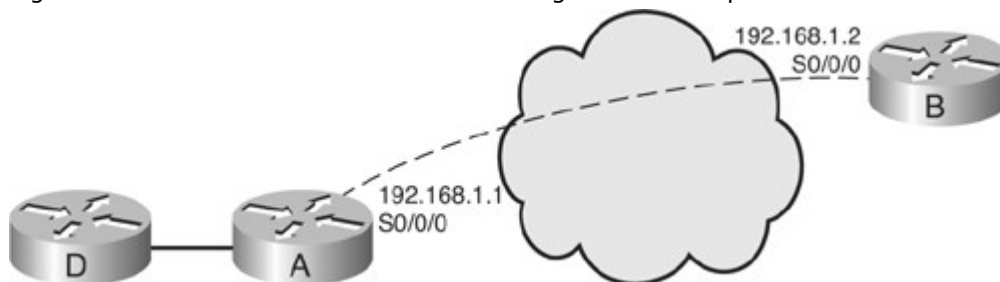
The Cisco point-to-point mode may also be used on other interface types. For example, when the **ip ospf network point-to-point** command is used on point-to-point Ethernet links, no DR or BDR is elected.

#### Note

Recall that, by default, OSPF advertises loopback interface addresses as /32 host routes. If the **ip ospf network point-to-point** command is configured on a loopback interface, OSPF advertises the actual loopback subnet mask, instead of a /32 host route.

Figure 3-24 and the partial configurations of routers A and B in Example 3-18 illustrate the use of point-to-point mode. Subinterfaces are not used, and only a single subnet is used. Because the default OSPF mode on a Frame Relay main interface is nonbroadcast mode, it is necessary to configure the **ip ospf network point-to-point** command.

Figure 3-24. Network for Point-to-Point Configuration Example.



Example 3-18. Point-to-Point Configuration for Routers A and B in Figure 3-24

```
RouterA#
interface Serial0/0/0
ip address 192.168.1.1 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-point
<output omitted>
router ospf 100
network 172.16.0.0 0.0.255.255 area 0
network 192.168.0.0 0.0.255.255 area 0

RouterB#
interface Serial0/0/0
ip address 192.168.1.2 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-point
```

#### OSPF over NBMA Modes of Operation Summary

Table 3-11 briefly compares the different modes of operation for OSPF.

Table 3-11. OSPF Mode Summary

| OSPF Mode                        | NBMA Preferred Topology     | Subnet Address     | Hello Timer | Adjacency                           | RFC or Cisco | Example                                                                                                            |
|----------------------------------|-----------------------------|--------------------|-------------|-------------------------------------|--------------|--------------------------------------------------------------------------------------------------------------------|
| Broadcast                        | Full or partial mesh        | Same               | 10 sec      | Automatic DR/BDR elected            | Cisco        | LAN interface such as Ethernet                                                                                     |
| Nonbroadcast (NBMA)              | Full or partial mesh        | Same               | 30 sec      | Manual configuration DR/BDR elected | RFC          | Frame Relay configured on a serial interface                                                                       |
| Point-to-multipoint              | Partial mesh or star        | Same               | 30 sec      | Automatic No DR/BDR                 | RFC          | OSPF over Frame Relay mode that eliminates the need for a DR; used when VCs support multicast and broadcast        |
| Point-to-multipoint nonbroadcast | Partial mesh or star        | Same               | 30 sec      | Manual configuration No DR/BDR      | Cisco        | OSPF over Frame Relay mode that eliminates the need for a DR; used when VCs do not support multicast and broadcast |
| Point-to-point                   | Partial mesh or star, using | Different for each | 10 sec      | Automatic No DR/BDR                 | Cisco        | Serial interface with point-to-                                                                                    |

| OSPF Mode | NBMA<br>Preferred<br>Topology | Subnet<br>Address | Hello Timer | Adjacency | RFC or<br>Cisco | Example                |
|-----------|-------------------------------|-------------------|-------------|-----------|-----------------|------------------------|
|           | subinterfaces                 | subinterface      |             |           |                 | point<br>subinterfaces |

### Displaying OSPF Adjacency Activity

Use the **debug ip ospf adj** command to track OSPF adjacencies as they go up or down. Debugging enables you to see exactly which OSPF packets are being sent between routers. The ability to see packets as they are sent over a link is an invaluable tool to the troubleshooter.

#### Note

The last parameter in this command is **adj**, not **adjacency**.

The debug output in Example 3-19 illustrates the activity on a serial interface in point-to-point mode. No DR/BDR election occurs. However, the adjacency forms, allowing DBD packets to be sent during the exchange process. Notice that the neighbor relationship passes through the two-way phase and into the exchange phase. After database description packets are sent between routers, the neighbors move into the final state, FULL adjacency.

Example 3-19. *debug ip ospf adj* Command Output for a Point-to-Point Serial Link

```
Code View: Scroll / Show All
RouterA# debug ip ospf adj
OSPF: Interface Serial0/0/0.1 going Up
OSPF: Build router LSA for area 0, router ID 192.168.1.1, seq 0x80000023
OSPF: Rcv DBD from 192.168.1.2 on Serial0/0/0.1 seq 0xCF0 opt 0x52 flag 0x7 len
32 mtu 1500 state INIT
OSPF: 2 Way Communication to 192.168.1.2 on Serial0/0/0.1, state 2WAY
OSPF: Send DBD to 192.168.1.2 on Serial0/0/0.1 seq 0xF4D opt 0x52 flag 0x7 len
32
OSPF: NBR Negotiation Done. We are the SLAVE
OSPF: Send DBD to 192.168.1.2 on Serial0/0/0.1 seq 0xCF0 opt 0x52 flag 0x2 len
132
OSPF: Rcv DBD from 192.168.1.2 on Serial0/0/0.1 seq 0xCF1 opt 0x52 flag 0x3 len
132 mtu 1500 state EXCHANGE
OSPF: Send DBD to 192.168.1.2 on Serial0/0/0.1 seq 0xCF1 opt 0x52 flag 0x0 len
32
OSPF: Database request to 192.168.1.2
OSPF: sent LS REQ packet to 192.168.1.2, length 12
OSPF: Rcv DBD from 192.168.1.2 on Serial0/0/0.1 seq 0xCF2 opt 0x52 flag 0x1 len
32 mtu 1500 state EXCHANGE
OSPF: Exchange Done with 192.168.1.2 on Serial0/0/0.1
OSPF: Send DBD to 192.168.1.2 on Serial0/0/0.1 seq 0xCF2 opt 0x52 flag 0x0 len
32
OSPF: Synchronized with 192.168.1.2 on Serial0/0/0.1, state FULL
%OSPF-5-ADJCHG: Process 100, Nbr 192.168.1.2 on Serial0/0/0.1 from LOADING to
FULL, Loading Done
OSPF: Build router LSA for area 0, router ID 192.168.1.1, seq 0x80000024
```

The debug output in Example 3-20 demonstrates the DR/BDR election process on a Fast Ethernet interface. The OSPF default behavior on a Fast Ethernet link is broadcast mode. First, the DR and BDR are selected, and then the exchange process occurs.

Example 3-20. *debug ip ospf adj* Command Output for a Fast Ethernet Link

```
Code View: Scroll / Show All
RouterA#debug ip ospf adj
OSPF: Interface FastEthernet0/0 going Up
OSPF: Build router LSA for area 0, router ID 192.168.1.1,seq 0x80000008
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
OSPF: 2 Way Communication to 172.16.1.1 on FastEthernet0/0, state 2WAY
OSPF: end of Wait on interface FastEthernet0/0
OSPF: DR/BDR election on FastEthernet0/0
OSPF: Elect BDR 192.168.1.1
OSPF: Elect DR 192.168.1.1
OSPF: Elect BDR 172.16.1.1
OSPF: Elect DR 192.168.1.1
DR: 192.168.1.1 (Id) BDR: 172.16.1.1 (Id)
OSPF: Send DBD to 172.16.1.1 on FastEthernet0/0 seq 0xDCE opt 0x52 flag 0x7 len 32
OSPF: No full nbrs to build Net Lsa for interface FastEthernet0/0
OSPF: Neighbor change Event on interface FastEthernet0/0
OSPF: DR/BDR election on FastEthernet0/0
OSPF: Elect BDR 172.16.1.1
OSPF: Elect DR 192.168.1.1
DR: 192.168.1.1 (Id) BDR: 172.16.1.1 (Id)
OSPF: Neighbor change Event on interface FastEthernet0/0
OSPF: DR/BDR election on FastEthernet0/0
OSPF: Elect BDR 172.16.1.1
OSPF: Elect DR 192.168.1.1
DR: 192.168.1.1 (Id) BDR: 172.16.1.1 (Id)
OSPF: Rcv DBD from 172.16.1.1 on FastEthernet0/0 seq 0x14B 7 opt 0x52 flag 0x7
len 32 mtu 1500 state EXSTART
OSPF: First DBD and we are not SLAVE-if)#
OSPF: Send DBD to 172.16.1.1 on FastEthernet0/0 seq 0xDCE opt 0x52 flag 0x7 len 32
OSPF: Retransmitting DBD to 172.16.1.1 on FastEthernet0/0[1]
OSPF: Rcv DBD from 172.16.1.1 on FastEthernet0/0 seq 0xDCE opt 0x52 flag 0x2 len
152 mtu 1500 state EXSTART
OSPF: NBR Negotiation Done. We are the MASTER
OSPF: Send DBD to 172.16.1.1 on FastEthernet0/0 seq 0xDCf opt 0x52 flag 0x3 len 132
OSPF: Database request to 172.16.1.1
OSPF: sent LS REQ packet to 172.16.1.1, length 24
OSPF: Rcv DBD from 172.16.1.1 on FastEthernet0/0 seq 0xDCf opt 0x52 flag 0x0 len
32 mtu 1500 state EXCHANGE
OSPF: Send DBD to 172.16.1.1 on FastEthernet0/0 seq 0xDD0
opt 0x52 flag 0x1 len 32
OSPF: No full nbrs to build Net Lsa for interface FastEthernet0/0
OSPF: Build network LSA for FastEthernet0/0, router ID 192.168.1.1
OSPF: Build network LSA for FastEthernet0/0, router ID 192.168.1.1
OSPF: Rcv DBD from 172.16.1.1 on FastEthernet0/0 seq 0xDD0 opt 0x52 flag 0x0 len
32 mtu 1500 state EXCHANGE
OSPF: Exchange Done with 172.16.1.1 on FastEthernet0/0
OSPF: Synchronized with 172.16.1.1 on FastEthernet0/0, state FULL
```



```
%OSPF-5-ADJCHG: Process 100, Nbr 172.16.1.1 on FastEthernet0/0 from LOADING to
FULL, Loading Done
OSPF: Build router LSA for area 0, router ID 192.168.1.1, seq 0x80000009
OSPF: Build network LSA for FastEthernet0/0, router ID 192.168.1.1Router(config-
if)# ip virtual-reassembly
OSPF: Build network LSA for FastEthernet0/0, router ID 192.168.1.1
```

## Understanding OSPF LSAs

This section describes each of the common LSA types and how they form the layout of the OSPF LSDB. LSAs are the building blocks of the OSPF LSDB. Table 3-12 summarizes the types of LSAs.

Table 3-12. Summary of OSPF LSA Types

| LSA Type     | Description                                               |
|--------------|-----------------------------------------------------------|
| 1            | Router LSA                                                |
| 2            | Network LSA                                               |
| 3 and 4      | Summary LSAs                                              |
| 5            | Autonomous system external LSA                            |
| 6            | Multicast OSPF LSA                                        |
| 7            | Defined for NSSAs                                         |
| 8            | External attributes LSA for Border Gateway Protocol (BGP) |
| 9, 10, or 11 | Opaque LSAs                                               |

Individually, LSAs act as database records. In combination, they describe the entire topology of an OSPF network or area. The following are descriptions of each type of LSA:

- **Type 1 (Router LSA)**— Every router generates router-link advertisements for each area to which it belongs. Router-link advertisements describe the states of the router's links to the area and are flooded only within a particular area.  
All types of LSAs have 20-byte LSA headers. One of the fields of the LSA header is the *link-state ID*. The link-state ID of the type 1 LSA is the originating router's ID.
- **Type 2 (Network LSA)**— DRs generate network link advertisements for multiaccess networks, which describe the set of routers attached to a particular multiaccess network. Network link advertisements are flooded in the area that contains the network. The link-state ID of the type 2 LSA is the DR's IP interface address.
- **Types 3 and 4 (Summary LSA)**— ABRs generate summary link advertisements. Summary link advertisements describe the following interarea routes:
  - Type 3 describes routes to the area's networks (and may include aggregate routes).
  - Type 4 describes routes to ASBRs.

The link-state ID is the destination network number for type 3 LSAs and the router ID of the described ASBR for type 4 LSAs.

These LSAs are flooded throughout the backbone area to the other ABRs. Type 3 LSAs are not flooded into totally stubby areas or totally stubby NSSAs. Type 4 LSAs are not flooded into any type of stub area. (Stub areas and NSSAs are discussed later in this chapter in the "Configuring OSPF Special Area Types" section.)

- **Type 5 (autonomous system external LSA)**— ASBRs generate autonomous system external link advertisements. External link advertisements describe routes to destinations external to the

autonomous system and are flooded everywhere except to any type of stub areas. The link-state ID of the type 5 LSA is the external network number.

- **Type 6 (Multicast OSPF LSA)**— These LSAs are used in multicast OSPF applications.
- **Type 7 (LSAs for NSSAs)**— These LSAs are used in NSSAs, as described in the “Configuring NSSAs” section, later in this chapter.
- **Type 8 (External attributes LSA for BGP)**— These LSAs are used to interconnect OSPF and BGP.
- **Types 9, 10, or 11 (Opaque LSAs)**— These LSA types are designated for future upgrades to OSPF for distributing application-specific information through an OSPF domain. For example, Cisco Systems uses Type 10 opaque LSAs for MPLS Traffic Engineering functionality with OSPF. Standard LSDB flooding mechanisms are used to distribute opaque LSAs. Each of the three types has a different flooding scope. Type 9 LSAs are not flooded beyond the local network or subnetwork. Type 10 LSAs are not flooded beyond the borders of their associated area. Type 11 LSAs are flooded throughout the autonomous system (the same as for Type 5 LSAs). (Opaque LSAs are defined in RFC 5250, *The OSPF Opaque LSA Option*.)

Note

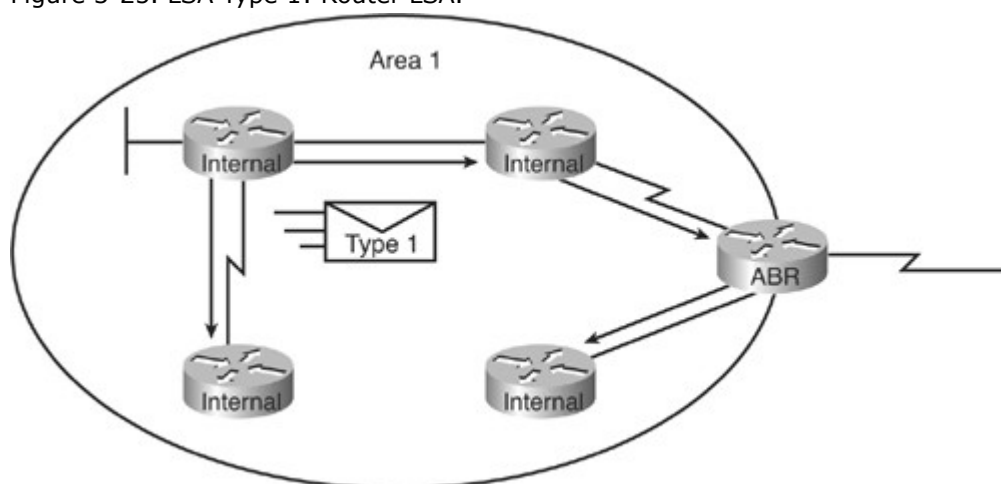
Type 6 and type 8 LSAs are not supported for OSPFv2 by Cisco.

The following sections describe the LSA types 1 to 5. (As previously mentioned, type 7 LSAs are described later in the chapter. The other types are not discussed further in this book.)

#### LSA Type 1: Router LSA

A router advertises a type 1 LSA that floods to all other routers in the area where it originated, as shown in Figure 3-25. A type 1 LSA describes the collective states of the router’s directly connected links (interfaces).

Figure 3-25. LSA Type 1: Router LSA.



Each type 1 LSA is identified by the originating router’s ID in the link-state ID field.

Each of the router’s links (interfaces) is defined as one of four types: type 1, 2, 3, or 4. The LSA includes a link ID field to identify what is on the other end of the link. Depending on the link type, the link ID field has different meanings. Type 1 LSA link types and their link ID meanings are described in Table 3-13.

Table 3-13. LSA Type 1 (Router LSA) Link Types and Link ID

| Link Type | Description                                 | Link ID Field Contents   |
|-----------|---------------------------------------------|--------------------------|
| 1         | Point-to-point connection to another router | Neighbor router ID       |
| 2         | Connection to a transit network             | DR’s interface address   |
| 3         | Connection to a stub network                | IP network/subnet number |
| 4         | Virtual link                                | Neighbor router ID       |

#### Note

A stub network is a dead-end link that has only one router attached. A virtual link is a special case in OSPF (and is described later in this chapter in the "Configuring OSPF Special Area Types" section).

A *link data* field is also specified for each link, providing 32 bits of extra information. For most link types this is the IP interface address of the associated router interface. For links to stub networks, this field provides the stub network's subnet mask.

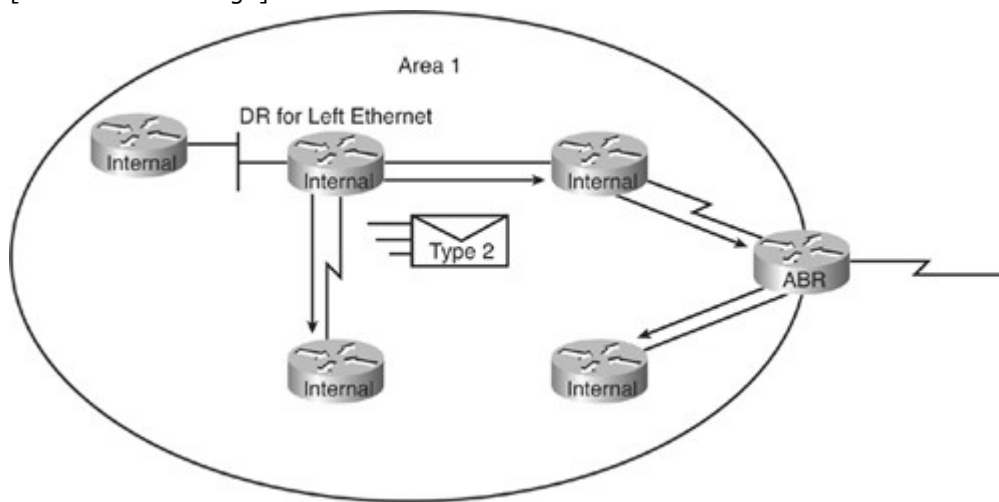
In addition, the type 1 LSA indicates the OSPF cost for each link, and whether the router is an ABR or ASBR.

#### LSA Type 2: Network LSA

A type 2 LSA is generated for every transit broadcast or NBMA network within an area. A transit network has at least two directly attached OSPF routers, as shown in Figure 3-26. A multiaccess network such as Ethernet is an example of a transit network. A type 2 network LSA lists each of the attached routers that make up the transit network, including the DR itself, and the subnet mask of the link.

Figure 3-26. LSA Type 2: Network LSA.

[View full size image]

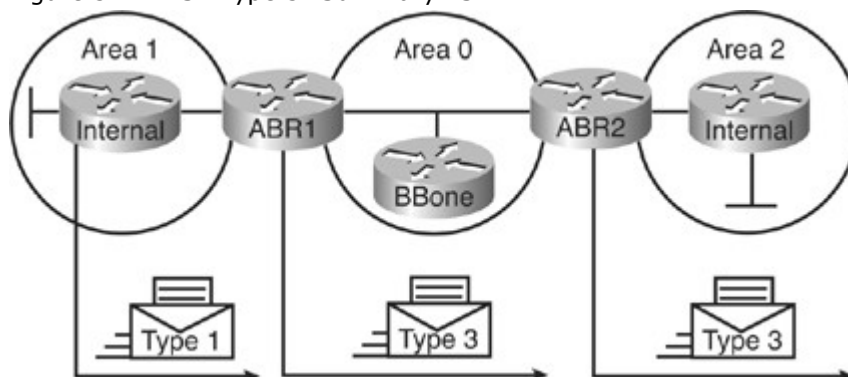


The transit link's DR is responsible for advertising the network LSA. The type 2 LSA then floods to all routers within the transit network area. Type 2 LSAs never cross an area boundary. The link-state ID for a network LSA is the IP interface address of the DR that advertises it.

#### LSA Type 3: Summary LSA

An ABR sends type 3 summary LSAs. A type 3 LSA advertises networks owned by an area to the rest of the areas in the OSPF autonomous system, as shown in Figure 3-27. As the figure illustrates, type 1 LSAs stay within an area. When an ABR receives type 1 LSAs from other routers within an area, it sends out type 3 summary LSAs to advertise the networks learned via these type 1 LSAs to other areas. The type 3 LSAs are flooded throughout a single area only but are regenerated by ABRs to flood into other areas.

Figure 3-27. LSA Type 3: Summary LSA.



By default, OSPF does not automatically summarize groups of contiguous subnets, or even summarize a network to its classful boundary. ABRs flood summary LSAs to other areas regardless of whether the routes listed in the LSAs are summarized. The network administrator, through configuration commands, must specify if and how the summarization will occur. By default, a type 3 summary LSA is advertised into the backbone area for every subnet defined in the originating area. Because Type 3 LSAs do not, by default, contain summarized routes, by default, all subnets in an area will be advertised. This can cause significant flooding problems. Consequently, manual route summarization (also called aggregation) at the ABR should always be considered. (OSPF route summarization is discussed later in this chapter, in the "Configuring OSPF Route Summarization" section.)

#### Note

Receiving a type 3 LSA that has been injected into its area does not cause a router to run the SPF algorithm. The routes being advertised in the type 3 LSAs are appropriately added to or deleted from the router's routing table, but a full SPF calculation is not necessary. (Debug output indicates that a "partial SPF" is necessary, but the counter for the number of times the SPF algorithm executes does not increment.)

Note that some Cisco documentation implies that the SPF algorithm is run when type 3 LSAs are injected into an area, but this is not the case.

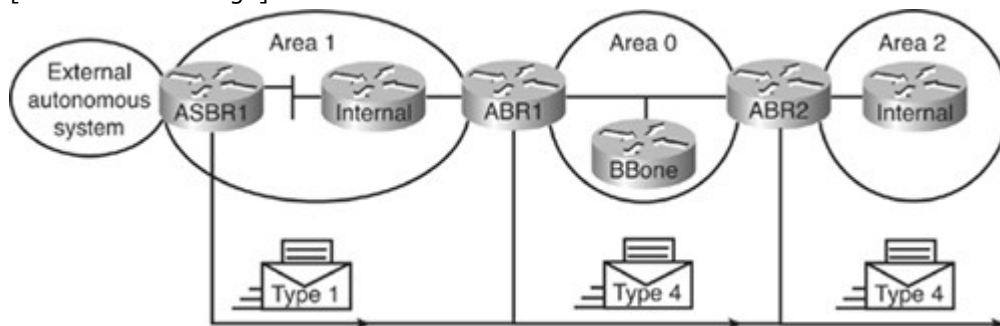
#### LSA Type 4: Summary LSA

A type 4 summary LSA is generated by an ABR only when an ASBR exists within an area. A type 4 LSA identifies the ASBR and provides a route to it; all traffic destined for an external autonomous system requires routing table knowledge of the ASBR that originated the external routes. The link-state ID is set to the ASBR's router ID.

In Figure 3-28, the ASBR sends a type 1 router LSA with a bit (known as the external bit [e bit]) that is set to identify itself as an ASBR. Type 1 LSAs stay within an area. However, when the ABR (identified with the border bit [b bit] in the router LSA) receives this type 1 LSA, it builds a type 4 LSA and floods it to the backbone, area 0. Subsequent ABRs regenerate a type 4 LSA to flood into their areas.

Figure 3-28. LSA Type 4: Summary LSA.

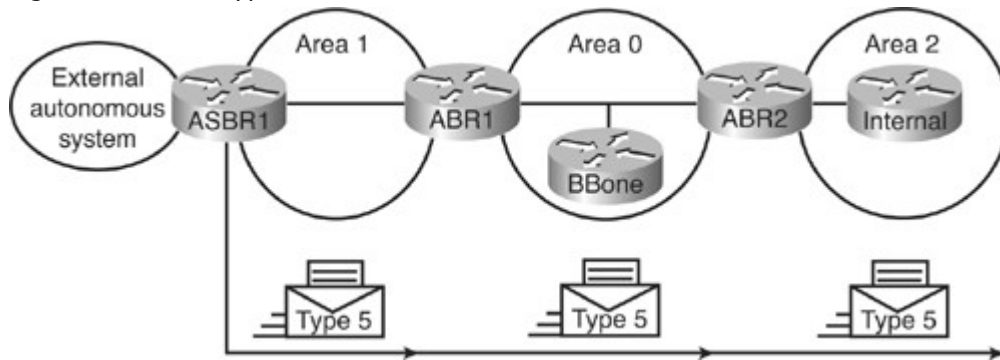
[View full size image]



#### LSA Type 5: External LSA

Type 5 external LSAs describe routes to networks outside the OSPF autonomous system. Type 5 LSAs are originated by the ASBR and are flooded to the entire autonomous system, as shown in Figure 3-29. The link-state ID is the external network number. Because of the flooding scope and depending on the number of external networks, the default lack of route summarization can also be a major issue with external LSAs. The network administrator should always attempt to summarize blocks of external network numbers at the ASBR to reduce flooding problems.

Figure 3-29. LSA Type 5: External LSA.

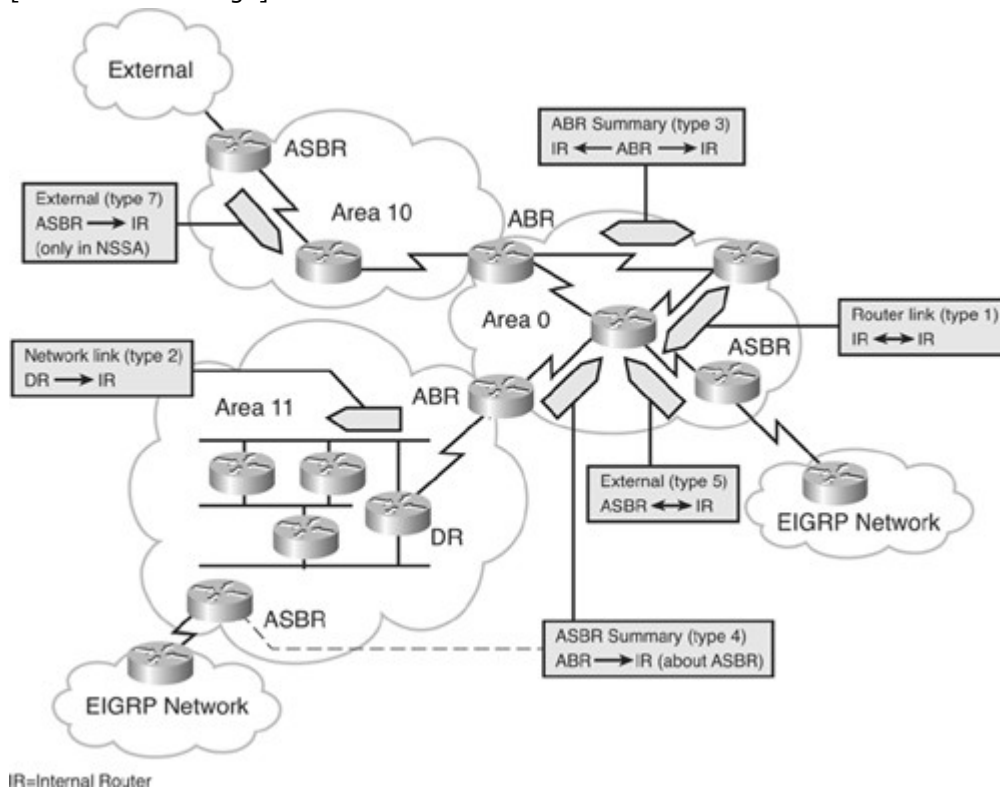


#### Example OSPF LSAs in a Network

Figure 3-30 illustrates the different types of LSAs found in an OSPF network. Note that only one example of each LSA type exchange is shown in this figure.

Figure 3-30. Example of a Variety of LSAs Within a Network.

[View full size image]



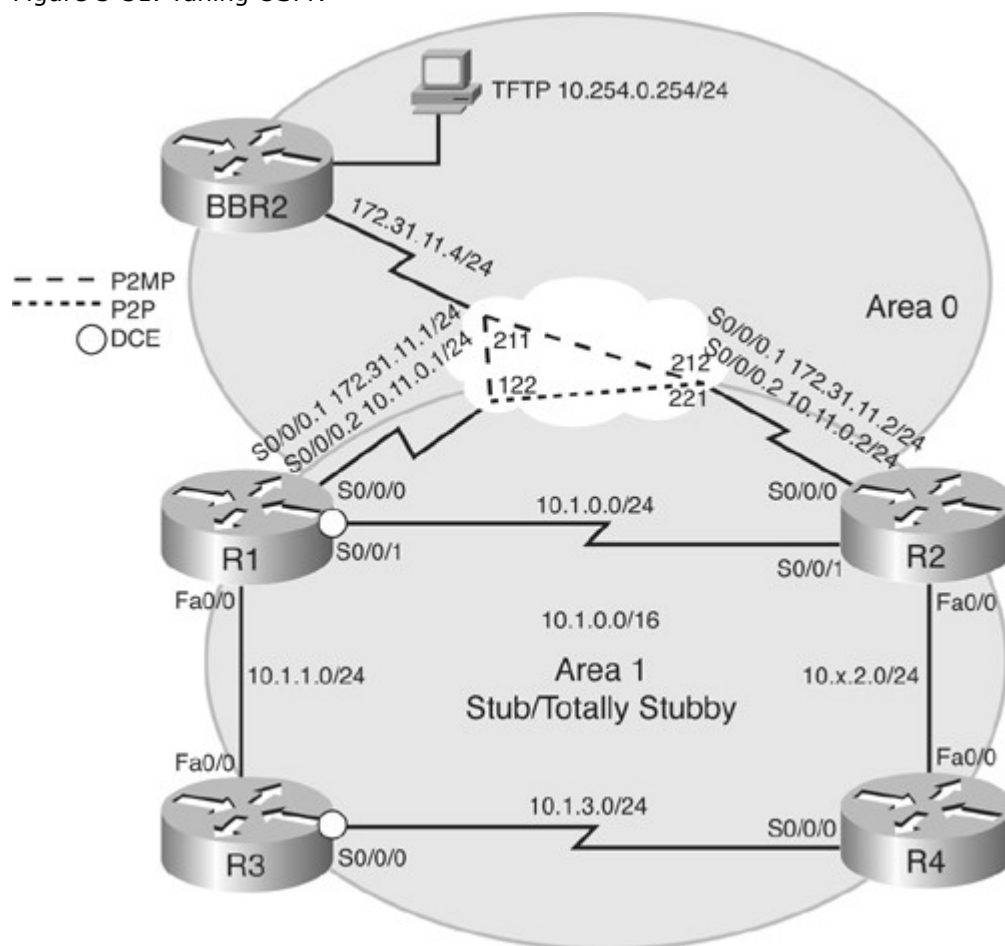
#### Interpreting the OSPF LSDB and Routing Table

This section explains the relationship between and how to interpret the OSPF LSDB and routing table. The two types of OSPF external costs are described, as is the OSPF LSDB overload protection feature.

##### OSPF LSDB

Example 3-21 illustrates output from the show ip ospf database command, used to get information about an OSPF LSDB, on an ABR. In this output, the router link states are type 1 LSAs, the net link states are type 2 LSAs, the summary net link states are type 3 LSAs, the summary ASBR link states are type 4 LSAs, and the external link states are type 5 LSAs. This output is from the R1 router in Figure 3-31.

Figure 3-31. Tuning OSPF.



#### Note

In this network, the BBR2 router is also connected to another set of routers that use subnets of 10.2.0.0/16 and 172.31.22.0/24.

The router ID of BBR2 is 200.200.200.200. The router ID of R1 is 10.0.0.11. The router ID of R2 is 10.0.0.12. The router ID of R3 is 10.200.200.13. The router ID of R4 is 10.200.200.14.

#### Example 3-21. *show ip ospf database* Command

Code View: Scroll / Show All

R1#**show ip ospf database**

OSPF Router with ID (10.0.0.11) (Process ID 1)

Router Link States (Area 0)

| Link ID         | ADV Router      | Age | Seq#       | Checksum   | Link count |
|-----------------|-----------------|-----|------------|------------|------------|
| 10.0.0.11       | 10.0.0.11       | 485 | 0x80000004 | 0x002EE5   | 2          |
| 10.0.0.12       | 10.0.0.12       | 540 | 0x80000002 | 0x0046CB   | 2          |
| 10.0.0.21       | 10.0.0.21       | 494 | 0x80000042 | 0x00F8E1   | 1          |
| 10.0.0.22       | 10.0.0.22       | 246 | 0x80000042 | 0x00F6E0   | 1          |
| 200.200.200.200 | 200.200.200.200 | 485 |            | 0x800001CB | 0x00E504 6 |

Summary Net Link States (Area 0)

| Link ID   | ADV Router | Age  | Seq#       | Checksum |
|-----------|------------|------|------------|----------|
| 10.1.0.0  | 10.0.0.11  | 486  | 0x8000001A | 0x00C92A |
| 10.1.0.0  | 10.0.0.12  | 541  | 0x8000001A | 0x00C32F |
| 10.1.1.0  | 10.0.0.11  | 486  | 0x8000001A | 0x002BD6 |
| 10.1.1.0  | 10.0.0.12  | 521  | 0x8000001C | 0x00BE30 |
| 10.1.2.0  | 10.0.0.11  | 486  | 0x8000001A | 0x00BD33 |
| 10.1.2.0  | 10.0.0.12  | 521  | 0x8000001C | 0x0016E7 |
| 10.1.3.0  | 10.0.0.11  | 487  | 0x8000001A | 0x00B23D |
| 10.1.3.0  | 10.0.0.12  | 527  | 0x80000001 | 0x00DE29 |
| 10.2.0.0  | 10.0.0.21  | 1759 | 0x8000003F | 0x00378C |
| 10.2.0.0  | 10.0.0.22  | 856  | 0x8000003F | 0x003191 |
| 10.2.1.0  | 10.0.0.21  | 1861 | 0x80000041 | 0x00943B |
| 10.2.1.0  | 10.0.0.22  | 856  | 0x8000003F | 0x003090 |
| 10.2.2.0  | 10.0.0.21  | 1861 | 0x80000049 | 0x00179F |
| 10.2.2.0  | 10.0.0.22  | 1359 | 0x80000044 | 0x007D4D |
| 10.2.3.0  | 10.0.0.21  | 1861 | 0x8000003F | 0x00209F |
| 10.2.3.0  | 10.0.0.22  | 1359 | 0x80000041 | 0x0016A6 |
| 10.11.0.0 | 10.0.0.11  | 589  | 0x80000018 | 0x005596 |
| 10.11.0.0 | 10.0.0.12  | 619  | 0x80000001 | 0x007D84 |

#### Router Link States (Area 1)

| Link ID       | ADV Router    | Age | Seq#       | Checksum | Link count |
|---------------|---------------|-----|------------|----------|------------|
| 10.0.0.11     | 10.0.0.11     | 613 | 0x80000006 | 0x000CF1 | 5          |
| 10.0.0.12     | 10.0.0.12     | 614 | 0x80000006 | 0x00F205 | 5          |
| 10.200.200.13 | 10.200.200.13 | 639 | 0x80000005 | 0x0006B4 | 3          |
| 10.200.200.14 | 10.200.200.14 | 635 | 0x80000005 | 0x00882C | 3          |

#### Net Link States (Area 1)

| Link ID  | ADV Router | Age | Seq#       | Checksum |
|----------|------------|-----|------------|----------|
| 10.1.1.1 | 10.0.0.11  | 640 | 0x80000001 | 0x00D485 |
| 10.1.2.2 | 10.0.0.12  | 635 | 0x80000001 | 0x00D183 |

#### Summary Net Link States (Area 1)

| Link ID     | ADV Router | Age | Seq#       | Checksum |
|-------------|------------|-----|------------|----------|
| 172.31.11.1 | 10.0.0.11  | 616 | 0x80000001 | 0x002F21 |
| 172.31.11.1 | 10.0.0.12  | 576 | 0x80000001 | 0x0064CA |
| 172.31.11.2 | 10.0.0.11  | 576 | 0x80000001 | 0x0060CE |
| 172.31.11.2 | 10.0.0.12  | 670 | 0x80000001 | 0x001F2F |
| 172.31.11.4 | 10.0.0.11  | 576 | 0x80000001 | 0x00AE8E |
| 172.31.11.4 | 10.0.0.12  | 630 | 0x80000001 | 0x00A893 |
| 172.31.22.4 | 10.0.0.11  | 576 | 0x80000001 | 0x0035FC |
| 172.31.22.4 | 10.0.0.12  | 630 | 0x80000001 | 0x002F02 |

#### Summary ASB Link States (Area 1)

| Link ID | ADV Router | Age | Seq# | Checksum |
|---------|------------|-----|------|----------|
|---------|------------|-----|------|----------|



|                 |           |     |            |          |
|-----------------|-----------|-----|------------|----------|
| 200.200.200.200 | 10.0.0.11 | 576 | 0x80000001 | 0x00688B |
| 200.200.200.200 | 10.0.0.12 | 631 | 0x80000001 | 0x006290 |

#### Type-5 AS External Link States

| Link ID    | ADV Router      | Age | Seq#       | Checksum | Tag |
|------------|-----------------|-----|------------|----------|-----|
| 10.254.0.0 | 200.200.200.200 | 451 | 0x8000019D | 0x00DADD | 0   |

R1#

The columns displayed in the OSPF database in Example 3-21 are described as follows:

- **Link ID**— Identifies each LSA.
- **ADV Router**— Advertising router—the LSA's source router.
- **Age**— The maximum age counter in seconds. The maximum age is 1 hour, or 3600 seconds.
- **Seq#**— The LSA's sequence number. It begins at 0x80000001 and increases with each update of the LSA.
- **Checksum**— Checksum of the individual LSA to ensure reliable receipt of that LSA.
- **Link count**— Used only on router LSAs, this is the total number of directly attached links. The link count includes all point-to-point, transit, and stub links. Point-to-point serial links count as two. All other links, including Ethernet links, count as one.

Because Router R1 in Figure 3-31 is an ABR, it includes an LSDB for both of the areas it is attached to, area 0 and area 1. In area 0, only type 1 and type 3 LSAs exist. Area 1 has 5 LSA types. Router BBR2 is an ASBR, which creates a type 5 LSA to advertise the route to 10.254.0.0 into OSPF. The type 5 LSA is flooded into all areas by default. The ABR for area 1, Router R1, creates a type 4 LSA describing how to reach the ASBR, Router BBR2.

Notice that R1 is advertising all the area 1 subnets as Type 3 LSAs (Summary net link states) in area 0, and R1 is advertising all the area 0 subnets as Type 3 LSAs in area 1.

#### OSPF Routing Table and Types of Routes

The show ip route command output shown in Example 3-22 displays the IP routing table on a router.

Example 3-22. *show ip route* Command Output with Internal and External OSPF Routes

```
RouterB>show ip route
<output omitted>
Gateway of last resort is not set
 172.31.0.0/24 is subnetted, 2 subnets
O IA 172.31.2.0 [110/1563] via 10.1.1.1, 00:12:35, FastEthernet0/0
O IA 172.31.1.0 [110/782] via 10.1.1.1, 00:12:35, FastEthernet0/0
 10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C 10.200.200.13/32 is directly connected, Loopback0
C 10.1.3.0/24 is directly connected, Serial0/0/0
O 10.1.2.0/24 [110/782] via 10.1.3.4, 00:12:35, Serial0/0/0
C 10.1.1.0/24 is directly connected, FastEthernet0/0
O 10.1.0.0/24 [110/782] via 10.1.1.1, 00:12:37, FastEthernet0/0
O E2 10.254.0.0/24 [110/50] via 10.1.1.1, 00:12:37, FastEthernet0/0
```

Table 3-14 describes each of the routing table designators for OSPF.

Table 3-14. Types of OSPF Routes

| Route Designator | Description                                  |                                                                                                                                                                                                                              |
|------------------|----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| O                | OSPF intra-area (router LSA) and network LSA | Networks from within the router's area, advertised by way of router LSAs and network LSAs.                                                                                                                                   |
| O IA             | OSPF interarea (summary LSA)                 | Networks from outside the router's area but within the OSPF autonomous system, advertised by way of summary LSAs.                                                                                                            |
| O E1             | Type 1 external routes                       | Networks from outside the router's autonomous system, advertised by way of external LSAs.                                                                                                                                    |
| O E2             | Type 2 external routes                       | Networks from outside the router's autonomous system, advertised by way of external LSAs. The difference between E1 and E2 external routes is described in the upcoming "Calculating the Costs of E1 and E2 Routes" section. |

Router and network LSAs describe the details within an area.

When an ABR receives summary or external LSAs, it adds them to its LSDB and regenerates and floods them into the local area. The internal routers then assimilate the information into their databases.

The SPF algorithm is then run against the LSDB to build the SPF tree, which is used to determine the best paths. The following is the order in which the best paths are calculated:

1. All routers calculate the best paths to destinations *within* their area (intra-area) and add these entries to the routing table. These are the type 1 and type 2 LSAs, which are noted in the routing table with a routing designator of O (OSPF).
2. All routers calculate the best paths to the *other* areas in the internetwork. These best paths are the interarea route entries, the type 3 and type 4 LSAs. They are noted with a routing designator of O IA (interarea).
3. All routers (except those that are in the form of a stub area) calculate the best paths to the external autonomous system (type 5) destinations. These routes are either external type 1 (E1), indicated with an O E1 in the routing table, or external type 2 (E2), indicated with an O E2 in the routing table, depending on the configuration.

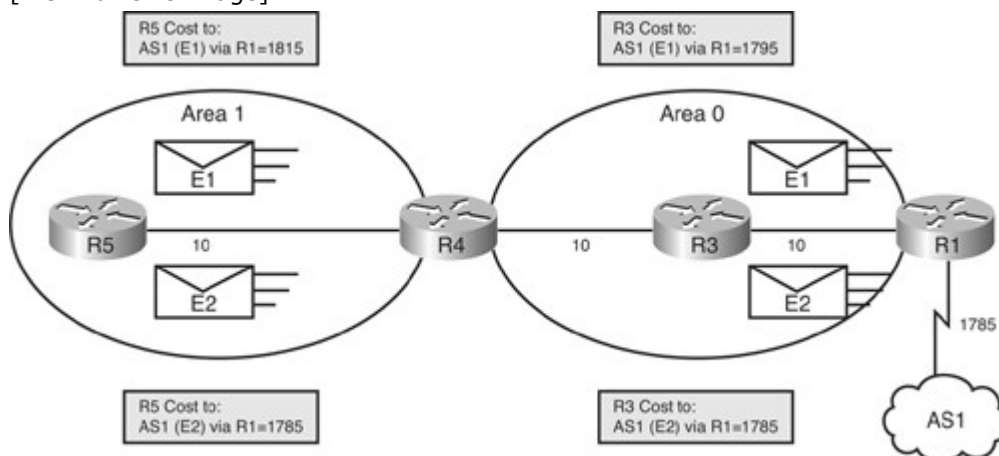
At this point, a router can communicate with any network within or outside the OSPF autonomous system.

#### Calculating the Costs of E1 and E2 Routes

The cost of an external route varies, depending on the external type configured on the ASBR, as shown in Figure 3-32.

Figure 3-32. Calculating the Costs of E1 and E2 Routes.

[View full size image]



The following external cost types can be configured:

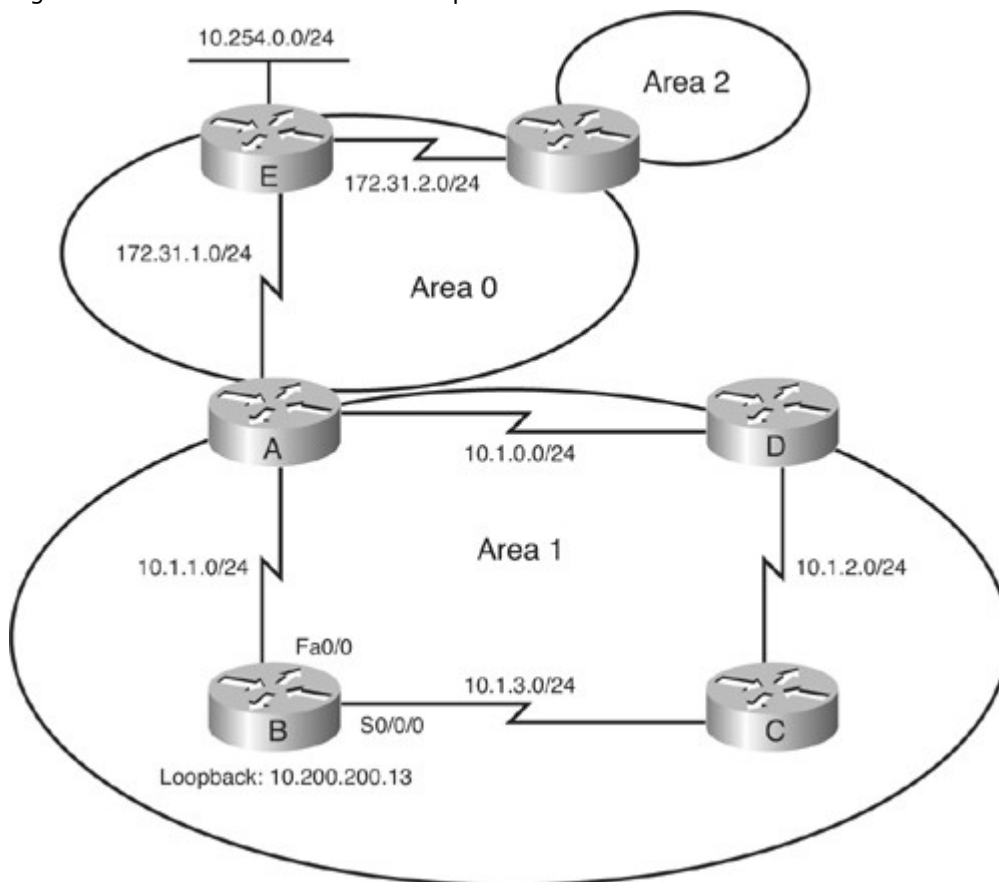
- **E1**— Type O E1 external routes calculate the cost by adding the external cost to the internal cost of each link the packet crosses. Use this type when multiple ASBRs are advertising an external route to the same autonomous system, to avoid suboptimal routing.
- **E2 (default)**— The external cost of O E2 packet routes is always the external cost only. Use this type if only one ASBR is advertising an external route to the autonomous system.

In the lower part of Figure 3-32, the E2 cost to the route in the external AS1 is always 1785. For example, if there were multiple paths to the external route, and E2 costs were used, there would be no distinction between the paths.

In the upper part of part of Figure 3-32, E1 costs are used, so the cost increases at each router as the internal cost is added to the external cost. This allows the optimal path to be selected if multiple paths are available.

The output in the earlier Example 3-22 is from Router B in Figure 3-33. The last entry (O E2) in Example 3-22 is an external route from the ASBR (Router E), via the ABR (Router A). The two numbers in brackets [110/50] are the administrative distance and the total cost of the route to the specific destination network, respectively. In this case, the administrative distance is the default for all OSPF routes of 110, and the E2 cost of the route has been calculated as 50.

Figure 3-33. Network Used for Example 3-22.



#### Configuring OSPF LSDB Overload Protection

If other routers are misconfigured, causing, for example, a redistribution of a large number of prefixes, large numbers of LSAs can be generated. These excessive LSAs can drain local CPU and memory resources. OSPF LSDB overload protection can be configured to protect against this issue with Cisco IOS Software Release 12.3(7)T and later (and some specific earlier releases) by using the **max-lsa** *maximum-number* [*threshold-percentage*] [**warning-only**] [**ignore-time** *minutes*] [**ignore-count** *count-number*] [**reset-time** *minutes*] router configuration command.

Table 3-15 lists the parameters of the max-lsa command.

Table 3-15. *max-lsa* Command Parameters

| Parameter                               | Description                                                                                                                                                               |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| maximum-number                          | Maximum number of non-self-generated LSAs that the OSPF process can keep in the OSPF LSDB.                                                                                |
| threshold-percentage                    | (Optional) The percentage of the maximum LSA number, as specified by the <i>maximum-number</i> argument, at which a warning message is logged. The default is 75 percent. |
| warning-only                            | (Optional) Specifies that just a warning message is sent when the maximum limit for LSAs is exceeded. The OSPF process never enters ignore state. Disabled by default.    |
| <b>ignore-time</b> <i>minutes</i>       | (Optional) Specifies the time, in minutes, to ignore all neighbors after the maximum limit of LSAs has been exceeded. The default is 5 minutes.                           |
| <b>ignore-count</b> <i>count-number</i> | (Optional) Specifies the number of times that the OSPF process can consecutively be placed into the ignore state. The default is five times.                              |
| <b>reset-time</b> <i>minutes</i>        | (Optional) Specifies the time, in minutes, after which the ignore count is reset to 0. The default is 10 minutes.                                                         |

When this feature is enabled, the router keeps count of the number of received (non-self-generated) LSAs that it keeps in its LSDB. An error message is logged when this number reaches a configured threshold number, and a notification is sent when it exceeds the threshold number.

If the LSA count still exceeds the threshold after 1 minute, the OSPF process takes down all adjacencies and clears the OSPF database. This is called the *ignore* state. In this ignore state, no OSPF packets are sent or received by interfaces that belong to that OSPF process.

The OSPF process remains in the ignore state for the time that is defined by the **ignore-time** parameter. The **ignore-count** parameter defines the maximum number of times that the OSPF process can consecutively enter the ignore state before remaining permanently down and requiring manual intervention. If the OSPF process remains normal for the time that is defined by the **reset-time** parameter, the ignore state counter is reset to 0.

## Configuring and Verifying Advanced OSPF Features

This section explores some OSPF advanced features, including the following:

- Using the passive-interface command with OSPF
- Propagating an OSPF default route
- Configuring OSPF route summarization
- OSPF virtual links
- Configuring OSPF special area types

### Using the passive-interface Command with OSPF

As described in Chapter 2, the passive-interface type number [default] router configuration command prevents routing updates from being sent through the specified router interface. This command can be used with all IP-based routing protocols except BGP. Table 3-16 describes the parameters in this command.

Table 3-16. *passive-interface* Command

| Parameter   | Description                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| type number | Specifies the type of interface and interface number that will not send routing updates (or establish neighbor relationships for link-state routing protocols and EIGRP) |
| default     | (Optional) A parameter that sets all interfaces on the router as passive by default                                                                                      |

OSPF's behavior with the **passive-interface** command is somewhat different than other protocols. With OSPF, the specified interface appears as a stub network in the OSPF domain, and OSPF routing information is neither sent nor received through the specified router interface.

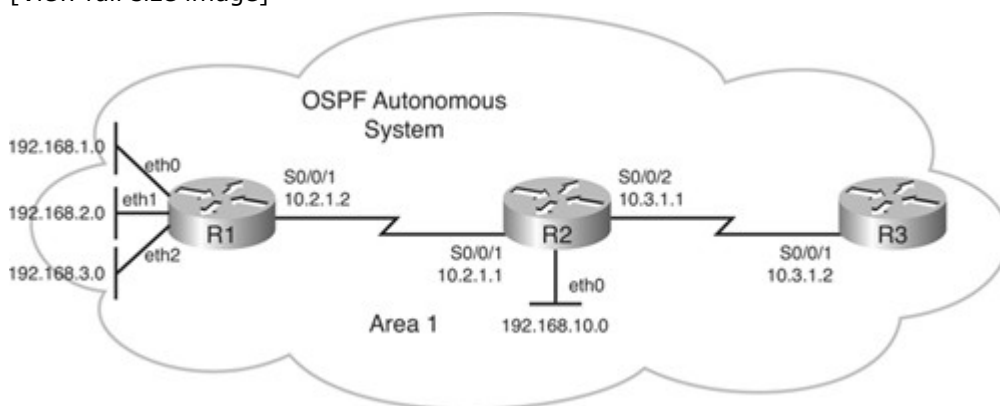
Recall that the **network** command defines the interfaces over which OSPF will attempt to establish neighbor relationships and the networks that will be advertised to OSPF neighbors. Configuring an interface as passive disables only the neighbor relationship establishment. The router will still advertise the network to its OSPF neighbors.

In Internet service providers (ISPs) and large enterprise networks, many distribution routers have more than 100 interfaces. Before the **passive-interface default** command was introduced in Cisco IOS Software Release 12.0, network administrators would configure the routing protocol on all interfaces and then manually set the **passive-interface** command on the interfaces on which they did not require adjacencies to be established. However, this solution meant entering many **passive-interface** commands. A single **passive-interface default** command can now be used to set all interfaces to passive by default. To enable routing on individual interfaces where you require adjacencies to be established, use the **no passive-interface** command.

In the example shown in Figure 3-34, routers are configured for OSPF. Example 3-23 provides the configuration of Routers R1 and R2.

Figure 3-34. The *passive-interface* Command Causes OSPF to Neither Send nor Receive.

[View full size image]



Example 3-23. Configurations of Router R1 and R2 in Figure 3-34

```
R1#
router ospf 100
network 192.168.0.0 0.0.255.255 area 1
network 10.2.0.0 0.0.255.255 area 1
passive-interface default
no passive-interface Serial0/0/1
```

```
R2#
router ospf 100
```

```

network 192.168.0.0 0.0.255.255 area 1
network 10.2.0.0 0.0.255.255 area 1
network 10.3.0.0 0.0.255.255 area 1
passive-interface Ethernet0

```

Router R1 has a set of interfaces that act as stub networks. LSAs are not received through these interfaces anyway, and there is no need for LSAs to be sent out of these interfaces. The only interface that should participate in the OSPF process is interface Serial0/0/1. The **passive-interface default** command is used on router R1, because it is easier to make all the interfaces passive and then enable the Serial 0/0/1 interface with the **no passive-interface Serial0/0/1** command.

For Router R2, only one interface is a stub interface, where the propagation of LSAs should be stopped. The **passive-interface Ethernet0** command is used to stop the propagation of LSAs through only interface Ethernet0.

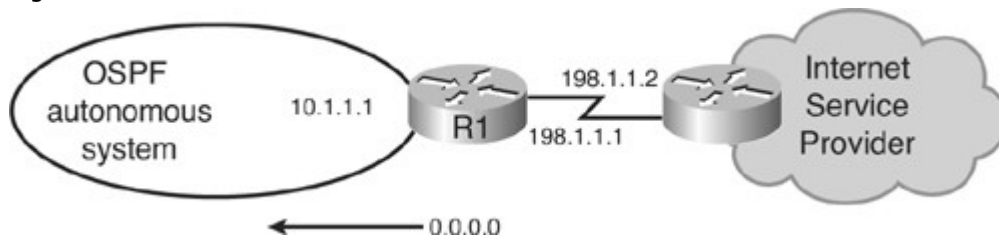
#### Propagating an OSPF Default Route

You may want to configure OSPF to advertise a default route into its autonomous system, as described in this section.

To be able to route from an OSPF autonomous system toward external networks or the Internet, static routes to all the external networks must be created, or the routes must be redistributed into OSPF, or a default route must be created. Using a default route is the most scalable solution and results in smaller routing tables and fewer resource and CPU load used. For example, there is no need to recalculate the SPF algorithm when external networks fail. If default routes are used, no recalculation will occur.

Figure 3-35 shows how OSPF injects a default route into a standard area (the different types of areas are covered in the “Configuring OSPF Special Area Types” section, later in this chapter). Any OSPF router can originate default routes injected into a standard area. However, OSPF routers do not, by default, generate a default route into the OSPF domain. For OSPF to generate a default route, you must use the **default-information originate** command. Whenever the **default-information originate** command (or redistribution into OSPF) is configured on an OSPF router, the router becomes an ASBR.

Figure 3-35. Default Routes in OSPF.



There are two ways to advertise a default route into a standard area. The first is to advertise 0.0.0.0 into the OSPF domain, provided that the advertising router already has a default route. This is accomplished with the **default-information originate** command. The second is to advertise 0.0.0.0 regardless of whether the advertising router already has a default route. This is accomplished by adding the keyword **always** to the **default-information originate** command.

To generate a default external route into an OSPF routing domain, use the **default-information originate [always] [metric metric-value] [metric-type type-value] [route-map map-name]** router configuration command. Table 3-17 explains the options of the **default-information originate** command.

Table 3-17. *default-information originate* Command Parameters

| Parameter                  | Description                                                                                                                                                                                                                                                        |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>always</b>              | (Optional) Specifies that OSPF always advertises the default route regardless of whether the router has a default route in the routing table.                                                                                                                      |
| <b>metric metric-value</b> | (Optional) A metric used for generating the default route. If you omit a value and do not specify a value using the <b>default-metric</b> router configuration command, the default metric value is 1. Cisco IOS Software documentation indicates that the default |

| Parameter                            | Description                                                                                                                                                                                                                                                                                          |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                      | metric value is 10, but testing shows that it is actually 1. See the "default-information originate Command Actual Behavior" sidebar for more details.                                                                                                                                               |
| <b>metric-type</b> <i>type-value</i> | (Optional) The external link type that is associated with the default route that is advertised into the OSPF routing domain. It can be one of the following values: 1—Type 1 external route, 2—Type 2 external route. The default is type 2 external route (indicated by O*E2 in the routing table). |
| <b>route-map</b> <i>map-name</i>     | (Optional) Specifies that the routing process generates the default route if the route map is satisfied.                                                                                                                                                                                             |

A default route shows up in the OSPF database as an external LSA type 5, as shown in Example 3-24.  
**Example 3-24. Default Route in the OSPF Database**

| Type-5 AS External Link States |            |     |            |          |     |
|--------------------------------|------------|-----|------------|----------|-----|
| Link ID                        | ADV Router | Age | Seq#       | Checksum | Tag |
| 0.0.0.0                        | 198.1.1.1  | 601 | 0x80000001 | 0xD0D8   | 0   |

#### default-information originate Command Actual Behavior

The Cisco IOS Software documentation for the **default-information originate** command mentions that if you omit a value for the *metric* variable and do not specify a value using the **default-metric** router configuration command, the default metric value for the **default-information originate** command is 10. Testing shows that it is actually 1.

To verify the default value of the *metric* when using the **default-information originate** command, tests were performed on Ethernet, serial HDLC, and serial Frame Relay links. In all cases, the metric was 1. The following is the configuration used on two routers in one of the tests:

Code View: Scroll / Show All

Router R1:

```
interface Serial1
no ip address
encapsulation frame-relay
!
interface Serial1.1 multipoint
ip address 10.0.0.1 255.0.0.0
ip ospf network broadcast
frame-relay interface-dlci 120
!
router ospf 10
log-adjacency-changes
network 10.0.0.0 0.0.0.255 area 0
default-information originate always
```

Router R2:

```
interface Serial0
ip address 10.0.0.2 255.0.0.0
encapsulation frame-relay
ip ospf network broadcast
no fair-queue
clockrate 2000000
```



```

frame-relay interface-dlci 120
frame-relay intf-type dce
!
router ospf 10
network 10.0.0.0 0.0.0.255 area 0

```

The following is the routing table on Router R2, when the command **default-information originate** was issued on R1 without specifying a value for the *metric* variable. The metric appearing in the routing table for the default route is 1, not 10 as mentioned in the documentation:

R2>**show ip route**

<output omitted>

Gateway of last resort is 10.0.0.1 to network 0.0.0.0

```

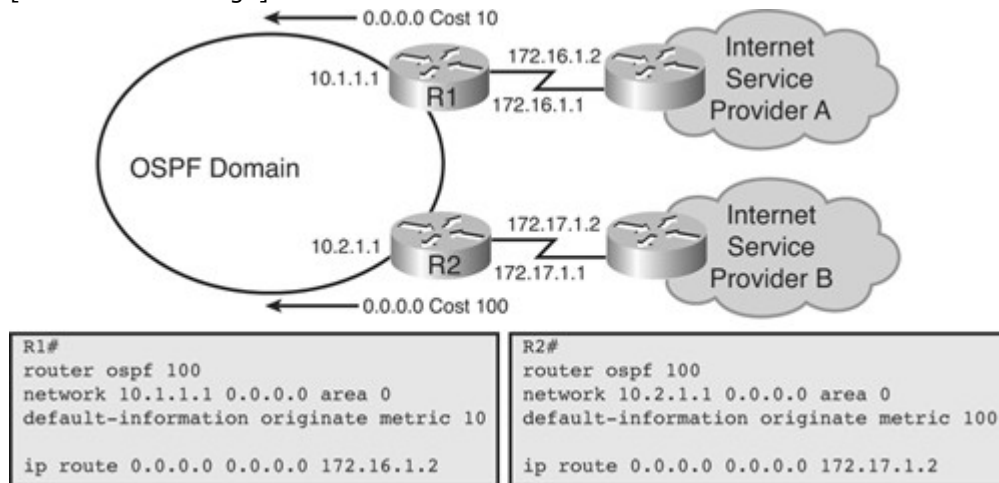
C 10.0.0.0/8 is directly connected, Serial0
O*E2 0.0.0.0/0 [110/1] via 10.0.0.1, 00:18:49, Serial0

```

Figure 3-36 shows an OSPF network multihomed to dual ISPs. It makes no sense to redistribute all the ISP's routes into OSPF. In this case, default routes are sent into OSPF. The optional metric parameter has been used to prefer the default route to ISP A with its metric of 10, over the backup connection to ISP B with its metric of 100. The default route being generated has a metric-type of E2 by default, so the metric does not increase as it goes through the area. As a result, all routers, regardless of their proximity to the border router, prefer ISP A over ISP B. Because the `always` parameter is not used on these routers, a default route must exist in the IP routing table for the router to advertise a default route into OSPF. Both routers have a default static route configured.

Figure 3-36. Default Route Example.

[View full size image]



#### Note

The `default-information originate` command causes the router to send a default route to all its OSPF neighbors. In Figure 3-36, notice that the R1 and R2 routers are not running OSPF on their connections to the ISP routers, and are therefore not passing a default route to the ISP routers.

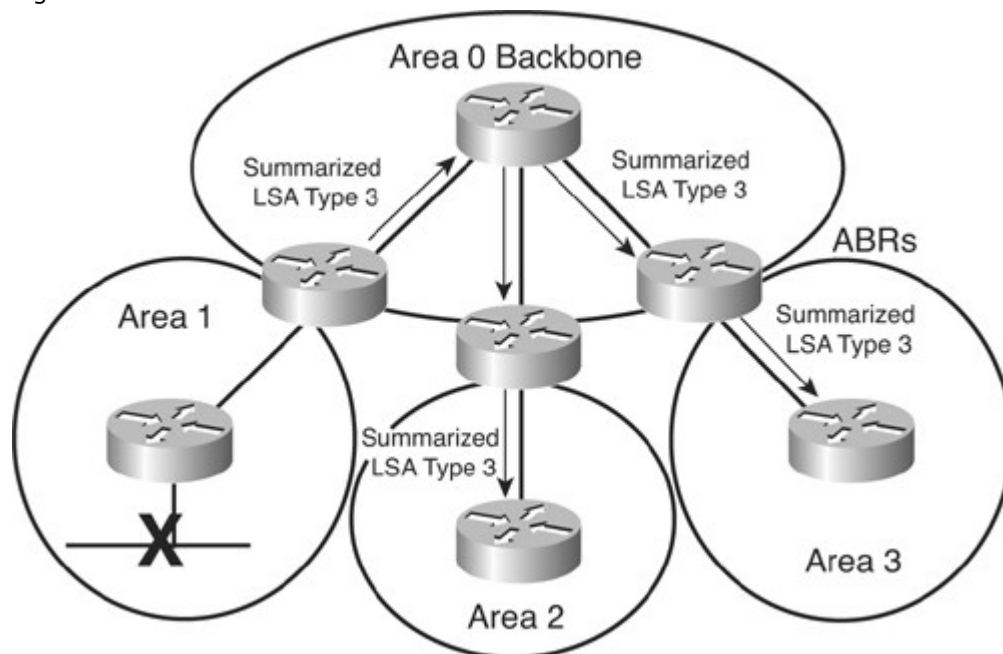
## Configuring OSPF Route Summarization

Route summarization involves consolidating multiple routes into a single advertisement. Proper route summarization directly affects the amount of bandwidth, CPU, and memory resources consumed by the OSPF routing process.

Without route summarization, every specific-link LSA is propagated into the OSPF backbone and beyond, causing unnecessary network traffic and router overhead.

Route summarization can be configured so that only summarized routes propagate into the backbone (area 0), as shown in Figure 3-37. Route summarization helps to solve two OSPF issues: large routing tables and frequent LSA flooding throughout the autonomous system. This summarization is important because it prevents every router from having to update its routing table, increases the network's stability, and reduces unnecessary LSA flooding. Also, if a network link fails, the topology change is not propagated into the backbone (and other areas by way of the backbone). Specific-link LSA flooding outside the area does not occur.

Figure 3-37. Benefits of Route Summarization.



Normally, type 1 and type 2 LSAs are generated inside each area, translated into type 3 LSAs, and sent to other areas. With route summarization, the ABR or ASBR routers consolidate multiple routes into fewer advertisements. ABR routers summarize type 3 LSAs and ASBR routers summarize type 5 LSAs. Instead of advertising many specific prefixes, the ABR routers and ASBR routers advertise only one summary prefix.

It is important to remember that summary LSAs (the type 3 LSAs) and external LSAs (the type 5 LSAs) by default do *not* contain summarized (aggregated) routes. In other words, by default, summary LSAs are not summarized.

The two types of summarization are as follows:

- **Interarea route summarization**— Interarea route summarization occurs on ABRs and applies to routes from within each area. It does not apply to external routes injected into OSPF via redistribution. To perform effective interarea route summarization, network numbers within areas should be assigned contiguously so that these addresses can be summarized into a minimal number of summary addresses.
- **External route summarization**— External route summarization is specific to external routes that are injected into OSPF via route redistribution. Again, it is important to ensure the contiguity of the external address ranges that are being summarized. Generally, only ASBRs summarize external routes.

### Note

Recall from Chapter 2 that any EIGRP router can perform summarization. In OSPF, however, summarization can only be performed on ABRs (interarea summarization) and ASBRs (external route summarization).

If there are multiple ASBRs, or multiple ABRs in an area, suboptimal routing is possible if summarization is not configured correctly. Summarizing overlapping ranges from two different routers can cause packets to be sent to the wrong destination.

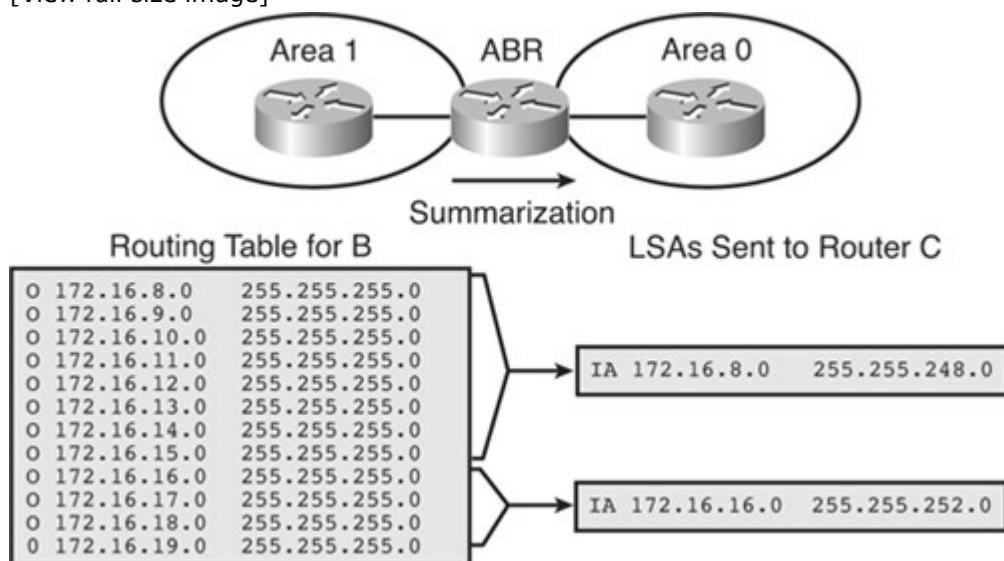
OSPF is a classless routing protocol, which means that it carries subnet mask information along with route information. Therefore, OSPF supports multiple subnet masks for the same major network, known as variable-length subnet masking (VLSM), and OSPF summary routes can have different subnet masks. OSPF also supports discontinuous subnets.

However, older protocols, such as Routing Information Protocol Version 1 (RIPv1), are classful and do not support VLSM or discontinuous subnets. Therefore, if the same major network crosses the boundaries of an OSPF and RIPv1 domain, VLSM information redistributed into RIPv1 would be lost, so static routes may have to be configured in the RIPv1 domain.

Network numbers in areas should be assigned contiguously to ensure that these addresses can be summarized into a minimal number of summary addresses. For example, in Figure 3-38, the list of 12 networks in Router B's routing table can be summarized into two summary address advertisements. The block of addresses from 172.16.8.0 to 172.16.15.0/24 can be summarized using 172.16.8.0/21, and the block from 172.16.16.0 to 172.16.19.0/24 can be summarized using 172.16.16.0/22.

Figure 3-38. Using Route Summarization.

[View full size image]



#### Configuring Inter-area OSPF Route Summarization on an ABR

OSPF does not perform autosummarization (on major network boundaries or elsewhere). To manually configure interarea route summarization on an ABR, use the following procedure:

Step 1. Configure OSPF.

Step 2. Use the `area area-id range address mask [advertise | not-advertise] [cost cost]` router configuration command, described in Table 3-18, to instruct the ABR to summarize routes for a specific area before injecting them into a different area via the backbone as type 3 summary LSAs.

Table 3-18. *area range* Command Parameters

| Parameter | Description                                                                                                |
|-----------|------------------------------------------------------------------------------------------------------------|
| area-id   | Identifies the area subject to route summarization (the area to which the routes being summarized belong). |
| address   | The summary address designated for a range of addresses.                                                   |
| mask      | The IP subnet mask used for the summary route.                                                             |
| advertise | (Optional) Sets the address range status to advertise and generates a type 3 summary LSA.                  |

| Parameter            | Description                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>not-advertise</b> | (Optional) Sets the address range status to DoNotAdvertise. The type 3 summary LSA is suppressed, and the component networks remain hidden from other networks.                     |
| <b>cost cost</b>     | (Optional) Metric or cost for this summary route, which is used during the OSPF SPF calculation to determine the shortest paths to the destination. The value can be 0 to 16777215. |

Summarization of internal routes can be done only by ABRs. When summarization is enabled on an ABR, it injects a single type-3 LSA describing the summary route into the backbone. Multiple routes inside the area are summarized by the one LSA.

A summary route will be generated if at least one subnet within the area falls in the summary address range. The summarized route metric will be equal to the lowest cost of all subnets within the summary address range.

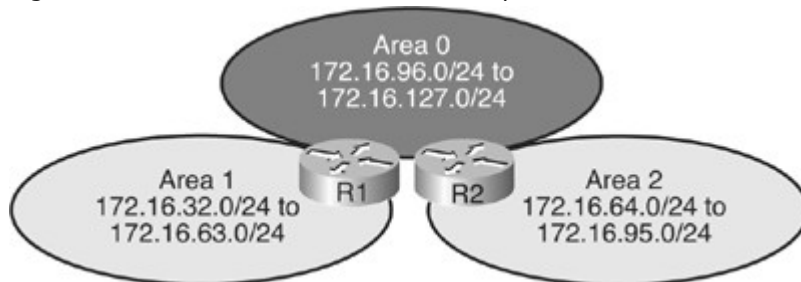
An ABR can only summarize routes that are within the areas connected to the ABR.

The Cisco IOS Software creates a summary route to interface null 0 when manual summarization is configured, to prevent routing loops. For example, if the summarizing router receives a packet to an unknown subnet that is part of the summarized range, the packet matches the summary route based on the longest match. The packet is forwarded to the null 0 interface (in other words, it is dropped), which prevents the router from forwarding the packet to a default route and possibly creating a routing loop.

Interarea Route Summarization Configuration Example on an ABR

Example 3-25 illustrates the configurations of the R1 and R2 routers in Figure 3-39, showing that route summarization can occur in both directions on an ABR—from a nonbackbone area to area 0 and from area 0 to a nonbackbone area. For example, the R1 configuration specifies the following summarization:

Figure 3-39. Route Summarization Example at the ABR.



- **Area 0 range 172.16.96.0 255.255.224.0**— Identifies area 0 as the area containing the range of networks to be summarized. The networks will be summarized into area 1. The ABR R1 summarizes the range of subnets from 172.16.96.0 to 172.16.127.0 into one range: 172.16.96.0 255.255.224.0.
- **Area 1 range 172.16.32.0 255.255.224.0**— Identifies area 1 as the area containing the range of networks to be summarized. The networks will be summarized into area 0. The ABR R1 summarizes the range of subnets from 172.16.32.0 to 172.16.63.0 into one range: 172.16.32.0 255.255.224.0.

Example 3-25. Enabling OSPF Routing on R1 and R2 in Figure 3-39

```
R1(config)#router ospf 100
R1(config-router)#network 172.16.32.1 0.0.0.0 area 1
R1(config-router)#network 172.16.96.1 0.0.0.0 area 0
R1(config-router)#area 0 range 172.16.96.0
255.255.224.0
R1(config-router)#area 1 range 172.16.32.0
255.255.224.0

R2(config)#router ospf 100
R2(config-router)#network 172.16.64.1 0.0.0.0 area 2
R2(config-router)#network 172.16.127.1 0.0.0.0 area 0
```

```
R2(config-router)#area 0 range 172.16.96.0
255.255.224.0
R2(config-router)#area 2 range 172.16.64.0
255.255.224.0
```

#### Note

Depending on your network topology, you may not want to summarize area 0 networks into other areas. For example, if you have more than one ABR between an area and the backbone area, sending a type 3 (summary) LSA with the explicit network information into an area ensures that the shortest path to destinations outside the area is selected. If you summarize the addresses, suboptimal path selection may occur.

#### Configuring External OSPF Route Summarization on an ASBR

By default, each external route, redistributed into OSPF from other protocols, is advertised individually with an external LSA. Summarization of external routes can be done on an ASBR for type 5 LSAs (redistributed routes) before injecting them into the OSPF domain. A summary route to null 0 is automatically created for each summary range.

To configure manual route summarization on an ASBR to summarize external routes, use the following procedure:

Step 1. Configure OSPF.

Step 2. Use the summary-address ip-address mask [not-advertise] [tag tag] router configuration command, described in Table 3-19, to instruct the ASBR to summarize external routes before injecting them into the OSPF domain as a type 5 external LSA.

Table 3-19. *summary-address* Command Parameters

| Parameter      | Description                                                                                         |
|----------------|-----------------------------------------------------------------------------------------------------|
| ip-address     | The summary address designated for a range of addresses                                             |
| mask           | The IP subnet mask used for the summary route                                                       |
| not-advertise  | (Optional) Used to suppress routes that match the address/mask pair                                 |
| <b>tag tag</b> | (Optional) A tag value that can be used as a "match" value to control redistribution via route maps |

#### Note

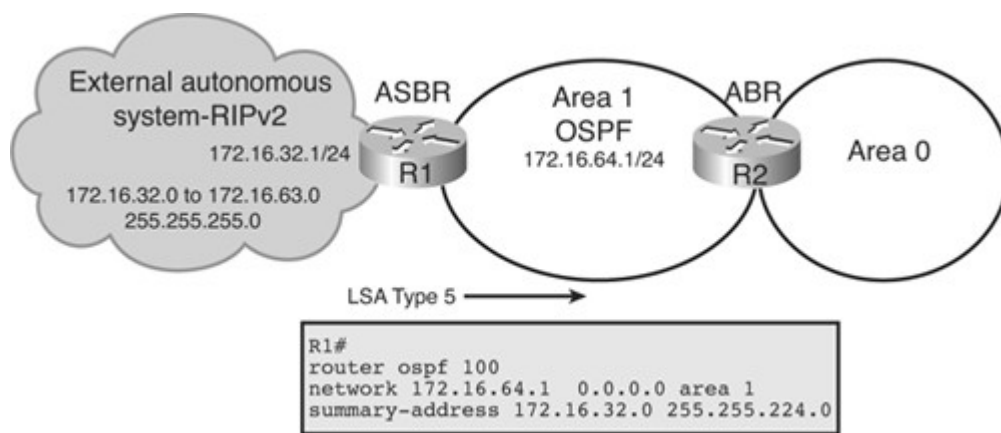
In an NSSA area, the ABR can summarize external routes when it creates a type 5 summary LSA from a type 7 LSA representing the external route. NSSA areas are described in the upcoming "Configuring NSSAs" section.

#### External Route Summarization Configuration Example on an ASBR

Figure 3-40 depicts route summarization on Router R1, an ASBR. The routes from the external autonomous system running RIPv2 are redistributed into OSPF on Router R1. Because of the contiguous subnet block in the RIP network, it is possible to summarize the 32 subnets into one summarized route, 172.16.32.0 255.255.224.0. Instead of 32 external type 5 LSAs flooding into the OSPF network, there is only one.

Figure 3-40. Route Summarization Example at the ASBR.

[View full size image]



#### Note

RIPv2 routes must also be redistributed into OSPF in this example. Redistribution is covered in Chapter 4.

#### OSPF Virtual Links

OSPF's two-tiered area hierarchy requires that if more than one area is configured, one of the areas must be area 0, the backbone area. All other areas must be directly connected to area 0, and area 0 must be contiguous. OSPF expects all nonbackbone areas to inject routes into the backbone, so that the routes can be distributed to other areas.

A virtual link is a link that allows discontinuous area 0s to be connected, or a disconnected area to be connected to area 0, via a transit area. The OSPF virtual link feature should be used only in very specific cases, for temporary connections or for backup after a failure. Virtual links should not be used as a primary backbone design feature.

Virtual links are part of the OSPF open standard and have been a part of Cisco IOS Software since software release 10.0.

The virtual link relies on the stability of the underlying intra-area routing. Virtual links cannot go through more than one area, nor through stub areas. Virtual links can only run through standard nonbackbone areas. If a virtual link needs to be attached to the backbone across two nonbackbone areas, then two virtual links are required, one per area.

In Figure 3-41, two companies running OSPF have merged and a direct link does not yet exist between their backbone areas. The resulting area 0 is discontinuous. A logical link (virtual link) is built between the two ABRs, routers A and B, across area 1, a nonbackbone area. The routers at each end of the virtual link become part of the backbone and act as ABRs. This virtual link is similar to a standard OSPF adjacency, except that in a virtual link, neighboring routers do not have to be directly attached.

Figure 3-41. Virtual Links Are Used to Connect a Discontinuous Area 0.

[View full size image]

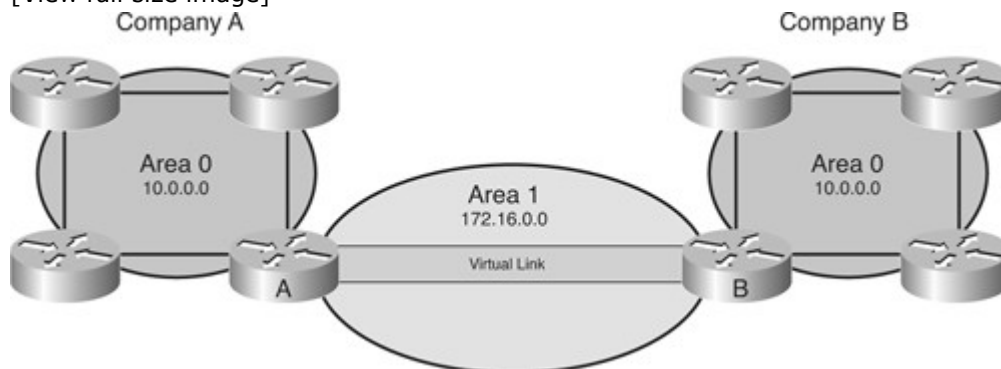
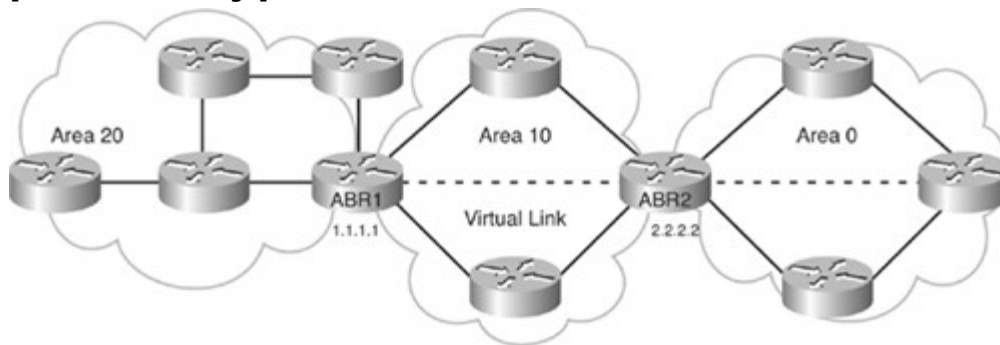


Figure 3-42 illustrates another example where a nonbackbone area is added to an OSPF network, and a direct physical connection to the existing OSPF area 0 does not yet exist. In this case, area 20 is added, and a virtual link across area 10 is created to provide a logical path between area 20 and the backbone area 0.



The OSPF database treats the virtual link between ABR1 and ABR2 as a direct link. For greater stability, loopback interfaces are used as router IDs, and virtual links are created using these loopback addresses. Figure 3-42. Virtual Links Are Used to Connect an Area to the Backbone Area. [View full size image]



The Hello protocol works over virtual links as it does over standard links, in 10-second intervals. However, LSA updates work differently on virtual links. An LSA usually refreshes every 30 minutes. However, LSAs learned through a virtual link have the DoNotAge (DNA) option set, so that the LSA does not age out. This DNA technique is required to prevent excessive flooding over the virtual link.

#### Configuring OSPF Virtual Links

Use the **area area-id virtual-link router-id [authentication [message-digest | null]] [hello-interval seconds] [retransmit-interval seconds] [transmit-delay seconds] [dead-interval seconds] [[authentication-key key] | [message-digest-key key-id md5 key]]** router configuration command to define an OSPF virtual link. To remove a virtual link, use the **no** form of this command.

Table 3-20 describes the options available with the **area area-id virtual-link** command. Make sure you understand the effect of these options before changing them. For instance, the smaller the hello interval, the faster the detection of topological changes, but the more routing traffic. You should be conservative with the setting of the retransmit interval, or the result is needless retransmissions. The value should be larger for serial lines and virtual links. The transmit delay value should take into account the interface's transmission and propagation delays.

Table 3-20. *area area-id virtual-link* Command Parameters

| Parameter                          | Description                                                                                                                                                                                                                                                                                                   |
|------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| area-id                            | Specifies the area ID of the transit area for the virtual link. This ID can be either a decimal value or in dotted-decimal format, like a valid IP address. There is no default. The transit area cannot be a stub area.                                                                                      |
| router-id                          | Specifies the router ID of the virtual link neighbor. The router ID appears in the <b>show ip ospf</b> display. This value is in an IP address format. There is no default.                                                                                                                                   |
| authentication                     | (Optional) Specifies an authentication type.                                                                                                                                                                                                                                                                  |
| message-digest                     | (Optional) Specifies the use of MD5 authentication.                                                                                                                                                                                                                                                           |
| null                               | (Optional) Overrides simple password or MD5 authentication if configured for the area. No authentication is used.                                                                                                                                                                                             |
| <b>hello-interval</b> seconds      | (Optional) Specifies the time (in seconds) between the hello packets that the Cisco IOS Software sends on an interface. The unsigned integer value is advertised in the hello packets. The value must be the same for all routers and access servers attached to a common network. The default is 10 seconds. |
| <b>retransmit-interval</b> seconds | (Optional) Specifies the time (in seconds) between LSA retransmissions for adjacencies belonging to the interface. The value must be greater than the expected round-trip delay                                                                                                                               |



| Parameter                                       | Description                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                 | between any two routers on the attached network. The default is 5 seconds.                                                                                                                                                                                                                                                                                                        |
| <b>transmit-delay</b> <i>seconds</i>            | (Optional) Specifies the estimated time (in seconds) to send an LSU packet on the interface. This integer value must be greater than 0. LSAs in the update packet have their age incremented by this amount before transmission. The default value is 1 second.                                                                                                                   |
| <b>dead-interval</b> <i>seconds</i>             | (Optional) Specifies the time (in seconds) that must pass without hello packets being seen before a neighboring router declares the router down. This is an unsigned integer value. The default is four times the default hello interval, or 40 seconds. As with the hello interval, this value must be the same for all routers and access servers attached to a common network. |
| authentication-key <i>key</i>                   | (Optional) Specifies the password used by neighboring routers for simple password authentication. It is any continuous string of up to 8 characters. There is no default value.                                                                                                                                                                                                   |
| message-digest-key <i>key-id</i> md5 <i>key</i> | (Optional) Identifies the key ID and key (password) used between this router and neighboring routers for MD5 authentication. There is no default value.                                                                                                                                                                                                                           |

#### Note

OSPF authentication, including details of the *key* and *key-id* parameters, is described further in the "Configuring OSPF Authentication" section, later in this chapter.

The virtual link configuration must be done on the routers at each end of the virtual link. The `area area-id virtual-link` command requires the router ID of the far-end router. To find the router ID of the far-end router, use the `show ip ospf` command, `show ip ospf interface` command, or `show ip protocols` command on that remote router. Example 3-26 illustrates the output of the `show ip ospf` command, displaying the OSPF router ID.

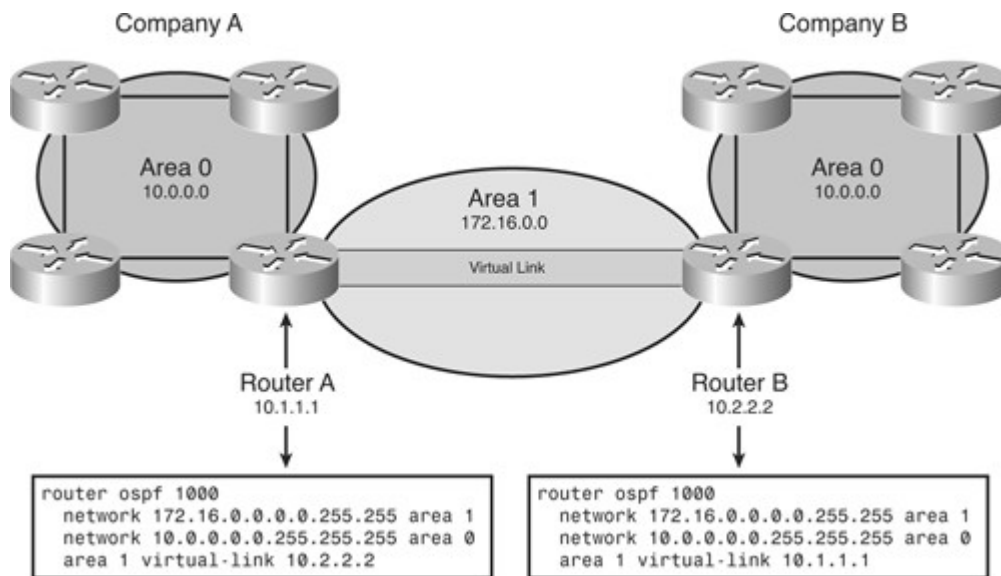
Example 3-26. Finding the OSPF Router ID for Use on a Virtual Link

```
remoterouter#show ip ospf
Routing Process "ospf 1000" with ID 10.2.2.2
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 It is an area border router
<output omitted>
```

In the example in Figure 3-43, area 0 is discontinuous. A virtual link is used as a backup strategy to temporarily connect area 0. Area 1 is used as the transit area. Router A builds a virtual link to Router B, and Router B builds a virtual link to the Router A. Each router points at the other router's router ID.

Figure 3-43. OSPF Virtual Link Configuration: Split Area 0.

[View full size image]



### Verifying OSPF Virtual Link Operation

The `show ip ospf virtual-links` command is used to verify OSPF virtual link operation. Example 3-27 provides the output of the `show ip ospf virtual-links` command on Router A in the example in Figure 3-43, verifying that the configured link works properly. In this example, the virtual link to Router B (with ID 10.2.2.2) is up. The virtual link uses transit area 1.

Example 3-27. `show ip ospf virtual-links` Command Output from Router A in Figure 3-43

```

RouterA#show ip ospf virtual-links
Virtual Link OSPF_VL0 to router 10.2.2.2 is up
 Run as demand circuit
 DoNotAge LSA allowed.
 Transit area 1, via interface Serial0/0/1, Cost of using 781
 Transmit Delay is 1 sec, State POINT_TO_POINT,
 Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
 Hello due in 00:00:07
 Adjacency State FULL (Hello suppressed)
 Index 1/2, retransmission queue length 0, number of retransmission 1
 First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
 Last retransmission scan length is 1, maximum is 1
 Last retransmission scan time is 0 msec, maximum is 0 msec
RouterA#

```

Table 3-21 describes some of the field of the output of the `show ip ospf virtual-links` command in detail.

Table 3-21. `show ip ospf virtual-links` Command Fields

| Field                                          | Description                                                                     |
|------------------------------------------------|---------------------------------------------------------------------------------|
| Virtual Link OSPF_VL0 to router 10.2.2.2 is up | Specifies the OSPF neighbor and whether the link to that neighbor is up or down |
| Transit area 1                                 | Specifies the transit area through which the virtual link is formed             |
| Via interface Serial0/0/1                      | Specifies the interface through which the virtual link is formed                |
| Cost of using 781                              | Specifies the cost of reaching the OSPF neighbor through                        |

| Field                      | Description                                                   |
|----------------------------|---------------------------------------------------------------|
|                            | the virtual link                                              |
| Transmit Delay is 1 sec    | Specifies the transmit delay on the virtual link              |
| State POINT_TO_POINT       | Specifies the state of the OSPF neighbor                      |
| Timer intervals configured | Specifies the various timer intervals configured for the link |
| Hello due in 0:00:07       | Specifies when the next hello is expected from the neighbor   |
| Adjacency State FULL       | Specifies the adjacency state between the neighbors           |

Routers across a virtual link become adjacent and exchange LSAs via the virtual link, similar to the process over a physical link.

Other commands that are useful when troubleshooting virtual links are **show ip ospf neighbor**, **show ip ospf database**, and **debug ip ospf adj**.

Example output from the show ip ospf neighbor command is provided in Example 3-28. Router A is in FULL state with Router B (with ID 10.2.2.2) on the virtual link.

Example 3-28. show ip ospf neighbor Command Output from Router A in Figure 3-43

```
RouterA#show ip ospf neighbor
Neighbor ID Pri State Dead Time Address Interface
10.200.200.13 1 FULL/DR 00:00:33 10.1.1.3 FastEthernet0/0
10.2.2.2 0 FULL/- - 172.16.1.2 OSPF_VL0
10.2.2.2 0 FULL/- 00:00:32 172.16.1.2 Serial0/0/1
RouterA#
```

Example output from the show ip ospf database command for router 10.2.2.2 is shown in Example 3-29. The LSAs learned through the virtual link have the DoNotAge option set.

Example 3-29. show ip ospf database Command Output from Router A in Figure 3-43

```
Code View: Scroll / Show All
RouterA#show ip ospf database router 10.2.2.2

 OSPF Router with ID (10.1.1.1) (Process ID 1000)

 Router Link States (Area 0)
Routing Bit Set on this LSA
LS age: 1 (DoNotAge)
Options: (No TOS-capability, DC)
LS Type: Router Links
Link State ID: 10.2.2.2
Advertising Router: 10.2.2.2
LS Seq Number: 80000003
Checksum: 0x8380
Length: 48
Area Border Router
Number of Links: 2

Link connected to: a Virtual Link
(Link ID) Neighboring Router ID: 10.1.1.1
```

(Link Data) Router Interface address: 172.16.1.2  
Number of TOS metrics: 0  
TOS 0 Metrics: 781

continues

Link connected to: a Transit Network  
(Link ID) Designated Router address: 10.1.2.2  
(Link Data) Router Interface address: 10.1.2.2  
Number of TOS metrics: 0  
TOS 0 Metrics: 1  
Router Link States (Area 1)

Routing Bit Set on this LSA  
LS age: 1688  
Options: (No TOS-capability, DC)  
LS Type: Router Links  
Link State ID: 10.2.2.2  
Advertising Router: 10.2.2.2  
LS Seq Number: 80000008  
Checksum: 0xCC81  
Length: 48  
Area Border Router  
Virtual Link Endpoint  
Number of Links: 2

Link connected to: another Router (point-to-point)  
(Link ID) Neighboring Router ID: 10.1.1.1  
(Link Data) Router Interface address: 172.16.1.2  
Number of TOS metrics: 0  
TOS 0 Metrics: 781

Link connected to: a Stub Network  
(Link ID) Network/subnet number: 172.16.1.0  
(Link Data) Network Mask: 255.255.255.0  
Number of TOS metrics: 0  
TOS 0 Metrics: 781

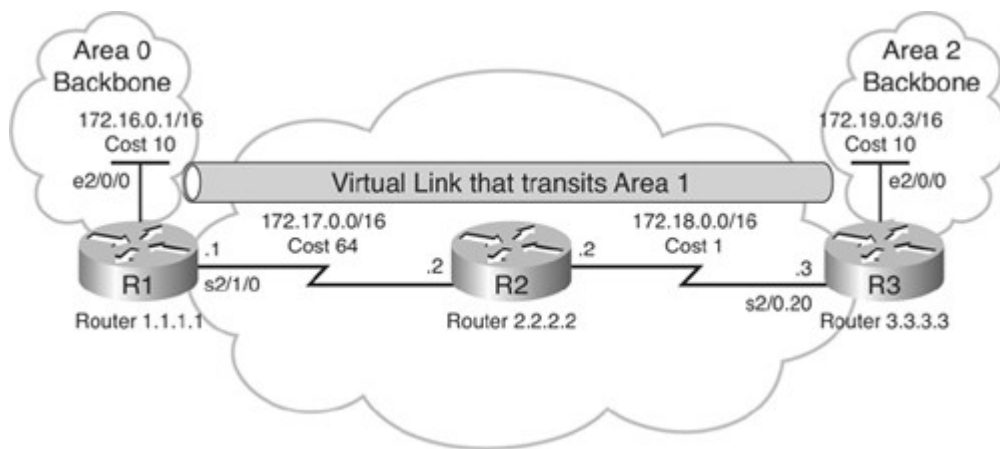
RouterA#

#### OSPF LSDB for Virtual Links

Figure 3-44 presents another example network. The configurations for Routers R1 and R3 are provided in Example 3-30.

Figure 3-44. OSPF Virtual Link Across Area 1.

[View full size image]



Example 3-30. Configurations for Routers R1 and R3 in Figure 3-44

Code View: Scroll / Show All

**R1#**

```
interface Loopback0
 ip address 1.1.1.1 255.0.0.0

interface Ethernet2/0/0
 ip address 172.16.0.1 255.255.0.0
interface Serial2/1/0
 ip address 172.17.0.1 255.255.0.0

router ospf 2
 network 172.16.0.0 0.0.255.255 area 0
 network 172.17.0.0 0.0.255.255 area 1
 area 1 virtual-link 3.3.3.3
```

**R3#**

```
interface Loopback0
 ip address 3.3.3.3 255.0.0.0

interface Ethernet0/0
 ip address 172.19.0.3 255.255.0.0

interface s2/0.20 point-to-point
 ip address 172.18.0.3 255.255.0.0

router ospf 2
 network 172.19.0.0 0.0.255.255 area 2
 network 172.18.0.0 0.0.255.255 area 1
 area 1 virtual-link 1.1.1.1
```

Example 3-31 illustrates output from the show ip ospf database command on Router R1. The router link states are type 1 LSAs and the summary net link states are type 3 LSAs, advertising routes from one area to another. Notice that LSAs learned through the virtual link have the DoNotAge (DNA) option. The virtual link is treated like a demand circuit.

Example 3-31. show ip ospf database Command Output on Router R1 in Figure 3-44

```
Code View: Scroll / Show All
R1#show ip ospf database
 OSPF Router with ID (1.1.1.1) (Process ID 2)

 Router Link States (Area 0)

Link ID ADV Router Age Seq# Checksum Link count
1.1.1.1 1.1.1.1 919 0x80000003 0xD5DF 2
3.3.3.3 3.3.3.3 5 (DNA) 0x80000002 0x3990 1

 Summary Net Link States (Area 0)

Link ID ADV Router Age Seq# Checksum
172.17.0.0 1.1.1.1 1945 0x80000002 0xAA48
172.17.0.0 3.3.3.3 9 (DNA) 0x80000001 0x7A70
172.18.0.0 1.1.1.1 1946 0x80000002 0xA749
172.18.0.0 3.3.3.3 9 (DNA) 0x80000001 0xEA3F
172.19.0.0 3.3.3.3 9 (DNA) 0x80000001 0xF624

 Router Link States (Area 1)

Link ID ADV Router Age Seq# Checksum Link count
1.1.1.1 1.1.1.1 1946 0x80000005 0xDDA6 2
2.2.2.2 2.2.2.2 10 0x80000009 0x64DD 4
3.3.3.3 3.3.3.3 930 0x80000006 0xA14C 2

 Summary Net Link States (Area 1)

Link ID ADV Router Age Seq# Checksum
172.16.0.0 1.1.1.1 1947 0x80000002 0x9990
172.16.0.0 3.3.3.3 911 0x80000001 0xEBF5
172.19.0.0 1.1.1.1 913 0x80000001 0xBF22
172.19.0.0 3.3.3.3 931 0x80000001 0xF624
```

Router R3 is an ABR because it has a link (the virtual link) to area 0. Therefore, Router R3 generates a summary LSA for 172.19.0.0 into area 1 and area 0, as illustrated in the show ip ospf database summary command output shown in Example 3-32.

Example 3-32. show ip ospf database summary Command Output on Router R3 in Figure 3-44

```
Code View: Scroll / Show All
R3#show ip ospf database summary 172.19.0.0

 OSPF Router with ID (3.3.3.3) (Process ID 2)
 Summary Net Link States (Area 0)

LS age: 1779
Options: (No TOS-capability, DC)
LS Type: Summary Links(Network)
```

```
Link State ID: 172.19.0.0 (summary Network Number)
Advertising Router: 3.3.3.3
LS Seq Number: 80000001
Checksum: 0xF624
Length: 28
Network Mask: /8
 TOS: 0 Metric: 10
```

#### Summary Net Link States (Area 1)

```
LS age: 1766
Options: (No TOS-capability, DC)
LS Type: Summary Links(Network)
Link State ID: 172.19.0.0 (summary Network Number)
Advertising Router: 1.1.1.1
LS Seq Number: 80000001
Checksum: 0xBF22
Length: 28
Network Mask: /8
 TOS: 0 Metric: 75
```

```
LS age: 1781
Options: (No TOS-capability, DC)
LS Type: Summary Links(Network)
Link State ID: 172.19.0.0 (summary Network Number)
Advertising Router: 3.3.3.3
LS Seq Number: 80000001
Checksum: 0xF624
Length: 28
Network Mask: /8
 TOS: 0 Metric: 10
```

#### Changing the Cost Metric

Recall that by default on Cisco routers, the OSPF metric for an interface is calculated according to the inverse of the interface's bandwidth. The default OSPF cost in Cisco routers is calculated using the formula  $(100) / (\text{bandwidth in Mbps})$ . This formula can also be written as  $(10^8) / (\text{bandwidth in bps})$ . The cost is a 16-bit value. The lower the cost the better the route is considered. For example, a 64-kbps link gets a metric of 1562, and a T1 link gets a metric of 64. However, this formula is based on a maximum bandwidth of 100 Mbps, which results in a cost of 1. If you have faster interfaces, you may want to recalibrate the cost of 1 to a higher bandwidth.

The **ip ospf cost**, **bandwidth**, and **auto-cost reference-bandwidth** commands can be used to manipulate the cost metric.

If interfaces that are faster than 100 Mbps are being used, use the **auto-cost reference-bandwidth ref-bw** router configuration command on all routers in the network to ensure accurate route calculations. The *ref-bw* parameter is the reference bandwidth in megabits per second. The range is from 1 to 4,294,967. The default is 100.

For example, in a network that has Fast Ethernet and Gigabit Ethernet interfaces, both would have a default OSPF cost of 1. In this case, the reference bandwidth could be changed to 10,000 Mbps using the **auto-cost reference-bandwidth 10000** command. The OSPF cost of a Fast Ethernet interface would then be 10,000/100



= 100 and the OSPF cost of a Gigabit Ethernet interface would be  $10,000/1000 = 10$ . Thus, the interface costs would be differentiated.

When using the interface's bandwidth to determine OSPF cost, always remember to use the **bandwidth value** interface configuration command to accurately define the bandwidth per interface, in kilobits per second.

To override the default cost, manually define the cost using the **ip ospf cost interface-cost** configuration command on a per-interface basis. The *interface-cost* is an integer from 1 to 65,535. The lower the number, the better (and more preferred) the link.

#### Configuring OSPF Special Area Types

As discussed, OSPF is based on a two-level hierarchical area structure with backbone and nonbackbone areas. Each area has its own topology database, which is invisible from outside the area. A router belonging to several areas (an ABR) has several topology databases, one per area to which it is attached. All areas have to be connected to a backbone area or linked to it with a virtual link. The backbone area has to be contiguous. A nonbackbone area can be discontinuous. This section describes special area types that can be configured for OSPF.

The characteristics assigned to an area control the type of route information it receives. The purpose behind any type of stub area is to inject default routes into an area so that external and/or summary LSAs are not flooded into the area. This reduces the LSDB size and the routing table size in the routers within the area.

The possible area types, some of which are shown in Figure 3-45, are as follows:

- **Standard area**— This default area type accepts link updates, route summaries, and external routes.
- **Backbone area**— The backbone area is labeled area 0, and all other areas connect to this area to exchange and route information. The OSPF backbone has all the properties of a standard OSPF area.
- **Stub area**— This area type does not accept information about routes external to the autonomous system, such as routes from non-OSPF sources. If routers need to route to networks outside the autonomous system, they use a default route, indicated as 0.0.0.0. Stub areas cannot contain ASBRs (except that the ABRs may also be ASBRs).
- **Totally stubby area**— This Cisco proprietary area type does not accept external autonomous system routes or summary routes from other areas internal to the autonomous system. If a router needs to send a packet to a network external to the area, it sends the packet using a default route. Totally stubby areas cannot contain ASBRs (except that the ABRs may also be ASBRs).
- **NSSA**— NSSA is an addendum to the OSPF RFC. This area type defines a special LSA type 7. NSSA offers benefits that are similar to those of a stub area. They do not accept information about routes external to the autonomous system, but instead use a default route for external networks. However, NSSAs allow ASBRs, which is against the rules in a stub area.
- **Totally stubby NSSA**— Cisco routers also allow an area to be configured as a totally stubby NSSA, which allows ASBRs, but does not accept external routes or summary routes from other areas. A default route is used to get to networks outside of the area.

Figure 3-45. Some Types of OSPF Areas.

[View full size image]

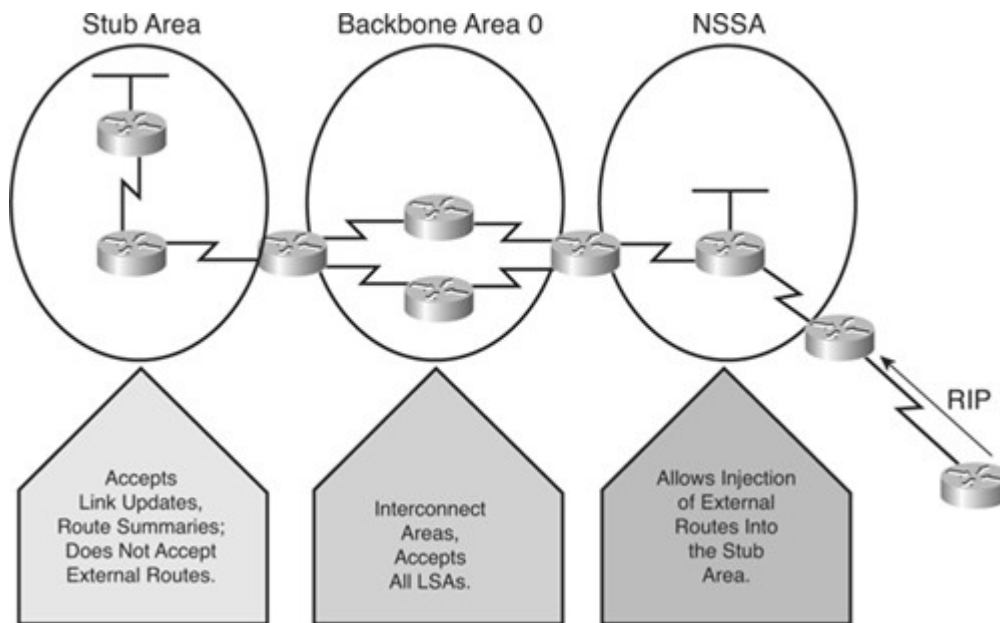


Table 3-22 summarizes the OSPF area types. Three of the column headers in this table also indicate how the various types of routes are indicated in the routing table.

Table 3-22. OSPF Area Types

| Area Type           | Accepts Routes Within Area (O) | Accepts Routes from Other Areas (O IA) | Accepts External Routes (O E1 and O E2) | Allows ASBR | Cisco Proprietary |
|---------------------|--------------------------------|----------------------------------------|-----------------------------------------|-------------|-------------------|
| Standard            | Yes                            | Yes                                    | Yes                                     | Yes         | No                |
| Backbone            | Yes                            | Yes                                    | Yes                                     | Yes         | No                |
| Stub                | Yes                            | Yes                                    | No (uses interarea default route)       | No          | No                |
| Totally stubby      | Yes                            | No (uses interarea default route)      | No (uses interarea default route)       | No          | Yes               |
| NSSA                | Yes                            | Yes                                    | No (uses interarea default route)       | Yes         | No                |
| Totally stubby NSSA | Yes                            | No (uses interarea default route)      | No (uses interarea default route)       | Yes         | Yes               |

Routers within stub and totally stub areas do not have any external routes (type 5 LSAs).

#### Note

Because type 5 LSAs, describing external routes from ASBRs, are not sent into stub areas, type 4 LSAs, which describe the path to the ASBR, are also not sent into stub areas. This fact is not explicitly stated in some Cisco documentation, but it is in the OSPF RFC and we confirmed it by testing.

An area qualifies as stub or totally stubby area if it has the following characteristics:

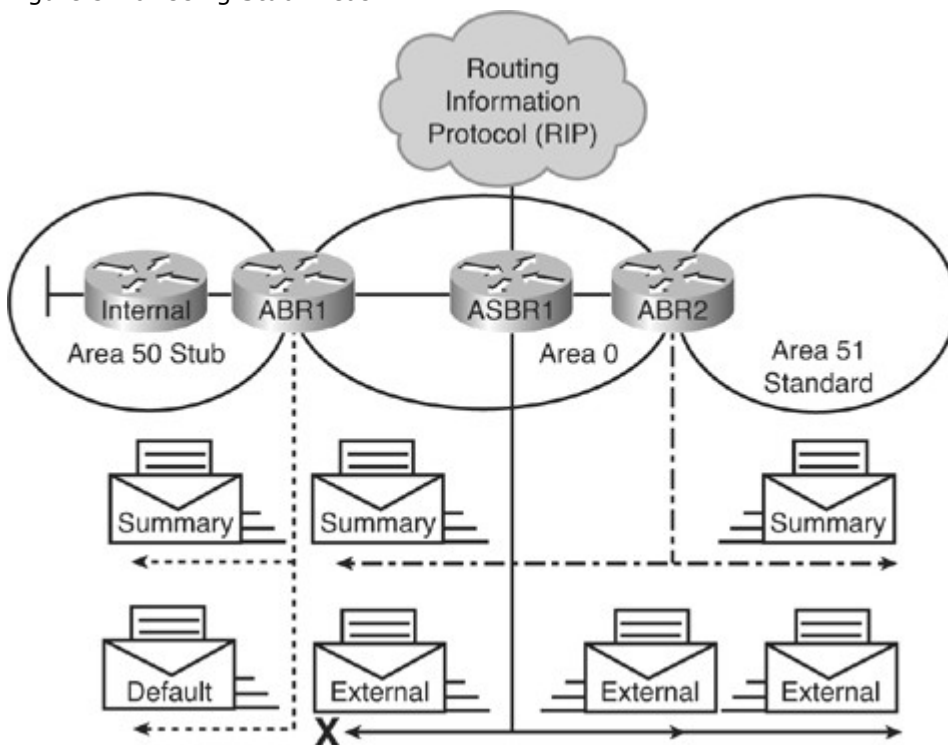
- There is a single exit point from that area; or if there are multiple exits, one or more ABRs inject a default route into the stub area and suboptimal routing paths are acceptable. In other words, it is acceptable if routing to other areas or autonomous systems can take a suboptimal path to reach the destination by exiting the area via a point that is farther from the destination than other exit points.

- All OSPF routers inside the stub area, including ABRs, are configured as stub routers. All of these routers must be configured as stub routers before they can become neighbors and exchange routing information.
- The area is not used as a transit area for virtual links.
- There is no ASBR inside the area.
- The area is not the backbone area (area 0).

#### Configuring Stub Areas

Configuring a stub area reduces the size of the LSDB inside an area, resulting in reduced memory requirements for routers in that area. Routers within the stub area also do not have to run the SPF algorithm as often because they will receive fewer routing updates. External network LSAs (type 5), such as those redistributed from other routing protocols into OSPF, are not permitted to flood into a stub area, as shown in Figure 3-46. (Type 4 LSAs are also not flooded, as described earlier.) Routing from these areas to a route external to the OSPF autonomous system is based on a default route (0.0.0.0). If a packet is addressed to a network that is not in the routing table of an internal router, the router automatically forwards the packet to the ABR that originates a 0.0.0.0 LSA. Forwarding the packet to the ABR allows routers within the stub area to reduce the size of their routing tables, because a single default route replaces many external routes.

Figure 3-46. Using Stub Areas.



A stub area is typically created using a hub-and-spoke topology, with a spoke being a stub area, such as a branch office. In this case, the branch office does not need to know about every network at the headquarters site, because it can use a default route to reach the networks.

To configure an area as a stub, use the following procedure:

- Step 1. Configure OSPF on all routers within the area.
- Step 2. Define an area as stub by adding the **area area-id stub** router configuration command to all routers within the area. The *area-id* parameter is the identifier for the stub area. The identifier can be either a decimal value or a value in dotted-decimal format, like an IP address.
- Step 3. Optionally configure the cost for the default route on the ABR.

#### Note

The hello packet exchanged between OSPF routers contains a stub area flag that must match on neighboring routers. The **area area-id stub** command must be enabled on all routers in the stub area so that they all have the stub flag set. The routers can then become neighbors and exchange routing information.

By default, the ABR of a stubby or totally stubby area advertises a default route with a cost of 1. To change the cost of the default route, use the `area area-id default-cost cost` router configuration command. This command is configured only on the ABR. The parameters of this command are shown in Table 3-23

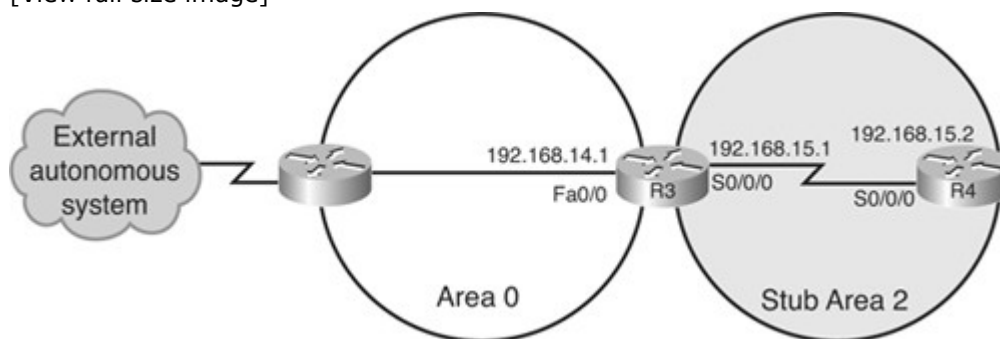
Table 3-23. *area default-cost* Command Parameters

| Parameter | Description                                                                                                                                                           |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| area-id   | The identifier for the stub area, totally stubby area, or NSSA. The identifier can be either a decimal value or a value in dotted-decimal format, like an IP address. |
| cost      | Cost for the default summary route. The acceptable values are 0 through 16777215. The default is 1.                                                                   |

Figure 3-47 illustrates an example. Area 2 is defined as the stub area. No routes from the external autonomous system are forwarded into the stub area. Example 3-33 shows the OSPF configuration on Routers R3 and R4, including enabling an OSPF stub area.

Figure 3-47. OSPF Stub Area Example.

[View full size image]



Example 3-33. OSPF Stub Area Configuration for Routers R3 and R4 in Figure 3-47

```

Router R3:
R3(config)#interface FastEthernet0/0
R3(config-if)#ip address 192.168.14.1 255.255.255.0
R3(config)#interface Serial 0/0/0
R3(config-if)#ip address 192.168.15.1 255.255.255.252
R3(config)#router ospf 100
R3(config-router)#network 192.168.14.0.0 0.0.0.255 area 0
R3(config-router)#network 192.168.15.0.0 0.0.0.255 area 2
R3(config-router)#area 2 stub

Router R4:
R4(config)#interface Serial 0/0/0
R4(config-if)#ip address 192.168.15.2 255.255.255.252

R4(config)#router ospf 100
R4(config-router)#network 192.168.15.0.0 0.0.0.255 area 2
R4(config-router)#area 2 stub

```

The last line in each router configuration (**area 2 stub**) defines the stub area. The R3 router (the ABR) automatically advertises 0.0.0.0 (the default route) with a default cost metric of 1 into the stub area. Each router in the stub area must be configured with the **area stub** command.

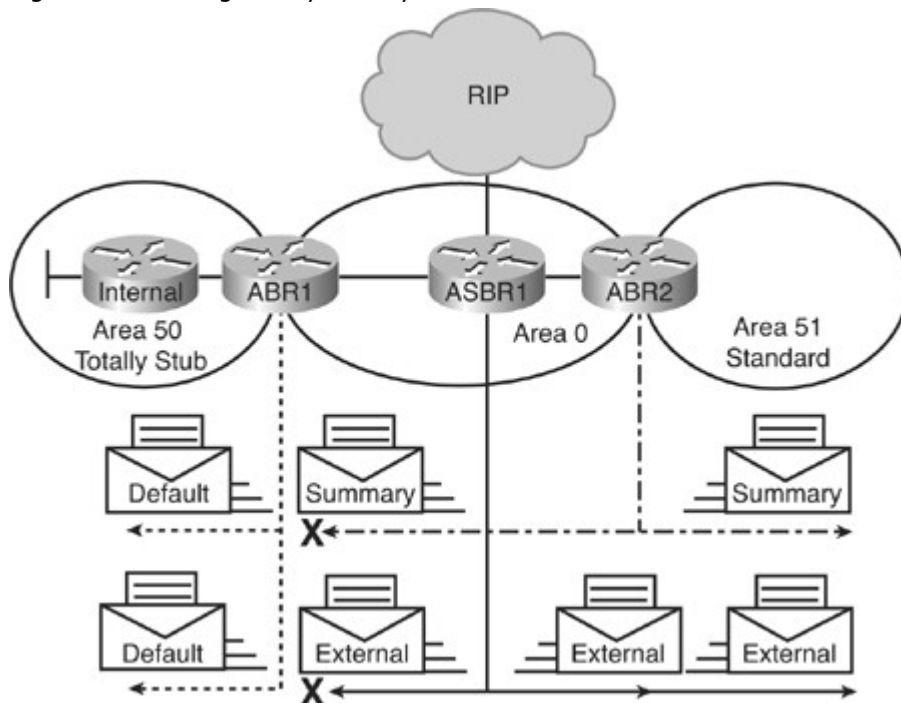
The routes that appear in the routing table of router R4 (the internal router) are as follows:

- Intra-area routes, which are designated with an O in the routing table.
- The default route and interarea routes, which are both designated with an IA in the routing table. The default route is also denoted with an asterisk (O\*IA).

#### Configuring Totally Stubby Areas

A totally stubby area is a Cisco-specific feature that further reduces the number of routes in the routing table. A totally stubby area blocks external type 5 LSAs and summary type 3 and type 4 LSAs (interarea routes) from entering the area, as shown in Figure 3-48. By blocking these routes, the totally stubby area recognizes only intra-area routes and the default route 0.0.0.0. ABRs inject the default summary link 0.0.0.0 into the totally stubby area. Each router within the area picks the closest ABR as a gateway to everything outside the area.

Figure 3-48. Using Totally Stubby Areas.



Totally stubby areas minimize routing information further than stub areas and increase stability and scalability of OSPF internetworks. Using totally stubby areas is typically a better solution than using stub areas, assuming the ABR is a Cisco router.

To configure an area as totally stubby, do the following:

- Step 1. Configure OSPF on all routers in the area.
- Step 2. Define an area as a stub area by adding the **area area-id stub** router configuration command to all routers in the area.
- Step 3. At the ABR only, add the **no-summary** parameter to the **area area-id stub** command. This makes the area totally stubby.
- Step 4. Optionally configure the cost for the default route on the ABR.

Table 3-24 explains the **area area-id stub no-summary** command.

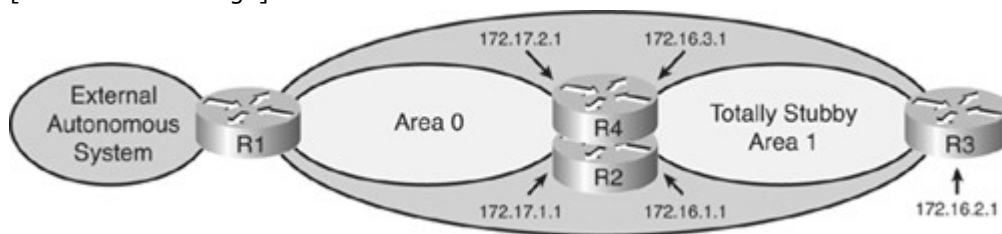
Table 3-24. *area area-id stub no-summary* Command Parameters

| Parameter  | Description                                                                                                                                   |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| area-id    | The identifier for the stub or totally stubby area. It can be either a decimal value or a value in dotted-decimal format, like an IP address. |
| no-summary | Stops summary LSAs, in addition to external LSAs, from flooding into the totally stubby area.                                                 |

Figure 3-49 shows an example of a totally stubby area topology. The configurations on routers R2, R3, and R4 are shown in Example 3-34. All routes advertised into area 1 (from area 0 and the external autonomous system) default to 0.0.0.0. The default route cost is set to 5 on R2 and to 10 on R4. Both default routes are advertised into area 1. However, the default route from R2 is advertised with a lower cost to make it more preferable if the internal cost from R3 to R4 is the same as the internal cost from R3 to R2.

Figure 3-49. Totally Stubby Example.

[View full size image]



Example 3-34. Totally Stubby Configuration for Routers in Figure 3-49

```

Router R2:
R2(config)#router ospf 10
R2(config-router)#network 172.17.0.0 0.0.255.255 area 0
R2(config-router)#network 172.16.0.0 0.0.255.255 area 1
R2(config-router)#area 1 stub no-summary
R2(config-router)#area 1 default-cost 5

Router R3:
R3(config)#router ospf 10
R3(config-router)#network 172.16.0.0 0.0.255.255 area 1
R3(config-router)#area 1 stub

Router R4:
R4(config)#router ospf 10
R4(config-router)#network 172.17.0.0 0.0.255.255 area 0
R4(config-router)#network 172.16.0.0 0.0.255.255 area 1
R4(config-router)#area 1 stub no-summary
R4(config-router)#area 1 default-cost 10

```

Remember that all routers in a stub or totally stubby area must be configured as stubs. An OSPF adjacency will not form between stub and nonstub routers. Notice that R3 requires the **area 1 stub** command, yet the **no-summary** keyword is not required. Only ABRs (R2 and R4) use the **no-summary** keyword to keep summary LSAs from being propagated into another area.

## Interpreting Routing Tables in Different Types of OSPF Areas

This section illustrates routing tables when different area types are configured.

Example 3-35 shows how the routing table of an OSPF router in a standard area (without any kind of stub configuration) might look. Intra-area (O), interarea (O IA), and external routes (O E1 and O E2) are all maintained in a standard area.

### Example 3-35. Routing Table in a Standard Area

```
RouterA#show ip route
<output omitted>

Gateway of last resort is not set
 172.31.0.0/32 is subnetted, 4 subnets
O IA 172.31.22.4 [110/782] via 10.1.1.1, 00:02:44, FastEthernet0/0
O IA 172.31.11.1 [110/1] via 10.1.1.1, 00:02:44, FastEthernet0/0
O IA 172.31.11.2 [110/782] via 10.1.3.4, 00:02:52, Serial0/0/0
 [110/782] via 10.1.1.1, 00:02:52, FastEthernet0/0
O IA 172.31.11.4 [110/782] via 10.1.1.1, 00:02:44, FastEthernet0/0
 10.0.0.0/8 is variably subnetted, 7 subnets, 2 masks
O 10.11.0.0/24 [110/782] via 10.1.1.1, 00:03:22, FastEthernet0/0
C 10.200.200.13/32 is directly connected, Loopback0
C 10.1.3.0/24 is directly connected, Serial0/0/0
O 10.1.2.0/24 [110/782] via 10.1.3.4, 00:03:23, Serial0/0/0
C 10.1.1.0/24 is directly connected, FastEthernet0/0
O 10.1.0.0/24 [110/782] via 10.1.1.1, 00:03:23, FastEthernet0/0
O E2 10.254.0.0/24 [110/50] via 10.1.1.1, 00:02:39, FastEthernet0/0
RouterA#
```

Example 3-36 shows how the same routing table looks if the area is configured as a stub area. Intra-area (O) and interarea (O IA) routes are all maintained. However, external routes (O E1 and O E2) are not visible in the routing table. These routes are accessible via the interarea default route (O\*IA), as shown in the last line of the routing table.

### Example 3-36. Routing Table in a Stub Area

```
RouterA#show ip route
<output omitted>

Gateway of last resort is 10.1.1.1 to network 0.0.0.0
 172.31.0.0/32 is subnetted, 4 subnets
O IA 172.31.22.4 [110/782] via 10.1.1.1, 00:01:49, FastEthernet0/0
O IA 172.31.11.1 [110/1] via 10.1.1.1, 00:01:49, FastEthernet0/0
O IA 172.31.11.2 [110/782] via 10.1.3.4, 00:01:49, Serial0/0/0
 [110/782] via 10.1.1.1, 00:01:49, FastEthernet0/0
O IA 172.31.11.4 [110/782] via 10.1.1.1, 00:01:49, FastEthernet0/0
 10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O 10.11.0.0/24 [110/782] via 10.1.1.1, 00:01:50, FastEthernet0/0
C 10.200.200.13/32 is directly connected, Loopback0
C 10.1.3.0/24 is directly connected, Serial0/0/0
O 10.1.2.0/24 [110/782] via 10.1.3.4, 00:01:50, Serial0/0/0
C 10.1.1.0/24 is directly connected, FastEthernet0/0
O 10.1.0.0/24 [110/782] via 10.1.1.1, 00:01:50, FastEthernet0/0
O*IA 0.0.0.0/0 [110/2] via 10.1.1.1, 00:01:51, FastEthernet0/0
RouterA#
```



Example 3-37 shows how the same routing table looks if summarization is performed on the ABR. The area is still configured as a stub area, and intra-area (O) and summarized interarea (O IA) routes are all maintained. In this example the two routes 172.31.11.1 and 172.31.11.2 were summarized to 172.31.11.0/24. External routes are not visible in the routing table but are accessible via the interarea default route (O\*IA).

Example 3-37. Routing Table in a Stub Area with Summarization

```
RouterA#show ip route
<output omitted>

Gateway of last resort is 10.1.1.1 to network 0.0.0.0
 172.31.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA 172.31.22.4/32 [110/782] via 10.1.1.1, 00:13:08, FastEthernet0/0
O IA 172.31.11.0/24 [110/1] via 10.1.1.1, 00:02:39, FastEthernet0/0
 10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O 10.11.0.0/24 [110/782] via 10.1.1.1, 00:13:08, FastEthernet0/0
C 10.200.200.13/32 is directly connected, Loopback0
C 10.1.3.0/24 is directly connected, Serial0/0/0
O 10.1.2.0/24 [110/782] via 10.1.3.4, 00:13:09, Serial0/0/0
C 10.1.1.0/24 is directly connected, FastEthernet0/0
O 10.1.0.0/24 [110/782] via 10.1.1.1, 00:13:09, FastEthernet0/0
O*IA 0.0.0.0/0 [110/2] via 10.1.1.1, 00:13:09, FastEthernet0/0
RouterA#
```

Example 3-38 shows how the same routing table looks if the area is configured as a totally stubby area. Notice that routers in the totally stubby area have the smallest routing tables. Intra-area routes (O) are maintained. Individual interarea (O IA) and external (O E1 and O E2) routes are not visible in the routing table but are accessible via the interarea (O\*IA) default route.

Example 3-38. Routing Table in a Totally Stubby Area

```
RouterA#show ip route
<output omitted>

Gateway of last resort is 10.1.1.1 to network 0.0.0.0
 10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O 10.11.0.0/24 [110/782] via 10.1.1.1, 00:16:53, FastEthernet0/0
C 10.200.200.13/32 is directly connected, Loopback0
C 10.1.3.0/24 is directly connected, Serial0/0/0
O 10.1.2.0/24 [110/782] via 10.1.3.4, 00:16:53, Serial0/0/0
C 10.1.1.0/24 is directly connected, FastEthernet0/0
O 10.1.0.0/24 [110/782] via 10.1.1.1, 00:16:53, FastEthernet0/0
O*IA 0.0.0.0/0 [110/2] via 10.1.1.1, 00:00:48, FastEthernet0/0
RouterA#
```

### Configuring NSSAs

The OSPF NSSA feature is described by RFC 3101 and was introduced in Cisco IOS Software Release 11.2. It is a nonproprietary extension of the existing stub area feature that allows the injection of external routes in a limited fashion into the stub area.

Redistribution into an NSSA area creates a special type of LSA known as type 7, which can exist only in an NSSA area. An NSSA ASBR generates this LSA, and an NSSA ABR translates it into a type 5 LSA, which gets propagated into the OSPF domain. Type 7 LSAs have a propagate (P) bit in the LSA header to prevent propagation loops between the NSSA and the backbone area. Type 7 LSAs are the same format as type 5 LSAs.

The NSSA feature allows an area to retain the other stub area features—the ABR sends a default route into the NSSA instead of external routes from other ASBRs—while also allowing an ASBR to be inside of the area. Recall that one of the rules of stub areas is that there must not be an ASBR inside of a stub area. An NSSA—a not-so-stubby area—bends this rule. Figure 3-50 illustrates an NSSA.

Figure 3-50. NSSA.

[View full size image]



Routers operating in NSSA areas set the N-bit to signify that they can support the type 7 LSA. These option bits are checked during neighbor establishment and must match for an adjacency to form.

The link-state ID in the type 7 LSA is the external network number. As is the case for type 5 LSAs, because of the flooding scope and depending on the number of external networks, the default lack of route summarization can be a major issue with these LSAs. Therefore, the network administrator should always attempt to summarize blocks of external network numbers at the ASBR to reduce flooding problems.

The type 7 LSA is described in the routing table as an O N2 or O N1 (N means NSSA). N1 means that the metric is calculated like external type 1 (internal costs are added to the external metric); N2 means that the metric is calculated like external type 2 (internal costs are not added to the external metric). The default is O N2.

To configure an area as an NSSA, do the following:

Step 1. Configure OSPF on all routers in the area.

Step 2. Define an area as an NSSA by adding **area area-id nssa [no-redistribution] [default-information-originate] [metric metric-value] [metric-type type-value] [no-summary]** router configuration command, instead of the **area area-id stub** command, to all routers in the area. Do not use the **no-summary** keyword for NSSA areas (it is used only on the ABR of totally stubby NSSA areas).

Step 3. Optionally configure the cost for the default route on the ABR.

Remember that all routers in the NSSA must have this command configured. Two routers will not form an adjacency unless both are configured as NSSA.

Table 3-25 defines the parameters of the area nssa command.

Table 3-25. *area area-id nssa* Command Parameters

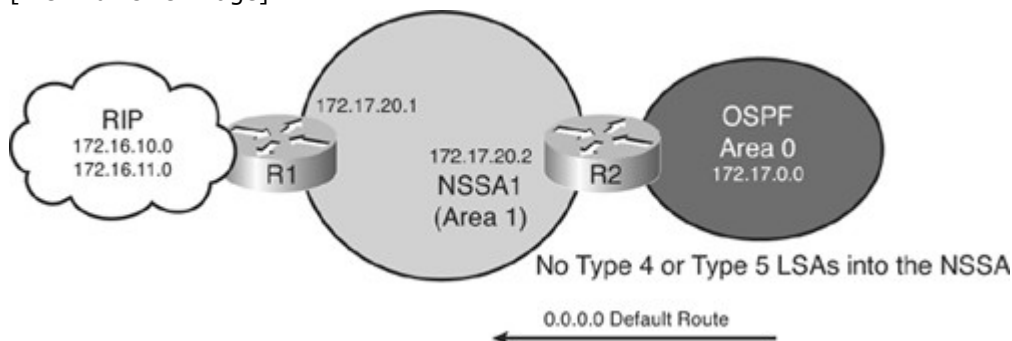
| Parameter                     | Description                                                                                                                                                     |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| area-id                       | The identifier for the NSSA. It can be either a decimal value or a value in dotted-decimal format, like an IP address.                                          |
| no-redistribution             | (Optional) Used when the router is an NSSA ABR and you want the redistribute command to import routes only into the standard areas, but not into the NSSA area. |
| default-information-originate | (Optional) Used to generate a type 7 default LSA into the NSSA area. This keyword takes effect only on an NSSA ABR or an NSSA ASBR.                             |
| <b>metric metric-value</b>    | (Optional) Metric that is used for generating the default route. Acceptable values are 0 through 16777214.                                                      |
| <b>metric-type type-value</b> | (Optional) OSPF metric type for default routes. It can be one of                                                                                                |

|            |                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------|
| Parameter  | Description                                                                                                                     |
|            | the following values:<br>1: type 1 external route<br>2: type 2 external route                                                   |
| no-summary | (Optional) Allows an area to be a totally stubby NSSA, which is like an NSSA but does not have summary routes injected into it. |

In Figure 3-51 and Example 3-39, R1 is the ASBR that redistributes RIP routes into area 1, the NSSA. R2 is the NSSA ABR. The NSSA ABR converts type 7 LSAs into type 5 LSAs for advertisement into backbone area 0. R2 is also configured to summarize the type 5

Figure 3-51. NSSA Example.

[View full size image]



LSAs that originate from the RIP network—the 172.16.0.0 subnets—to 172.16.0.0/16 and advertise this summary route into area 0. To cause R2 (the NSSA ABR) to generate an O\*N2 default route (O\*N2 0.0.0.0/0) into the NSSA, the **default-information-originate** parameter is used on the **area area-id nssa** command on R2.

Example 3-39. OSPF NSSA Configuration for Routers in Figure 3-51

```
Router R1:
R1(config)#router ospf 10
R1(config-router)#redistribute rip subnets
R1(config-router)#default metric 150
R1(config-router)#network 172.17.0.0 0.0.255.255 area 1
R1(config-router)#area 1 nssa

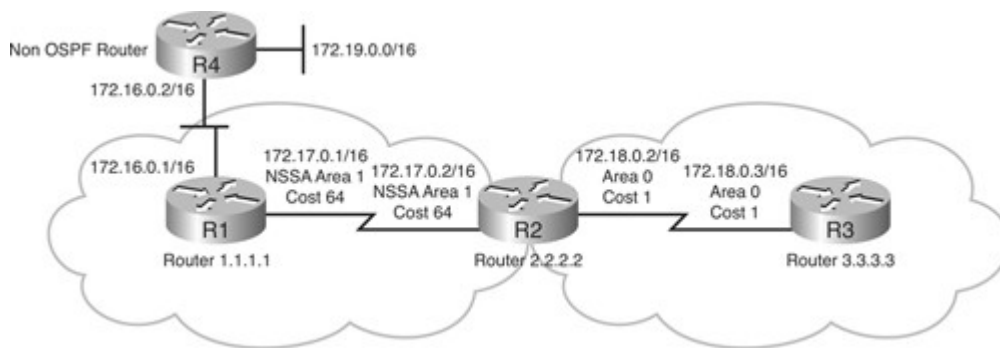
Router R2:
R2(config)#router ospf 10
R2(config-router)#summary-address 172.16.0.0
255.255.0.0
R2(config-router)#network 172.17.20.0 0.0.0.255 area 1
R2(config-router)#network 172.17.0.0 0.0.255.255 area 0
R2(config-router)#area 1 nssa default-information-originate
```

OSPF NSSA LSDB

Figure 3-52 illustrates another example network, used to illustrate an NSSA LSDB. The configuration of Routers R1, R2, and R3 are provided in Example 3-40.

Figure 3-52. OSPF NSSA Example Network.

[View full size image]



Example 3-40. OSPF NSSA Configuration for Routers in Figure 3-52

Code View: Scroll / Show All

R1#:

```
interface Loopback0
 ip address 1.1.1.1 255.0.0.0
interface Serial2/1/0
 ip address 172.17.0.1 255.255.0.0
interface Ethernet2/0/0
 ip address 172.16.0.1 255.255.0.0
router ospf 4
 redistribute static metric 5 metric-type 1
 network 172.17.0.0 0.0.255.255 area 1
 network 172.16.0.0 0.0.255.255 area 1
 area 1 nssa
 ip route 172.19.0.0 255.255.0.0 172.16.0.2
```

R2#:

```
interface Loopback0
 ip address 2.2.2.2 255.0.0.0
interface Serial0/1/0
 ip address 172.17.0.2 255.255.0.0
interface s1/0.20 point-to-point
 ip address 172.18.0.2 255.255.0.0
router ospf 2
 network 172.17.0.0 0.0.255.255 area 1
 network 172.18.0.0 0.0.255.255 area 0
 area 1 nssa
```

R3#

```
interface Loopback0
 ip address 3.3.3.3 255.0.0.0
interface s2/0.20 point-to-point
 ip address 172.18.0.3 255.255.0.0
router ospf 2
 network 172.18.0.0 0.0.255.255 area 0
```

Area 1 is configured as an NSSA. Router R1 is an ASBR and is redistributing a static route into the OSPF area. Router R2 is the ABR, and is connected to the NSSA area 1 and to the backbone area 0.

Example 3-41 is the output of the show ip ospf database command on the R2 router, the ABR. The router link states are type 1 LSAs. The summary net link states are type 3 LSAs, advertising routes from one area into another. To advertise external routes into an NSSA, the ASBR Router R1 creates a type 7 LSA. The ABR converts the type 7 LSA into a type 5 LSA and propagates the type 5 LSA into other areas.

Example 3-41. show ip ospf database Output on Router R2 in Figure 3-52

Code View: Scroll / Show All

R2#**show ip ospf database**

OSPF Router with ID (2.2.2.2) (Process ID 2)

Router Link States (Area 0)

| Link ID | ADV Router | Age  | Seq#       | Checksum | Link count |
|---------|------------|------|------------|----------|------------|
| 2.2.2.2 | 2.2.2.2    | 1235 | 0x8000001D | 0xD9FF   | 2          |
| 3.3.3.3 | 3.3.3.3    | 1100 | 0x8000000B | 0x9455   | 2          |

Summary Net Link States (Area 0)

| Link ID    | ADV Router | Age  | Seq#       | Checksum |
|------------|------------|------|------------|----------|
| 172.16.0.0 | 2.2.2.2    | 1979 | 0x80000002 | 0xFDE7   |
| 172.17.0.0 | 2.2.2.2    | 1483 | 0x80000004 | 0x8864   |

Router Link States (Area 1)

| Link ID | ADV Router | Age | Seq#       | Checksum | Link count |
|---------|------------|-----|------------|----------|------------|
| 1.1.1.1 | 1.1.1.1    | 319 | 0x8000000C | 0xAFA8   | 3          |
| 2.2.2.2 | 2.2.2.2    | 220 | 0x8000002F | 0xD478   | 2          |

Summary Net Link States (Area 1)

| Link ID    | ADV Router | Age  | Seq#       | Checksum |
|------------|------------|------|------------|----------|
| 172.18.0.0 | 2.2.2.2    | 1483 | 0x8000001C | 0x7894   |

Type-7 AS External Link States (Area 1)

| Link ID    | ADV Router | Age | Seq#       | Checksum | Tag |
|------------|------------|-----|------------|----------|-----|
| 172.19.0.0 | 1.1.1.1    | 334 | 0x80000005 | 0xD738   | 0   |

Type-5 AS External Link States

| Link ID    | ADV Router | Age  | Seq#       | Checksum | Tag |
|------------|------------|------|------------|----------|-----|
| 172.19.0.0 | 2.2.2.2    | 1725 | 0x80000004 | 0x50C6   | 0   |

Notice that ASBR summary LSAs (type 4) are not used in this case because the ABR, not the ASBR, originates the external type 5 LSA, and the ABR is reachable within area 0. In contrast if the area was a standard area, the ASBR would originate the type 5 LSA and the ABR would create a type 4 LSA describing how other routers can reach the ASBR.

### Configuring Totally Stubby NSSAs

The OSPF totally stubby NSSA feature is a Cisco proprietary extension to NSSA that blocks type 3, 4, and 5 LSAs. A single default route replaces both inbound external (type 5) LSAs and summary (type 3 and 4) LSAs into the totally NSSA area.

The ABR of a totally stubby NSSA must be configured with the **no-summary** keyword to prevent the flooding of summary routes for other areas into the NSSA area.

To configure an area as a totally stubby NSSA, do the following:

Step 1. Configure OSPF on all routers in the area.

Step 2. Define an area as an NSSA by adding **area area-id nssa** [**no-redistribution**] [**default-information-originate**] [**metric metric-value**] [**metric-type type-value**] [**no-summary**] router configuration command, instead of the **area area-id stub** command, to all routers in the area. Do not use the **no-summary** keyword on the routers within the area.

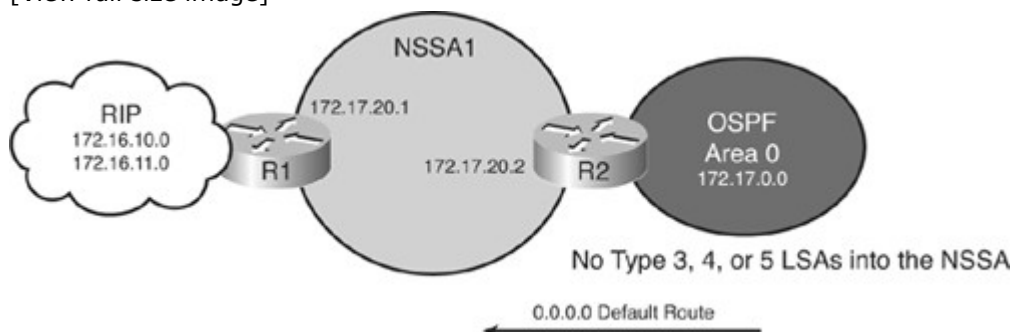
Step 3. On the ABR only, add the **no-summary** keyword to the **area nssa** command.

Step 4. Optionally configure the cost for the default route on the ABR.

In the example in Figure 3-53 and Example 3-42, notice that the ABR, R2, is using the area 1 nssa no-summary command. This command works exactly the same as the totally stubby technique. A single default route replaces both inbound external (type 5) LSAs and summary (type 3 and 4) LSAs into the area. The NSSA ABR, Router R2, automatically generates the O\*N2 default route into the NSSA area when the no-summary option is configured at the ABR, so the default-information-originate parameter is not required.

Figure 3-53. NSSA Totally Stubby.

[View full size image]



Example 3-42. NSSA Totally Stubby Configuration for Routers in Figure 3-53

Router R1:

```
R1(config)#router ospf 10
R1(config-router)#redistribute rip subnets
R1(config-router)#default metric 150
R1(config-router)#network 172.17.0.0 0.0.255.255 area 1
```

```
R1(config-router)#area 1 nssa
```

Router R2:

```
R2(config)#router ospf 10
R2(config-router)#summary-address 172.16.0.0 255.255.0.0
R2(config-router)#network 172.17.20.0 0.0.0.255 area 1
R2(config-router)#network 172.17.0.0 0.0.255.255 area 0
R2(config-router)#area 1 nssa no-summary
```

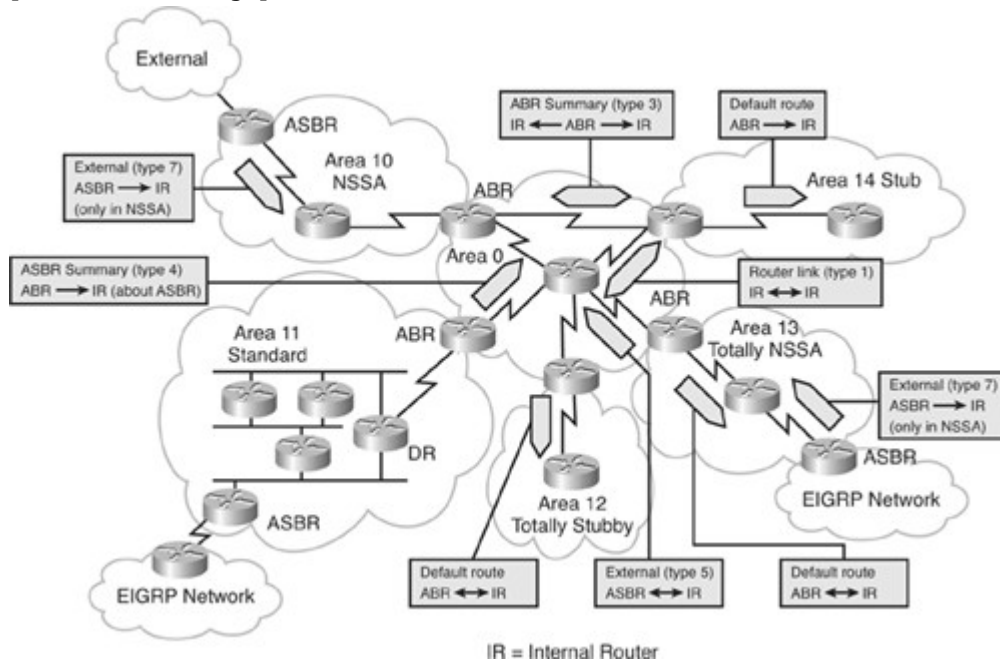
All other routers in the NSSA area require the **area 1 nssa** command only. The NSSA totally stubby configuration is a Cisco-specific feature, just as the totally stubby area feature is.

Example OSPF Area Types in a Network

Figure 3-54 illustrates a network with a variety of area types as follows:

Figure 3-54. Example of Different OSPF Area Types.

[View full size image]



- Standard area 11 accepts link updates, summaries, and external routes.
- Stub area 14 does not accept type 4 summary or type 5 external LSAs, but does accept type 3 summary LSAs.
- Totally stubby area 12 does not accept summary or external LSAs.
- NSSA area 10 does not accept type 4 summary or type 5 external LSAs, but does accept type 3 summary LSAs and allows an ASBR.
- Totally stubby NSSA area 13 does not accept summary or external LSAs, but allows an ASBR.

#### Verifying All Area Types

The show commands in Table 3-26 are used to display the area type that has been configured and other information about the area.

Table 3-26. *show* Commands for All Area Types

| Command                             | Description                                                                   |
|-------------------------------------|-------------------------------------------------------------------------------|
| show ip ospf                        | Displays OSPF information,, including which areas are standard, stub, or NSSA |
| show ip ospf database               | Displays details of LSAs                                                      |
| show ip ospf database nssa-external | Displays specific details of each LSA type 7 update in the database           |



| Command       | Description         |
|---------------|---------------------|
| show ip route | Displays all routes |

#### How Does OSPF Generate Default Routes?

How OSPF generates default routes (0.0.0.0) varies depending on the type of area the default route is being injected into—a standard area, stub area, totally stubby area, NSSA, or totally stubby NSSA.

By default, in standard areas, routers don't generate default routes. To have an OSPF router generate a default route, use the **default-information originate [always] [metric metric-value] [metric-type type-value] [route-map map-name]** command. By default, this command generates an E2 route with link-state ID 0.0.0.0 and network mask 0.0.0.0, which makes the router an ASBR.

There are two ways to inject a default route into a standard area. If the ASBR already has the default route, you can advertise 0.0.0.0 into the area. If the ASBR does not have the route, you can add the keyword **always** to the **default-information originate** command, which then advertises 0.0.0.0.

For stub and totally stubby areas, the ABR generates a summary LSA with the link-state ID 0.0.0.0. This is true even if the ABR does not have a default route. In this scenario, you do not need to use the **default-information originate** command.

The ABR for an NSSA generates the default route, but not by default. To force the ABR to generate the default route, use the **area area-id nssa default-information-originate** command. The ABR generates a type 7 LSA with the link-state ID 0.0.0.0. If you want to import routes only into the standard areas, not into the NSSA area, you can use the **no-redistribution** option on the NSSA ABR.

The ABR for a totally stubby NSSA automatically generates a default route.

#### Configuring and Verifying OSPF Authentication

As introduced in Chapter 2, you can prevent your router from receiving fraudulent route updates by configuring neighbor router authentication. OSPF neighbor authentication (also called neighbor router authentication or route authentication) can be configured such that routers can participate in routing based on predefined passwords.

Recall that when neighbor authentication has been configured on a router, the router authenticates the source of each routing update packet that it receives. This is accomplished by the exchange of an authenticating key (sometimes referred to as a *password*) that is known to both the sending and the receiving router.

By default, OSPF uses null authentication, which means that routing exchanges over a network are not authenticated. OSPF supports two other authentication methods: simple password authentication (also called plain-text authentication), and MD5 authentication.

##### Planning for OSPF Authentication

Before configuring OSPF authentication, the network administrator must examine the existing OSPF configuration and define the authentication requirements. The OSPF authentication requirements include the authentication type—none, simple password, or MD5—and the key (the password).

##### Configuring, Verifying, and Troubleshooting OSPF Simple Password Authentication

This section describes how to configure, verify, and troubleshoot OSPF simple password authentication. Configuring simple password authentication on virtual links is also examined.

##### Configuring OSPF Simple Password Authentication

To configure OSPF simple password authentication, complete the following steps:

- Step 1. Assign a password (key) to be used when using OSPF simple password authentication with neighboring routers, using the **ip ospf authentication-key password** interface configuration command. The *password* parameter is any continuous string of characters that can be entered from the keyboard up to 8 bytes in length

##### Note

In Cisco IOS Release 12.4, the router will give a warning message if you try to configure a

password longer than eight characters; only the first eight characters will be used. Some earlier Cisco IOS releases did not provide this warning.

The password created by this command is used as a “key” that is inserted directly into the OSPF header when the Cisco IOS Software originates routing protocol packets. A separate password can be assigned to each network on a per-interface basis. All neighboring routers on the same network must have the same password to be able to exchange OSPF information.

#### Note

If the **service password-encryption** command is not used when configuring OSPF authentication, the password will be stored as plain text in the router configuration. If you configure the **service password-encryption** command, the password will be stored and displayed in an encrypted form, and *when it is displayed*, there will be an *encryption type of 7 specified before the encrypted password*.

- Step 2. Specify the authentication type using the `ip ospf authentication [message-digest | null]` interface configuration command. Table 3-27 describes the parameters of the `ip ospf authentication` command.

Table 3-27. *ip ospf authentication* Command Parameters

| Parameter      | Description                                                                                                                  |
|----------------|------------------------------------------------------------------------------------------------------------------------------|
| message-digest | (Optional) Specifies that MD5 authentication will be used.                                                                   |
| null           | (Optional) No authentication is used. Useful for overriding simple password or MD5 authentication if configured for an area. |

For simple password authentication, use the **ip ospf authentication** command without any parameters. Before using this command, configure a password for the interface using the **ip ospf authentication-key** command.

The `ip ospf authentication` command was introduced in Cisco IOS Software Release 12.0. For backward compatibility, authentication type for an area is still supported. If the authentication type is not specified for an interface, the authentication type for the area will be used (the area default is null authentication). To enable authentication for an OSPF area, use the `area area-id authentication [message-digest]` router configuration command. Table 3-28 describes the parameters of the area authentication command.

Table 3-28. *area authentication* Command Parameters

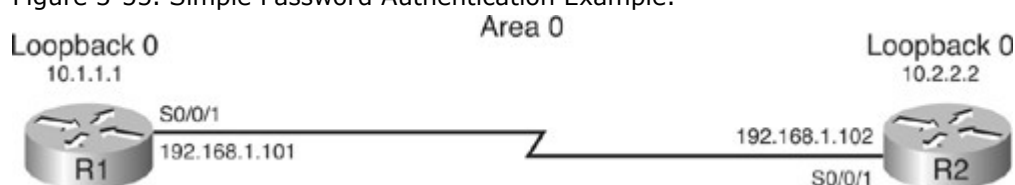
| Parameter      | Description                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| area-id        | Identifier of the area for which authentication is to be enabled. The identifier can be specified as either a decimal value or an IP address. |
| message-digest | (Optional) Enables MD5 authentication for the area specified by the <i>area-id</i> argument.                                                  |

For simple password authentication, use the **area authentication** command with no parameters.

#### Simple Password Authentication Example

Figure 3-55 shows the network used to illustrate the configuration, verification, and troubleshooting of simple password authentication. The configuration of the R1 and R2 routers are shown in Example 3-43.

Figure 3-55. Simple Password Authentication Example.



Example 3-43. Configuration of Routers R1 and R2 in Figure 3-55

Code View: Scroll / Show All

```

Router R1:
<output omitted>
interface Loopback0
 ip address 10.1.1.1 255.255.255.0

<output omitted>
interface Serial0/0/1
 ip address 192.168.1.101 255.255.255.224
 ip ospf authentication
 ip ospf authentication-key plainpas

<output omitted>
router ospf 10
 log-adjacency-changes
 network 10.1.1.1 0.0.0.0 area 0
 network 192.168.1.0 0.0.0.255 area 0

Router R2:
<output omitted>
interface Loopback0
 ip address 10.2.2.2 255.255.255.0

<output omitted>
interface Serial0/0/1
 ip address 192.168.1.102 255.255.255.224
 ip ospf authentication
 ip ospf authentication-key plainpas

<output omitted>
router ospf 10
 log-adjacency-changes
 network 10.2.2.2 0.0.0.0 area 0
 network 192.168.1.0 0.0.0.255 area 0

```

Notice that the connecting interfaces on both R1 and R2 are configured for the same type of authentication with the same authentication key. Simple password authentication is configured on interface Serial 0/0/1 on both routers, with the **ip ospf authentication** command. The interfaces are configured with an authentication key of *plainpas*.

#### Verifying Simple Password Authentication

Example 3-44 shows the output of the `show ip ospf interface`, `show ip ospf neighbor`, and `show ip route` commands on the R1 router in Figure 3-55. From the `show ip ospf interface` and `show ip ospf neighbor` output, you see that R1 has one adjacent neighbor, and simple password authentication is enabled. The results of a ping to the R2 loopback interface address are also displayed to illustrate that the link is working.

Example 3-44. Verifying Simple Password Authentication on R1 in Figure 3-55

```

Code View: Scroll / Show All
R1#show ip ospf interface
Serial0/0/1 is up, line protocol is up
 Internet Address 192.168.1.101/27, Area 0

```

```
Process ID 1, Router ID 10.1.1.1, Network Type POINT_TO_POINT, Cost: 64
Transmit Delay is 1 sec, State POINT_TO_POINT
```

```
<output omitted>
```

```
Neighbor Count is 1, Adjacent neighbor count is 1
```

```
Adjacent with neighbor 10.2.2.2
```

```
Suppress hello for 0 neighbor(s)
```

```
Simple password authentication enabled
```

```
<output omitted>
```

```
R1#show ip ospf neighbor
```

| Neighbor ID | Pri | State   | Dead Time | Address       | Interface   |
|-------------|-----|---------|-----------|---------------|-------------|
| 10.2.2.2    | 0   | FULL/ - | 00:00:32  | 192.168.1.102 | Serial0/0/1 |

```
R1#show ip route
```

```
<output omitted>
```

```
Gateway of last resort is not set
```

```
10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
```

```
O 10.2.2.2/32 [110/782] via 192.168.1.102, 00:01:17, Serial0/0/1
```

```
C 10.1.1.0/24 is directly connected, Loopback0
```

```
192.168.1.0/27 is subnetted, 1 subnets
```

```
C 192.168.1.96 is directly connected, Serial0/0/1
```

```
R1#ping 10.2.2.2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 10.2.2.2, timeout is 2 seconds:
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```

Notice in the **show ip ospf neighbor** command output that the neighbor state is FULL, indicating that the two routers have successfully formed an OSPF adjacency. The routing table verifies that the 10.2.2.2 address has been learned via OSPF over the serial connection.

Troubleshooting Simple Password Authentication

If the authentication configuration between routers is correct, an OSPF neighbor relationship is established, routing updates are exchanged, and OSPF routes enter the IP routing table.

OSPF authentication issues between routers may arise from the following problems:

- If authentication is not configured on both routers
- If different authentication types are configured on the routers
- If different passwords are configured on the routers

The **debug ip ospf adj** command is used to display OSPF adjacency-related events and is useful when troubleshooting authentication.

Successful Simple Password Authentication Example

The output of the debug ip ospf adj command in Example 3-45 illustrates successful communication on the R1 router in Figure 3-55 after the serial 0/0/1 interface, on which simple password authentication has been configured, comes up.

Note

Although this **debug ip ospf adj** output does not indicate anything about the authentication, it does show that the two routers successfully form a FULL adjacency. As the output in the next section illustrates, this command output does display authentication failures if there are any. During testing we were unable to find any **debug** command output that displayed information about successful OSPF simple password authentication.

#### Example 3-45. Successful: Simple Password Authentication on R1 in Figure 3-55

Code View: Scroll / Show All

```
*Apr 20 18:41:51.242: OSPF: Interface Serial0/0/1 going Up
*Apr 20 18:41:51.742: OSPF: Build router LSA for area 0, router ID 10.1.1.1, seq
0x80000013
*Apr 20 18:41:52.242: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1,
changed state to up
*Apr 20 18:42:01.250: OSPF: 2 Way Communication to 10.2.2.2 on Serial0/0/1, state
2WAY
*Apr 20 18:42:01.250: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x9B6 opt
0x52 flag 0x7 len 32
*Apr 20 18:42:01.262: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x23ED
opt0x52 flag 0x7 len 32 mtu 1500 state EXSTART
*Apr 20 18:42:01.262: OSPF: NBR Negotiation Done. We are the SLAVE
*Apr 20 18:42:01.262: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x23ED opt
0x52 flag 0x2 len 72
*Apr 20 18:42:01.294: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x23EE
opt0x52 flag 0x3 len 72 mtu 1500 state EXCHANGE
*Apr 20 18:42:01.294: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x23EE opt
0x52 flag 0x0 len 32
*Apr 20 18:42:01.294: OSPF: Database request to 10.2.2.2
*Apr 20 18:42:01.294: OSPF: sent LS REQ packet to 192.168.1.102, length 12
*Apr 20 18:42:01.314: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x23EF
opt0x52 flag 0x1 len 32 mtu 1500 state EXCHANGE
*Apr 20 18:42:01.314: OSPF: Exchange Done with 10.2.2.2 on Serial0/0/1
*Apr 20 18:42:01.314: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x23EF opt
0x52 flag 0x0 len 32
*Apr 20 18:42:01.326: OSPF: Synchronized with 10.2.2.2 on Serial0/0/1, state FULL
*Apr 20 18:42:01.330: %OSPF-5-ADJCHG: Process 10, Nbr 10.2.2.2 on Serial0/0/1 from
LOADING to
FULL, Loading Done
*Apr 20 18:42:01.830: OSPF: Build router LSA for area 0, router ID 10.1.1.1, seq
0x80000014
```

The output of the show ip ospf neighbor command shown in Example 3-46 illustrates that R1 has successfully formed an adjacency with R2.

#### Example 3-46. R1 and R2 in Figure 3-55 Have Formed an Adjacency

Code View: Scroll / Show All

R1#**show ip ospf neighbor**

| Neighbor ID | Pri | State   | Dead Time | Address       | Interface   |
|-------------|-----|---------|-----------|---------------|-------------|
| 10.2.2.2    | 0   | FULL/ - | 00:00:34  | 192.168.1.102 | Serial0/0/1 |

#### Troubleshooting Simple Password Authentication Problems Example

Using the network in Figure 3-55, if simple password authentication is configured on the R1 serial 0/0/1 interface but no authentication is configured on the R2 serial 0/0/1 interface, the routers will not be able to form an adjacency over that link. The output of the debug ip ospf adj command shown in Example 3-

47 illustrates that the routers report a mismatch in authentication type. No OSPF packets will be sent between the neighbors.

Example 3-47. Simple Password Authentication on R1 and no Authentication on R2 in Figure 3-55

Code View: Scroll / Show All

R1#

```
*Apr 17 18:51:31.242: OSPF: Rcv pkt from 192.168.1.102, Serial0/0/1 : Mismatch
Authentication type. Input packet specified type 0, we use type 1
```

R2#

```
*Apr 17 18:50:43.046: OSPF: Rcv pkt from 192.168.1.101, Serial0/0/1 : Mismatch
Authentication type. Input packet specified type 1, we use type 0
```

Note

The different types of OSPF authentication have the following type codes:

**Null**—Type 0

**Simple password**—Type 1

**MD5**—Type 2

If simple password authentication is configured on the R1 Serial 0/0/1 interface and on the R2 Serial 0/0/1 interface, but with different passwords, the routers will not be able to form an adjacency over that link. The outputs of the debug ip ospf adj command shown in Example 3-48 illustrate that the routers report a mismatch in authentication key. No OSPF packets will be sent between the neighbors.

Example 3-48. Simple Password Authentication on R1 and R2 in Figure 3-55, but with Different Passwords

Code View: Scroll / Show All

R1#

```
*Apr 17 18:54:01.238: OSPF: Rcv pkt from 192.168.1.102, Serial0/0/1 : Mismatch
Authentication Key - Clear Text
```

R2#

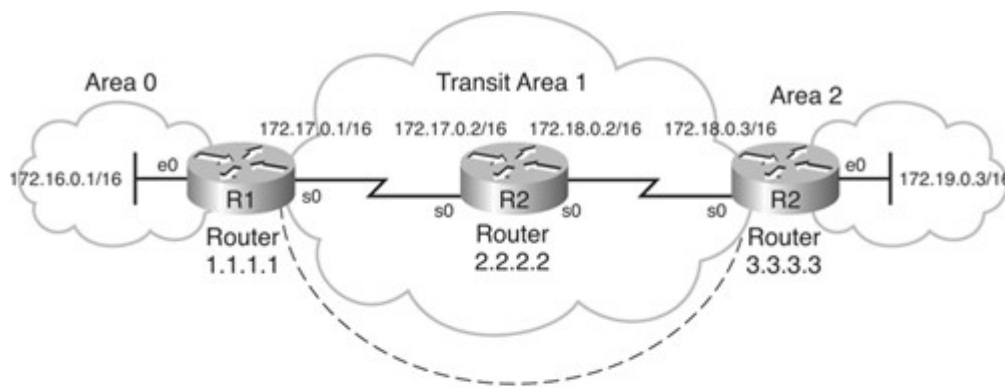
```
*Apr 17 18:53:13.050: OSPF: Rcv pkt from 192.168.1.101, Serial0/0/1 : Mismatch
Authentication Key - Clear Text
```

Configuring OSPF Simple Password Authentication for Virtual Links

Figure 3-56 illustrates a network with a virtual link. The configuration of simple password authentication for the virtual link, on routers R1 and R3, is also shown in the figure.

Figure 3-56. OSPF Simple Password Authentication over a Virtual Link.

[View full size image]



R1#

```
router ospf 10
 network 172.16.0.0 0.0.255.255 area 0
 network 172.17.0.0 0.0.255.255 area 1
 area 0 authentication
 !
 area 1 virtual-link 3.3.3.3 authentication-key cisco
```

R3#

```
router ospf 10
 network 172.19.0.0 0.0.255.255 area 2
 network 172.18.0.0 0.0.255.255 area 1
 area 0 authentication
 !
 area 1 virtual-link 1.1.1.1 authentication-key cisco
```

On router R1, simple password authentication is configured for the whole area 0, with the **area 0 authentication** command. The virtual link, connecting area 2 to area 0, is created via transit area 1 with plain text authentication and the authentication key *cisco*, with the **area 1 virtual-link 3.3.3.3 authentication-key cisco** command.

The configuration of router R3 is similar to router R1.

Configuring, Verifying, and Troubleshooting MD5 Authentication

This section describes how to configure, verify, and troubleshoot OSPF MD5 authentication.

Note

OSPF MD5 authentication includes a nondecreasing sequence number in each OSPF packet to protect against replay attacks.

Configuring OSPF MD5 Authentication

With OSPF MD5 authentication, a key and key ID are configured on each router. To configure OSPF MD5 authentication, complete the following steps:

Step 1. Assign a key ID and key to be used with neighboring routers that are using the OSPF MD5 authentication, using the `ip ospf message-digest-key key-id md5 key interface` configuration command. Table 3-29 describes the parameters in the `ip ospf message-digest-key` command.

Table 3-29. *ip ospf message-digest-key* Command Parameters

| Parameter | Description                              |
|-----------|------------------------------------------|
| key-id    | An identifier in the range from 1 to 255 |
| Key       | Alphanumeric password of up to 16 bytes  |

The key and the key ID specified in this command are used to generate a message digest (also called a *hash*) of each OSPF packet. The message digest is appended to the packet. A separate password can be assigned to each network on a per-interface basis.

Note



In Cisco IOS Release 12.4, the router will give a warning message if you try to configure a password longer than 16 characters, and only the first 16 characters will be used. Some earlier Cisco IOS releases did not provide this warning.

Usually, one key per interface is used to generate authentication information when sending packets and to authenticate incoming packets. All neighboring routers on the same network must have the same password to be able to exchange OSPF information. In other words, the same *key-id* on the neighbor router must have the same *key* value.

The *key-id* allows for uninterrupted transitions between keys, which is helpful for administrators who want to change the OSPF password without disrupting communication. If an interface is configured with a new key, the router will send multiple copies of the same packet, each authenticated by different keys. The router will stop sending duplicate packets when it detects that all of its neighbors have adopted the new key.

The process of changing keys is as follows. Suppose the current configuration is as follows:

```
interface FastEthernet 0/0
ip ospf message-digest-key 100 md5 OLD
```

The following configuration is then added:

```
interface FastEthernet 0/0
ip ospf message-digest-key 101 md5 NEW
```

The router assumes its neighbors do not have the new key yet, so it begins a rollover process. It sends multiple copies of the same packet, each authenticated by different keys. In this example, the router sends out two copies of the same packet: the first one authenticated by key 100 and the second one authenticated by key 101.

Rollover allows neighboring routers to continue communication while the network administrator is updating them with the new key. Rollover stops once the local system finds that all its neighbors know the new key. The system detects that a neighbor has the new key when it receives packets from the neighbor authenticated by the new key.

After all neighbors have been updated with the new key, the old key should be removed. In this example, you would enter the following:

```
interface FastEthernet 0/0
no ip ospf message-digest-key 100
```

From then on, only key 101 is used for authentication on interface Fast Ethernet 0/0.

Cisco recommends that you not keep more than one key per interface. Every time you add a new key, you should remove the old key to prevent the local router from continuing to communicate with a hostile system that knows the old key.

Note

If the **service password-encryption** command is not used when implementing OSPF authentication, the key will be stored as plain text in the router configuration. If you configure the **service password-encryption** command, the key will be stored and displayed in an encrypted form; when it is displayed, there will be an *encryption-type* of 7 specified before the encrypted key.

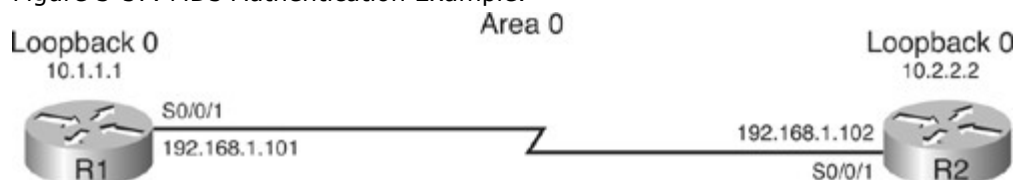
- Step 2. Specify the authentication type using the `ip ospf authentication [message-digest | null] interface` configuration command. The parameters for this command are as described in the "Configuring OSPF Simple Password Authentication" section, earlier in this chapter. For MD5 authentication, use the `ip ospf authentication` command with the `message-digest` parameter. Before using this command, configure the message digest key for the interface with the `ip ospf message-digest-key` command.

Recall that the `ip ospf authentication` command was introduced in Cisco IOS Software Release 12.0. As for simple password authentication, the MD5 authentication type for an area is still supported using the `area area-id authentication message-digest router` configuration command, for backward compatibility.

## MD5 Authentication Example

Figure 3-57 shows the network used to illustrate the configuration, verification, and troubleshooting of MD5 authentication. The configuration of the R1 and R2 routers are shown in Example 3-49.

Figure 3-57. MD5 Authentication Example.



Example 3-49. Configuration of Routers R1 and R2 in Figure 3-57

Code View: Scroll / Show All

Router R1:

<output omitted>

interface Loopback0

ip address 10.1.1.1 255.255.255.0

<output omitted>

interface Serial0/0/1

ip address 192.168.1.101 255.255.255.224

ip ospf authentication message-digest

ip ospf message-digest-key 1 md5 secretpass

<output omitted>

router ospf 10

log-adjacency-changes

network 10.1.1.1 0.0.0.0 area 0

network 192.168.1.0 0.0.0.255 area 0

Router R2:

<output omitted>

interface Loopback0

ip address 10.2.2.2 255.255.255.0

<output omitted>

interface Serial0/0/1

ip address 192.168.1.102 255.255.255.224

ip ospf authentication message-digest

ip ospf message-digest-key 1 md5 secretpass

<output omitted>

router ospf 10

log-adjacency-changes

network 10.2.2.2 0.0.0.0 area 0

Notice that the connecting interfaces on both R1 and R2 are configured for the same type of authentication with the same authentication key and key ID. MD5 authentication is configured on interface Serial 0/0/1 on both routers with the **ip ospf authentication message-digest** command. The interfaces on both routers are configured with an authentication key number 1 set to *secretpass*.

## Verifying MD5 Authentication

Example 3-50 shows the output of the show ip ospf interface, show ip ospf neighbor and show ip route commands on the R1 router in Figure 3-57. The results of a ping to the R2 loopback interface address is also displayed to illustrate that the link is working.

Example 3-50. Verifying MD5 Authentication on R1 in Figure 3-57

```
Code View: Scroll / Show All
R1#show ip ospf interface
Serial0/0/1 is up, line protocol is up
 Internet Address 192.168.1.101/27, Area 0
 Process ID 10, Router ID 10.1.1.1, Network Type POINT_TO_POINT, Cost: 64
 Transmit Delay is 1 sec, State POINT_TO_POINT
<output omitted>
 Neighbor Count is 1, Adjacent neighbor count is 1
 Adjacent with neighbor 10.2.2.2
 Suppress hello for 0 neighbor(s)
 Message digest authentication enabled
 Youngest key id is 1
<output omitted>

R1#show ip ospf neighbor
Neighbor ID Pri State Dead Time Address Interface
10.2.2.2 0 FULL/ - 00:00:31 192.168.1.102 Serial0/0/1

R1#show ip route
<output omitted>
Gateway of last resort is not set
 10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
O 10.2.2.2/32 [110/782] via 192.168.1.102, 00:00:37, Serial0/0/1
C 10.1.1.0/24 is directly connected, Loopback0
 192.168.1.0/27 is subnetted, 1 subnets
C 192.168.1.96 is directly connected, Serial0/0/1

R1#ping 10.2.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.2.2.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
```

Notice that the **show ip ospf interface** output shows that router R1 has one adjacent neighbor and that message digest authentication is enabled. The **show ip ospf neighbor** command output shows that the neighbor state is FULL, indicating that the two routers have successfully formed an OSPF adjacency. The routing table verifies that the 10.2.2.2 address has been learned via OSPF over the serial connection.

## Troubleshooting MD5 Authentication

As for simple password authentication, the **debug ip ospf adj** command is used to display OSPF adjacency-related events and is very useful when troubleshooting MD5 authentication.

## Successful MD5 Authentication Example

The output of the debug ip ospf adj command in Example 3-51 illustrates successful MD5 authentication on the R1 router in Figure 3-57 after the Serial 0/0/1 interface, on which authentication has been configured, comes up.

Example 3-51. Successful MD5 Authentication on R1 in Figure 3-57

Code View: Scroll / Show All

R1#**debug ip ospf adj**

OSPF adjacency events debugging is on

```
*Apr 20 17:13:56.530: %LINK-3-UPDOWN: Interface Serial0/0/1, changed state to up
*Apr 20 17:13:56.530: OSPF: Interface Serial0/0/1 going Up
*Apr 20 17:13:56.530: OSPF: Send with youngest Key 1
*Apr 20 17:13:57.030: OSPF: Build router LSA for area 0, router ID 10.1.1.1, seq
0x80000009
*Apr 20 17:13:57.530: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1,
changed state to up
*Apr 20 17:14:06.530: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.546: OSPF: 2 Way Communication to 10.2.2.2 on Serial0/0/1, state
2WAY
*Apr 20 17:14:06.546: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0xB37 opt
0x52 flag 0x7 len 32
*Apr 20 17:14:06.546: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.562: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x32F opt 0
x52 flag 0x7 len 32 mtu 1500 state EXSTART
*Apr 20 17:14:06.562: OSPF: NBR Negotiation Done. We are the SLAVE
*Apr 20 17:14:06.562: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x32F opt
0x52 flag 0x2 len 72
*Apr 20 17:14:06.562: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.602: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x330 opt
0x52 flag 0x3 len 72 mtu 1500 state EXCHANGE
*Apr 20 17:14:06.602: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x330 opt
0x52 flag 0x0 len 32
*Apr 20 17:14:06.602: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.602: OSPF: Database request to 10.2.2.2
*Apr 20 17:14:06.602: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.602: OSPF: sent LS REQ packet to 192.168.1.102, length 12
*Apr 20 17:14:06.614: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.634: OSPF: Rcv DBD from 10.2.2.2 on Serial0/0/1 seq 0x331 opt
0x52 flag 0x1 len 32 mtu 1500 state EXCHANGE
*Apr 20 17:14:06.634: OSPF: Exchange Done with 10.2.2.2 on Serial0/0/1
*Apr 20 17:14:06.634: OSPF: Send DBD to 10.2.2.2 on Serial0/0/1 seq 0x331 opt
0x52 flag 0x0 len 32
*Apr 20 17:14:06.634: OSPF: Send with youngest Key 1
*Apr 20 17:14:06.650: OSPF: Synchronized with 10.2.2.2 on Serial0/0/1, state FULL
*Apr 20 17:14:06.650: %OSPF-5-ADJCHG: Process 10, Nbr 10.2.2.2 on Serial0/0/1 from
LOADING to FULL, Loading Done
*Apr 20 17:14:07.150: OSPF: Send with youngest Key 1
*Apr 20 17:14:07.150: OSPF: Build router LSA for area 0, router ID 10.1.1.1, seq
0x8000000A
*Apr 20 17:14:09.150: OSPF: Send with youngest Key 1
```

The output of the show ip ospf neighbor command shown Example 3-52 illustrates that R1 has successfully formed an adjacency with R2.

#### Example 3-52. R1 and R2 in Figure 3-57 Have Formed an Adjacency

Code View: Scroll / Show All

R1#**show ip ospf neighbor**

| Neighbor ID | Pri | State   | Dead Time | Address       | Interface   |
|-------------|-----|---------|-----------|---------------|-------------|
| 10.2.2.2    | 0   | FULL/ - | 00:00:34  | 192.168.1.102 | Serial0/0/1 |

#### Troubleshooting MD5 Authentication Problems Example

Using the network in Figure 3-57 if MD5 authentication is configured on the R1 Serial 0/0/1 interface and on the R2 Serial 0/0/1 interface, but R1 has key 1 and R2 has key 2, the routers will not be able to form an adjacency over that link, even though both have the same passwords configured. The outputs of the debug ip ospf adj command shown in Example 3-53 illustrate that the routers report a mismatch in authentication key. No OSPF packets will be sent between the neighbors.

#### Example 3-53. MD5 Authentication on R1 and R2 in Figure 3-57 but with Different Key IDs

Code View: Scroll / Show All

R1#

\*Apr 20 17:56:16.530: OSPF: Send with youngest Key 1

\*Apr 20 17:56:26.502: OSPF: Rcv pkt from 192.168.1.102, Serial0/0/1 : Mismatch

Authentication Key - No message digest key 2 on interface

\*Apr 20 17:56:26.530: OSPF: Send with youngest Key 1

R2#

\*Apr 20 17:55:28.226: OSPF: Send with youngest Key 2

\*Apr 20 17:55:28.286: OSPF: Rcv pkt from 192.168.1.101, Serial0/0/1 : Mismatch

Authentication Key - No message digest key 1 on interface

\*Apr 20 17:55:38.226: OSPF: Send with youngest Key 2

#### Summary

In this chapter you learned about the OSPF link-state routing protocol. The chapter focused on the following topics:

- Characteristics of link-state routing protocols such as OSPF, including the OSPF tables—the neighbor table (also called the adjacency database), the topology table (also called the topology database or the LSDB), and the routing table (also called the forwarding database).
- OSPF's two-tier hierarchical area structure, with a backbone area 0 and regular areas.
- The different types of OSPF routers: internal routers, backbone routers, ABRs, and ASBRs.
- How OSPF routers use the Hello protocol to build adjacencies. After two routers establish neighbor adjacency using hello packets, they synchronize their LSDBs by exchanging LSAs and confirming the receipt of LSAs from the adjacent router. Routers on point-to-point links form a full adjacency with each other, whereas routers on LAN links only form a full adjacency with the DR and BDR.
- The OSPF metric calculation on Cisco routers, which is based on the link bandwidth by default. If interfaces that are faster than 100 Mbps are being used, you should use the **auto-cost reference-bandwidth ref-bw** router configuration command on all routers. To override the default cost, manually define the cost using the **ip ospf cost interface-cost** interface configuration command.
- The five types of OSPF packets—hello, DBD, LSR, LSU, and LSAck. Hello packets are used to discover neighbor and build adjacencies. DBDs are used to synchronize the LSDBs. LSRs are used to request specific link-state records, and LSUs are used to send the requested records. LSAck is used to acknowledge the other packet types. OSPF packets are sent in IP packets with protocol 89.

- The neighbor states that OSPF may pass through: down, (possibly attempt), init, two-way, exstart, exchange, loading, and full.
- The five fields in the hello packet must match on neighboring routers: Hello Interval, Dead Interval, Area ID, Authentication Password, And Stub Area Flag.
- Planning OSPF implementations, including the IP addressing, network topology, and OSPF areas. The list of tasks for each router in the network include enabling the OSPF routing protocol (with a process number) directly on an interface or by configuring the proper **network** commands, assigning the correct area ID to the interface, and optionally configuring the metric to appropriate interfaces.
- Basic OSPF configuration commands:
  - **router ospf** *process-id* global configuration command
  - **network** *ip-address wildcard-mask area area-id* interface configuration command, or the **ip ospf** *process-id area area-id [secondaries none]* interface configuration command
  - **bandwidth** *kilobits* interface configuration command
  - **router-id** *ip-address* router configuration command
- Commands for verifying OSPF operation:
  - show ip ospf
  - show ip route
  - show ip ospf interface
  - show ip ospf neighbor
  - show ip route ospf
  - show ip protocols
  - debug ip ospf events
  - debug ip ospf adj
  - debug ip ospf packet
- How the OSPF router ID is selected: with the **router-id** *ip-address* router configuration command, the highest IP address on any active loopback interface, or the highest IP address of any active physical interface when OSPF starts.
- The three types of networks defined by OSPF: point-to-point, broadcast, and NBMA.
- How a DR and BDR are selected: The router with the highest priority is the DR, the router with the second highest priority is the BDR. The router ID breaks a tie. A router with priority 0 does not become either the DR or the BDR. The priority is set with the **ip ospf priority** *number* interface configuration command
- The five modes of OSPF operation available for NBMA networks: nonbroadcast and point-to-multipoint RFC modes, and broadcast, point-to-multipoint nonbroadcast, and point-to-point Cisco modes. (Refer to Table 3-11 for a summary of these modes).
- The 11 different OSPF LSA types. The first five are the most commonly used: type 1 (router, generated by every router), type 2 (network, generated by DR), type 3 (summary, describes area routes, generated by ABR), type 4 (summary, describes route to ASBR, generated by ABR), and type 5 (external, describes routes to external destinations, generated by ASBR).
- The three kinds of OSPF routes: intra-area (O), interarea (O IA), and external (either O E1 or O E2).
- Configuring OSPF LSDB overload protection using the **max-lsa** *maximum-number* [*threshold-percentage*] [**warning-only**] [**ignore-time** *minutes*] [**ignore-count** *count-number*] [**reset-time** *minutes*] router configuration command.
- Using the **passive-interface** *type number* [**default**] router configuration command to prevent a routing protocol's routing updates from being sent through the specified router interface. With OSPF, the specified interface appears as a stub network, and OSPF routing information is neither sent nor received through the specified interface.
- How default routes can be used in OSPF to prevent the need for a specific route to all destination networks. The benefit is a much smaller routing table and LSDB, with complete reachability. Propagate an OSPF default route using the **default-information originate** [**always**] [**metric** *metric-value*] [**metric-type** *type-value*] [**route-map** *map-name*] router configuration command.
- Configuring route summarization to improve CPU utilization, reduce LSA flooding, and reduce LSDB and routing table sizes. OSPF does not automatically summarize. OSPF summarization can be configured on an ABR using the **area** *area-id* **range** *address mask* [**advertise** | **not-advertise**]

[**cost** *cost*] router configuration command, and on an ASBR using the **summary-address** *ip-address mask* [**not-advertise**] [**tag** *tag*] router configuration command.

- The OSPF virtual link feature, used to *temporarily* mend backbone failures or connect a disconnected area to the backbone. Virtual links are configured with the **area** *area-id* **virtual-link** *router-id* [**authentication** [**message-digest** | **null**]] [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [**dead-interval** *seconds*] [[**authentication-key** *key*] | [**message-digest-key** *key-id md5 key*]] router configuration command, and verified with the **show ip ospf virtual-links** command.
- The several area types defined in OSPF: standard areas, backbone areas, stub areas, totally stubby areas, NSSAs, and totally stubby NSSAs. (Refer to Table 3-23 for a summary of the area types.) The **area** *area-id* **stub** router configuration command is used to configure stub areas. The **no-summary** keyword is added to this command on the ABR to make the area totally stubby. The **area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate**] [**metric** *metric-value*] [**metric-type** *type-value*] [**no-summary**] router configuration command is used to configure an NSSA, instead of the **area** *area-id* **stub** command. The **no-summary** keyword is only used on the ABR to make the area a totally stubby NSSA area.
- The types of OSPF authentication: null, simple password authentication (also called plain-text authentication), and MD5 authentication. Authentication troubleshooting is done with the **debug ip ospf adj** command.
- The commands to configure OSPF simple password authentication:
  - **ip ospf authentication-key** *password* interface configuration command
  - **ip ospf authentication** interface configuration command or the **area** *area-id* **authentication** router configuration command
- The commands to configure OSPF MD5 authentication:
  - **ip ospf message-digest-key** *key-id md5 key* interface configuration command
  - **ip ospf authentication message-digest** interface configuration command or the **area** *area-id* **authentication message-digest** router configuration command

## References

For additional information, see these resources:

- "OSPF: Frequently Asked Questions"  
at [http://www.cisco.com/en/US/tech/tk365/technologies\\_q\\_and\\_a\\_item09186a0080094704.shtml](http://www.cisco.com/en/US/tech/tk365/technologies_q_and_a_item09186a0080094704.shtml)
- "OSPF Not-So-Stubby Area (NSSA)"  
at [www.cisco.com/en/US/tech/tk365/technologies\\_tech\\_note09186a0080094a88.shtml](http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094a88.shtml)
- "OSPF Design Guide"  
at [www.cisco.com/en/US/tech/tk365/technologies\\_white\\_paper09186a0080094e9e.shtml](http://www.cisco.com/en/US/tech/tk365/technologies_white_paper09186a0080094e9e.shtml)
- "Designing Large-Scale IP Internetworks"  
at [www.cisco.com/en/US/docs/internetworking/design/guide/nd2003.html](http://www.cisco.com/en/US/docs/internetworking/design/guide/nd2003.html)
- "Cisco IOS Software Releases 12.4 Mainline" support page  
at [http://www.cisco.com/en/US/products/ps6350/tsd\\_products\\_support\\_series\\_home.html](http://www.cisco.com/en/US/products/ps6350/tsd_products_support_series_home.html)
- Perlman, Radia. *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols* (2nd Edition), Addison-Wesley Professional (1999). An excellent reference for understanding routing protocol sequence numbering and aging

## Review Questions

Answer the following questions, and then see Appendix A, "Answers to Review Questions," for the answers.

1. Which of the following is not a characteristic of link-state routing protocols?
  - a. They respond quickly to network changes.
  - b. They broadcast every 30 minutes.
  - c. They send triggered updates when a network change occurs.
  - d. They may send periodic updates, known as link-state refresh, at long time intervals, such as every 30 minutes.
2. For all the routers in the network to make consistent routing decisions, each link-state router must keep a record of all the following items except which one?
  - a. Its immediate neighbor routers



- b. All the other routers in the network, or in its area of the network, and their attached networks
  - c. The best paths to each destination
  - d. The version of the routing protocol used
- 3. Link-state routing protocols use a two-layer area hierarchy composed of which two areas?
  - a. Backbone area
  - b. Transmit area
  - c. Regular area
  - d. Linking area
- 4. Which of the following is not a characteristic of an OSPF area?
  - a. It may minimize routing table entries.
  - b. It requires a flat network design.
  - c. It may localize the impact of a topology change within an area.
  - d. It may stop detailed LSA flooding at the area boundary.
- 5. What does an ABR do?
- 6. When a router receives an LSA (within an LSU), it does not do which of the following?
  - a. If the LSA entry does not already exist, the router adds the entry to its LSDB, sends back an LSack, floods the information to other routers, runs SPF, and updates its routing table.
  - b. If the entry already exists and the received LSA has the same sequence number, the router overwrites the information in the LSDB with the new LSA entry.
  - c. If the entry already exists but the LSA includes newer information (it has a higher sequence number), the router adds the entry to its LSDB, sends back an LSack, floods the information to other routers, runs SPF, and updates its routing table.
  - d. If the entry already exists but the LSA includes older information, it sends an LSU to the sender with its newer information.
- 7. What is an OSPF type 2 packet?
  - a. Database description (DBD), which checks for database synchronization between routers
  - b. Link-state request (LSR), which requests specific link-state records from router to router
  - c. Link-state update (LSU), which sends specifically requested link-state records
  - d. Link-state acknowledgment (LSack), which acknowledges the other packet types
- 8. Which of the following is true of hellos and dead intervals?
  - a. They do not need to be the same on neighboring routers, because the lowest common denominator is adopted.
  - b. They do not need to be the same on neighboring routers, because the highest common denominator is adopted.
  - c. They do not need to be the same on neighboring routers, because it is a negotiated interval between neighboring routers.
  - d. They need to be the same on neighboring routers.
- 9. Which IP address is used to send an updated LSA entry to OSPF DRs and BDRs?
  - a. Unicast 224.0.0.5
  - b. Unicast 224.0.0.6
  - c. Multicast 224.0.0.5
  - d. Multicast 224.0.0.6
- 10. To ensure an accurate database, how often does OSPF flood (refresh) each LSA record?
  - a. Every 60 minutes.
  - b. Every 30 minutes.
  - c. Every 60 seconds.
  - d. Every 30 seconds.
  - e. Flooding each LSA record would defeat the purpose of a link-state routing protocol, which strives to reduce the amount of routing traffic it generates.
- 11. Which of the following is *not* information necessary to implement OSPF routing?
  - a. The IP addresses to be configured on router interfaces
  - b. The process number to be used for OSPF.

- c. The area in which the router is to be configured
- 12. What command is used to display the router ID, timers, and statistics?
  - a. show ip ospf
  - b. show ip ospf neighbors
  - c. show ip ospf stats
  - d. show ip ospf neighborship
- 13. Which of the following is not a way in which the OSPF router ID (a unique IP address) can be assigned?
  - a. The highest IP address of any physical interface
  - b. The lowest IP address of any physical interface
  - c. The IP address of a loopback interface
  - d. The **router-id** command
- 14. Two routers are connected by a point-to-point network. How does each the router dynamically detect its neighboring router?
- 15. Which of the following best describes where an adjacency relationship exists?
  - a. Between routers located on the same physical network
  - b. Between routers in different OSPF areas
  - c. Between a router and its DR and BDR on different networks
  - d. Between a backbone DR and a transit BDR
- 16. Which of the following is not true regarding the OSPF DR/BDR election?
  - a. The router with the highest priority value is the DR.
  - b. The router with the second-highest priority value is the BDR.
  - c. If all routers have the default priority, the router with the lowest router ID becomes the DR.
  - d. The router with a priority set to 0 cannot become the DR or BDR.
- 17. You are using OSPF over a Layer 3 MPLS VPN, connecting office A to office B. The router in office A is connected to the MPLS provider's PEA router, and the router in office B is connected to the MPLS provider's PEB router. Which routers establish an OSPF adjacency?
- 18. Which of the following is *not* true of OSPF point-to-multipoint mode?
  - a. It does not require a full-mesh network.
  - b. It does not require a static neighbor configuration.
  - c. It uses multiple IP subnets.
  - d. It duplicates LSA packets.
- 19. What is the default OSPF mode on a point-to-point Frame Relay subinterface?
  - a. Point-to-point mode
  - b. Multipoint mode
  - c. Nonbroadcast mode
  - d. Broadcast mode
- 20. What is the default OSPF mode on a Frame Relay multipoint subinterface?
  - a. Point-to-point mode
  - b. Multipoint mode
  - c. Nonbroadcast mode
  - d. Broadcast mode
- 21. What is the default OSPF mode on a main Frame Relay interface?
  - a. Point-to-point mode
  - b. Multipoint mode
  - c. Nonbroadcast mode
  - d. Broadcast mode
- 22. What are some issues that may arise if an OSPF area becomes too large?
- 23. Match the type of router with its description:

| Type of Router    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1—Internal router | A—A router that sits in the perimeter of the backbone area and that has at least one interface connected to area 0. It maintains OSPF routing information using the same procedures and algorithms as an internal router.                                                                                                                                                                                                                                                                                                                        |
| 2—Backbone router | B—A router that has interfaces attached to multiple areas, maintains separate LSDBs for each area to which it connects, and routes traffic destined for or arriving from other areas. This router is an exit point for the area, which means that routing information destined for another area can get there only via the local area's router of this type. This kind of router can be configured to summarize the routing information from the LSDBs of its attached areas. This router distributes the routing information into the backbone. |
| 3—ABR             | C—A router that has all its interfaces in the same area.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 4—ASBR            | D—A router that has at least one interface attached to a different routing domain, such as a non-OSPF network. This router can redistribute external routes into the OSPF domain and vice versa.                                                                                                                                                                                                                                                                                                                                                 |

24. How many different types of LSAs are there?
  - a. 5
  - b. 9
  - c. 10
  - d. 11
25. What kind of router generates LSA type 5 in a standard area?
  - a. DR
  - b. ABR
  - c. ASBR
  - d. ADR
26. Where does a type 1 LSA flood to?
  - a. To immediate peers
  - b. To all other routers in the area where it originated
  - c. To routers located in other areas
  - d. To all areas
27. How does a routing table reflect the link-state information of an intra-area route?
  - a. The route is marked with O.
  - b. The route is marked with I.
  - c. The route is marked with IO.
  - d. The route is marked with EA.
  - e. The route is marked with O IA.
28. Which type of external route is the default?
  - a. E1.
  - b. E2.
  - c. E5.
  - d. There is no default external route. OSPF adapts and chooses the most accurate one.
29. How is the cost of an E1 external route calculated?
  - a. The sum of the internal cost of each link the packet crosses
  - b. The sum of the external cost and the internal cost of each link the packet crosses
  - c. The external cost only
  - d. The sum of all area costs, even those that are not used

30. What does the OSPF **max-lsa** command do?
- a. Defines the maximum number of LSAs that the router can generate
  - b. Protects the router from an excessive number of received (non-self-generated) LSAs in its LSDB
  - c. Defines the maximum size of the LSAs that the router generates
  - d. Protects the router from excessively large received (non-self-generated) LSAs in its LSDB
31. What does the **passive-interface** command do when configured on an OSPF router?
32. Select the true statements about different area types.
- a. Routers in a standard area can have O IA routes (other than the default route) in their routing table.
  - b. Routers in a stub area can have O IA routes (other than the default route) in their routing table.
  - c. Routers in a totally stubby area can have O IA routes (other than the default route) in their routing table.
  - d. Routers in an NSSA area can have O IA routes (other than the default route) in their routing table.
  - e. Routers in a totally stubby NSSA area can have O IA routes (other than the default route) in their routing table.
33. What command makes an OSPF router generate a default route?
- a. ospf default-initiate
  - b. default-information originate
  - c. default information-initiate
  - d. ospf information-originate
34. How can you disable OSPF auto summarization?
35. Router A is an ABR between area 0 and area 1. Write the command to summarize the following area 1 routes into area 0:
- 10.0.0.0/24
  - 10.0.1.0/24
  - 10.0.2.0/24
  - 10.0.3.0/24
  - 10.0.4.0/24
  - 10.0.5.0/24
  - 10.0.6.0/24
  - 10.0.7.0/24
36. What is an OSPF virtual link?
37. If your router has an interface faster than 100 Mbps that is used with OSPF, consider using the \_\_\_\_\_ command under the \_\_\_\_\_ process.
- a. **auto-cost reference-bandwidth, OSPF**
  - b. **auto-cost reference-bandwidth, interface**
  - c. **autocost reference-speed, OSPF**
  - d. **autocost reference-speed, interface**
38. How is the OSPF metric calculated on a Cisco router, by default?
- a. OSPF calculates the OSPF metric for the router according to the bandwidth of all its interfaces.
  - b. OSPF calculates the OSPF metric by referencing the DR.
  - c. OSPF calculates the OSPF metric for an interface according to the inverse of the interface's bandwidth.
  - d. OSPF calculates the OSPF metric by using the lowest bandwidth value among all of its interfaces.
39. Why is configuring a stub area advantageous?
- a. It reduces the size of the LSDB inside an area.
  - b. It increases the memory requirements for routers in that area.
  - c. It further segments the hierarchy.
  - d. It starts to behave like a distance vector routing protocol, thus speeding up convergence.

40. A stub area is typically created using what kind of topology?
  - a. Point to point
  - b. Broadcast
  - c. Hub and spoke
  - d. Full mesh
41. Which of the following would result in the smallest routing tables on OSPF internal routers?
  - a. Stub area
  - b. Totally stubby area
  - c. Standard area
  - d. Backbone area
42. What is the default OSPF authentication?
  - a. Simple password
  - b. MD5
  - c. Null
  - d. IPsec
43. Write the command to configure the password *cisco* for OSPF simple password authentication.
44. What command is used to troubleshoot OSPF authentication?
  - a. debug ip ospf adj
  - b. debug ip ospf auth
  - c. debug ip ospf md5
  - d. debug ip ospf packet
45. Write the command to configure the password *cisco* as key 1 for OSPF MD5 authentication.

## Chapter 4. Manipulating Routing Updates

This chapter discusses different means of controlling routing update information. It covers the following topics:

- Assessing Network Routing Performance Issues
- Using Multiple IP Routing Protocols on a Network
- Implementing Route Redistribution
- Controlling Routing Update Traffic

This chapter starts with a discussion of network performance issues related to routing and using multiple IP routing protocols on a network. Implementing route redistribution between different routing protocols is described, and methods of controlling the routing information sent between these routing protocols are explored, including using route maps, distribute lists, and prefix lists. The chapter concludes with a comprehensive example of controlling routing updates.

### Note

This chapter on manipulating routing updates is placed before the chapter on Border Gateway Protocol (BGP) because knowledge of route redistribution and route maps is required for the BGP discussion.

### Assessing Network Routing Performance Issues

This section discusses common network performance issues related to routing, and solutions to these issues.

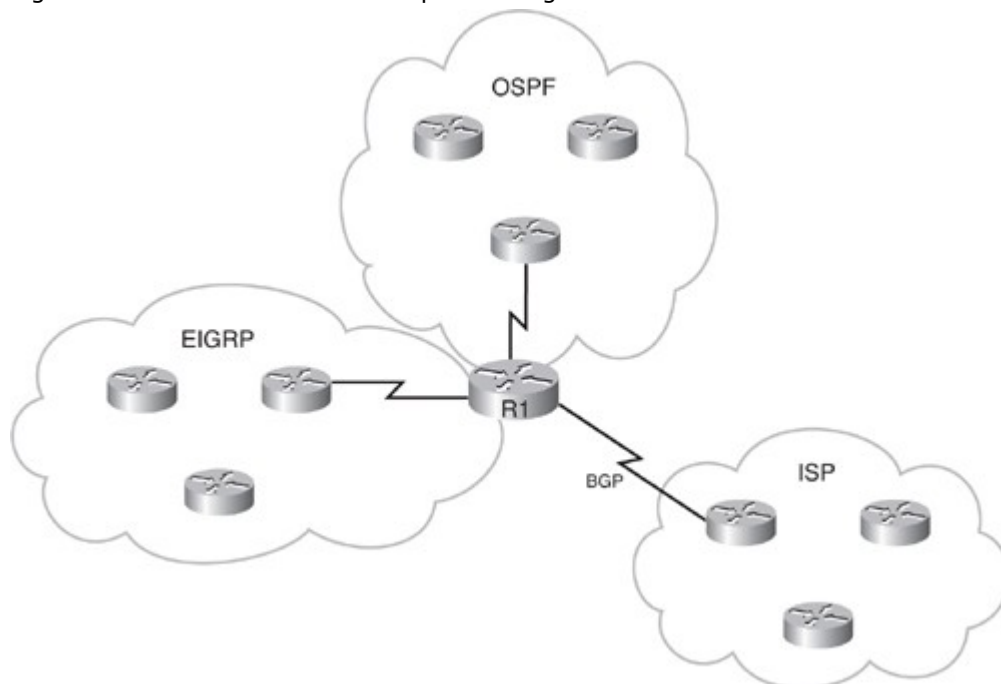
#### Routing Protocol Performance Issues

Common network performance issues related to routing include the following:

- **Excessive routing updates—** Excessive routing updates can decrease network performance, because if a Layer 3 switch or a router receives a large number of updates, it must process all of those updates. The size of the CPU utilization spike during this processing depends on the following factors:
  - The size of the routing update: This depends on the routing protocol used, the number of changes being reported, and the number of devices in the autonomous system (AS).
  - The frequency of the updates: This depends on the routing protocol used.
  - The design: The IP addressing plan and use of summarization can affect the number of routing updates sent.
- **The presence of any route maps or filters—** Incorrectly configured route maps or filters can cause too much or the wrong data to be sent.
- **The number of routing protocols running in the same autonomous system—** This affects the number of routing protocol processes receiving and processing the updates. Routes may also be redistributed between protocols, which can add to the number of updates that a specific protocol must process.

You can configure multiple routing protocols in a single router to connect networks that use different routing protocols. For example, you can run Enhanced Interior Gateway Routing Protocol (EIGRP) on one network, Open Shortest Path First (OSPF) on another network, and exchange routing information between them, in a controlled fashion. In addition, the same router can be connected to an Internet service provider (ISP) and exchange Border Gateway Protocol (BGP) routes with the ISP. As an example, Router R1 in Figure 4-1 runs EIGRP, OSPF, and BGP.

Figure 4-1. Routers Can Run Multiple Routing Protocols.



Routing protocols were not designed to interoperate with one another, so each protocol collects different types of information and reacts to topology changes in its own way, resulting in routing update traffic that must be processed by each protocol separately in a different way. For example, Routing Information Protocol (RIP) uses hop count as a metric, and OSPF uses link cost as a metric; these metrics are incompatible and routers exchanging routing information from these protocols must account for these differences. Along with high CPU utilization, more memory resources are needed to maintain all the routing, topology, and database tables in the routers running multiple routing protocols.

Cisco routers allow internetworks using different routing protocols (referred to as routing domains or autonomous systems) to exchange routing information through a feature called route redistribution. Route redistribution is the capability of boundary routers connecting different routing domains to exchange and advertise routing information between those routing domains and is detailed in the “Understanding Route Redistribution” section, later in this chapter.

#### Note

The term *autonomous system* as used here denotes internetworks using different routing protocols. These routing protocols may be Interior Gateway Protocols (IGPs) or Exterior Gateway Protocols (EGPs). This use of the term *autonomous system* is different from that used for BGP.

#### Routing Protocol Performance Solutions

Controlling routing updates involves a variety of solutions, including the following:

- Design changes, such as limiting the number of routing protocols used.
- Using passive interfaces, to prevent all updates from a routing protocol from being advertised out of an interface.
- Route filtering, to block only specific routes from being advertised, such as during redistribution. The following possibilities can be configured to filter routes:
  - Access control lists (ACLs)
  - Route maps
  - Distribute lists
  - Prefix lists

ACLs are usually associated with interfaces and are usually used to control *user* traffic (data plane traffic) rather than routing protocol traffic (or other control plane traffic). However, routers can have many interfaces, and route information can also be obtained through route redistribution, which does not involve a specific interface. In addition, access lists do not affect traffic *originated* by the router, so applying one on an interface has no effect on outgoing routing advertisements (but could affect incoming advertisements).



Using route maps, distribute lists, or prefix lists instead of access lists for route filtering gives the administrator greater flexibility in determining just which routes will be permitted and which will be denied. Route filtering works by regulating the routes that are entered into or advertised out of the routing table.

#### Note

Filtering affects link state routing protocols differently than distance vector protocols. For example, distance vector routers advertise and can filter routes from their routing tables. In contrast, link state routers send link-state advertisements (LSAs) and put received information in their link-state database (LSDB), from which they calculate their routing table. Because OSPF routers within the same area must have the same LSDB, LSA filtering for OSPF can be done only between areas.

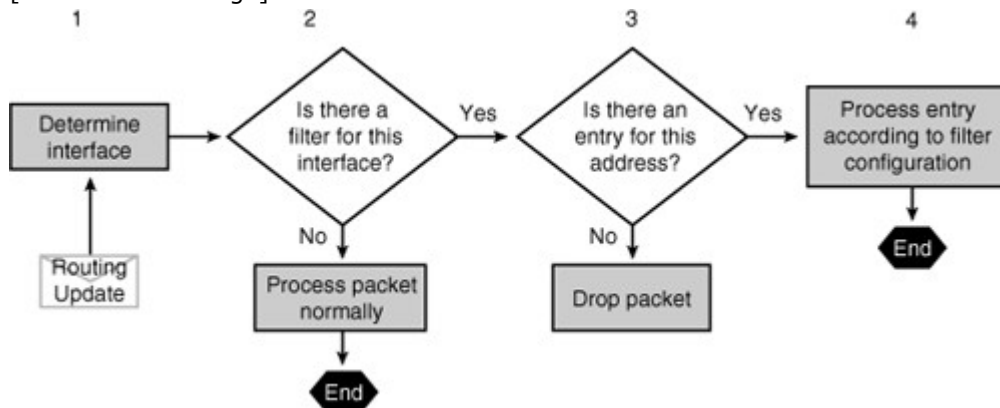
Filters can be configured to prevent updates through router interfaces, to control the advertising of routes in routing updates, or to control the processing of routing updates. If filters are not configured correctly or if filters are applied to wrong interfaces, network performance issues may occur.

For example, Figure 4-2 illustrates how inbound filtering is used to control incoming routing update traffic. The process is as follows:

1. A routing update arrives at a router's interface from a neighboring router. The router stores the packet in the interface buffer and triggers the CPU to make a decision.
2. The router's CPU checks if there is an incoming filter applied to this interface. If there is no filter, the routing update packet is processed normally.
3. If there is a filter, the router's CPU checks whether there is an entry for the address in the routing update packet in the filter. If there is no entry, the routing update is dropped.
4. If the entry exists, the router's CPU processes the routing update packet according to the filter configuration.

Figure 4-2. Incoming Route Filter Processing.

[View full size image]



Route redistribution is detailed in the next section. Passive interfaces, distribute lists, and prefix lists are all explored later in this chapter.

#### Using Multiple IP Routing Protocols on a Network

Simple routing protocols work well for simple networks, but as networks grow and become more complex, it might be necessary to change routing protocols, or to run multiple protocols. The transition to a new routing protocol may take place gradually, so multiple routing protocols might run in a network for some time. This section examines several reasons for using more than one routing protocol, how routing information is exchanged between them, and how Cisco routers operate in a multiple routing protocol environment.

#### Understanding a Network with Complex Routing

As a network grows and becomes more complex, the original routing protocol might not be the best choice anymore. For example, routers running RIP periodically send their entire routing tables in their updates. As the network grows larger, the traffic from those updates can slow the network down, such that that a change

to a more scalable routing protocol might be necessary. As another example, the network might be using Cisco's EIGRP and a protocol that supports multiple vendors might be required, or a new policy that specifies a particular routing protocol might be introduced.

The many reasons why a change in routing protocols or running multiple protocols might be required, resulting in a more complex routing environment, include the following:

- You are migrating from an older IGP to a new IGP. Multiple redistribution boundaries may exist until the new protocol has displaced the old protocol completely. Running multiple routing protocols during a migration is effectively the same as a network that has multiple routing protocols running as part of its design.
- You want to use another protocol but need to keep the old routing protocol because of the host system's needs. For example, UNIX host-based routers might run only RIP.
- Different departments might not want to upgrade their routers to support a new routing protocol.
- If you have a mixed-router vendor environment, you can use a Cisco-specific routing protocol, such as EIGRP, in the Cisco portion of the network and then use a common standards-based routing protocol, such as OSPF, to communicate with non-Cisco devices.
- Your network may become larger and more complex because of political and geographic border issues, and as you merge with or are acquired by other organizations.
- Your organization uses a Layer 3 Multiprotocol Label Switching (MPLS) virtual private network (VPN) service from a service provider that does not use the same IGP as the rest of your network.

Complex networks require careful routing protocol design and traffic optimization solutions, including the following:

- Redistribution between routing protocols
- Route filtering
- Summarization

The following sections describe redistribution. Route filtering is described in the "Controlling Routing Update Traffic" section, later in this chapter. Summarization is described in Chapter 2, "Configuring the Enhanced Interior Gateway Routing Protocol," and Chapter 3, "Configuring the Open Shortest Path First Protocol," for EIGRP and OSPF, respectively.

#### Understanding Route Redistribution

This section introduces redistribution and some considerations for implementing redistribution, and describes how routes are selected in a redistribution environment.

#### Redistribution Overview

When multiple routing protocols are running in different parts of the network, hosts in one part of the network might need to reach hosts in the other part. One way to accomplish this is to advertise a default route into each routing protocol, but default routes might not always be the best policy. For example, the network design might not allow default routes, and if there is more than one way to get to a destination network, routers might need information about routes in the other parts of the network to determine the best path to that destination. In addition, if multiple paths exist, a router must have sufficient information to determine a loop-free path to the remote networks.

When any of these situations arise, Cisco routers allow internetworks using different routing protocols (referred to as routing domains or autonomous systems) to exchange routing information through a feature called *route redistribution*. As mentioned earlier, route redistribution is defined as the capability of boundary routers connecting different routing domains to exchange and advertise routing information between those routing domains (autonomous systems).

Networks may run multiple routing protocols as part of their design, not only as part of a migration. In some cases the same protocol may be used in multiple different domains or autonomous systems within a network. The multiple instances of the protocol are treated no differently than if they were distinct protocols; redistribution is required to exchange routes between them. Thus, redistribution of routing information might be required in many networks.

Network administrators must manage the migration from one routing protocol to another, or to multiple protocols, carefully and thoughtfully. An accurate topology map of the network and an inventory of all network devices are critical for success. When redistributing between routing protocols, the two routing protocols will most likely have different requirements and capabilities, so it is important for network administrators to create a detailed plan before making any routing protocol changes. For example, link-state routing protocols, such as OSPF and Intermediate System-to-Intermediate System (IS-IS), require a hierarchical network structure; network administrators need to decide which routers will reside in the

backbone area and how to divide the other routers into areas. Although EIGRP does not require a hierarchical structure, it operates much more effectively within one.

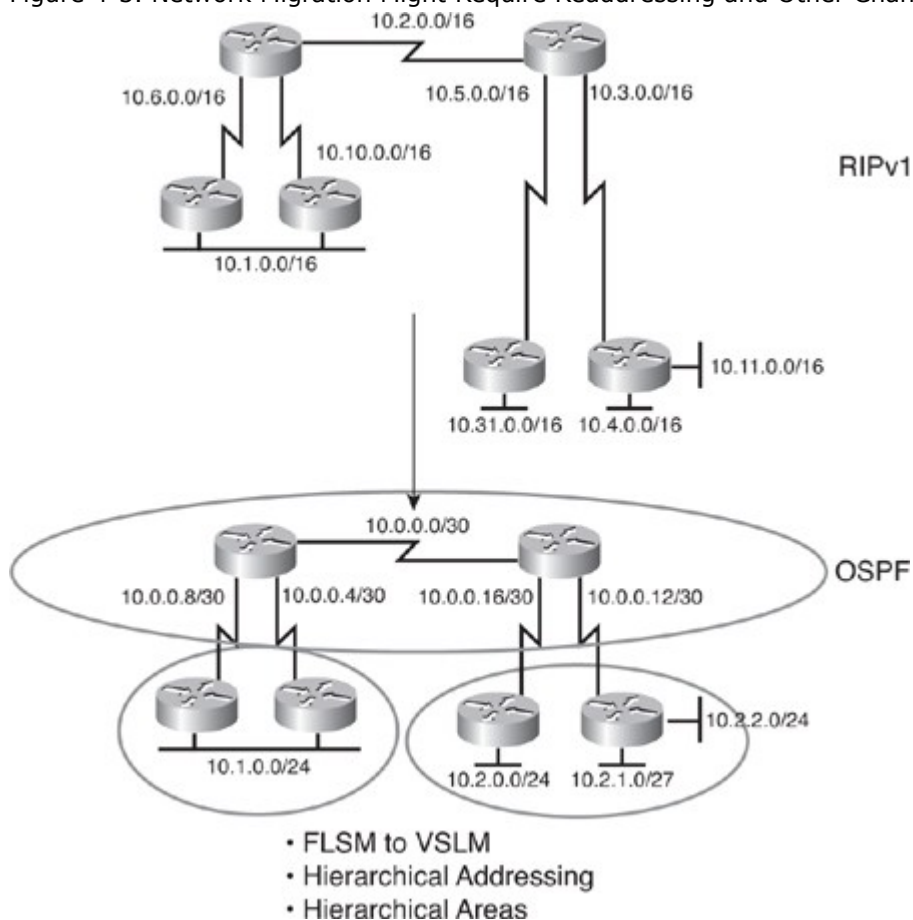
Note

IS-IS is included in this chapter for completeness, in tables and so forth. However, IS-IS configuration and redistribution are not covered in this book.

Figure 4-3 shows a sample network migration. This network initially used RIP Version 1 (RIPv1) and is migrating to OSPF, necessitating the following changes:

- Conversion of the old fixed-length subnet mask (FLSM) addressing scheme to a variable-length subnet mask (VLSM) configuration
- Use of a hierarchical addressing scheme to facilitate route summarization and make the network more scalable
- Division of the network from one large area into a transit backbone area and two other areas

Figure 4-3. Network Migration Might Require Readdressing and Other Changes.



Within each autonomous system, the internal routers have complete knowledge of their network. The router that interconnects the autonomous systems is called a *boundary router* (also called an *edge router*). The boundary router must be running all the routing protocols between which routes must be exchanged. In most cases, route redistribution must be configured to redistribute routes from one routing protocol to another. The only time redistribution is automatic in IP routing protocols is between Interior Gateway Routing Protocol (IGRP) and EIGRP processes running on the same router and using the same autonomous system number.

Note

IGRP is no longer supported, as of Cisco IOS Release 12.3. It is included in this chapter for completeness.

#### Redistributed Routes

When a router redistributes routes, it propagates routes learned by one routing source into a routing domain that uses a different routing protocol. These redistributed routes could have been learned via a different

routing protocol, such as when redistributing between EIGRP and OSPF, or they could have been learned from static routes or by a direct connection to a network. (Routers can redistribute static and connected routes and routes from other routing protocols.)

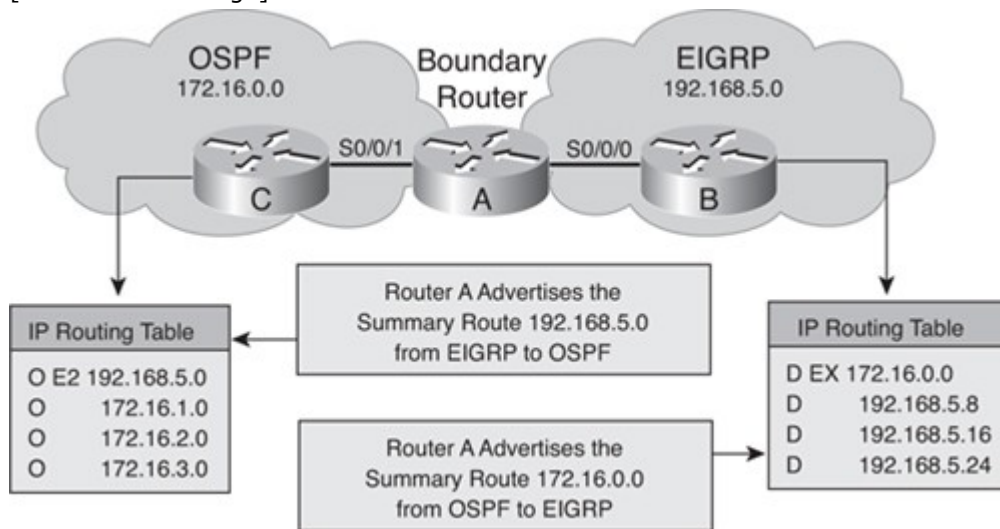
Redistribution is always performed *outbound*; the router doing redistribution does not change its own routing table. For example, when redistribution between OSPF and EIGRP is configured, the OSPF process on the boundary router takes the EIGRP routes in the routing table and advertises them as OSPF routes to its OSPF neighbors. Likewise, the EIGRP process on the boundary router takes the OSPF routes in the routing table and advertises them as EIGRP routes to its EIGRP neighbors. With this redistribution, both autonomous systems know about the routes of the other, and each autonomous system can then make informed routing decisions for these networks. The boundary router's neighbors see the redistributed routes as external routes. In this example, if a packet destined for one of the networks in the OSPF domain arrives from the EIGRP autonomous system, the boundary router must have the OSPF routes for the networks in the OSPF domain in its routing table to be able to forward the traffic.

It is important to note that routes must be in the routing table for them to be redistributed. This requirement might seem self-evident, but it can be a source of confusion. For instance, if a router learns about a network via EIGRP and OSPF, by default only the EIGRP route is put in the routing table because it has a lower administrative distance. Suppose RIP is also running on this router, and you are redistributing OSPF routes into RIP. That network is not redistributed into RIP, because it is placed in the routing table as an EIGRP route, not as an OSPF route.

Figure 4-4 illustrates an autonomous system running OSPF that is connected to an autonomous system running EIGRP. The internal routers within each autonomous system have complete knowledge of their networks, but without redistribution, they do not know about the routes present in the other autonomous system. Router A is the boundary router, and it has active OSPF and EIGRP processes.

Figure 4-4. Redistribution Between OSPF and EIGRP.

[View full size image]



Without redistribution, Router A performs ships-in-the-night (SIN) routing: Router A passes OSPF route updates to its OSPF neighbors on the interfaces participating in OSPF, and it passes EIGRP route updates to its EIGRP neighbors on the interfaces participating in EIGRP. Router A does not exchange information between EIGRP and OSPF. If routers in the OSPF routing domain need to learn about the routes in the EIGRP domain, or vice versa, Router A must redistribute routes between EIGRP and OSPF.

Router A learns about network 192.168.5.0 from Router B via the EIGRP routing protocol running on its S0/0/0 interface. After redistribution is configured, Router A redistributes that information to Router C via OSPF on its S0/0/1 interface. Routing information is also passed in the other direction, from OSPF to EIGRP. The routing table in Router B shows that it has learned about network 172.16.0.0 via EIGRP (as indicated by the D in the routing table) and that the route is external to this autonomous system (as indicated by the EX in the routing table). The routing table in Router C shows that it has learned about network 192.168.5.0 via OSPF (as indicated by the O in the routing table) and that the route is external (type 2) to this autonomous system (as indicated by the E2 in the routing table).

Note that in this example Router A is redistributing routes that are summarized on the network class boundary, to help improve routing table stability and decrease the routing tables' size. (Recall that EIGRP

automatically summarizes on the class boundary, whereas OSPF must be configured to summarize. EIGRP does not automatically summarize redistributed routes though; if they are not already summarized before they are redistributed then summarization must be configured if it is desired.)

#### Note

As mentioned in Chapter 1, “Routing Services,” the Cisco IOS documentation for EIGRP says that automatic summarization is now disabled by default. However, testing has confirmed it is still on, at least in some versions of the IOS. Thus, it would be prudent to confirm the auto summary configuration, or to configure it explicitly.

#### Redistribution Implementation Considerations

Redistribution of routing information, although powerful, adds to a network’s complexity and increases the potential for routing confusion, so it should be used only when necessary. The key issues that arise when using redistribution are as follows:

- **Routing feedback (loops)**— Depending on how you employ redistribution—for example, if more than one boundary router is performing route redistribution—routers might send routing information received from one autonomous system back into that same autonomous system. This feedback is similar to the routing loop problem that occurs with distance vector protocols.
- **Incompatible routing information**— Because each routing protocol uses different metrics to determine the best path and because the metric information about a route cannot be translated exactly into a different protocol, path selection using the redistributed route information might not be optimal.
- **Inconsistent convergence times**— Different routing protocols converge at different rates. For example, RIP converges more slowly than EIGRP, so if a link goes down, the EIGRP network learns about it before the RIP network.

Good planning is able to solve the majority of issue but additional configuration might be required. Some issues might be solved by changing the administrative distance, manipulating the metrics, and filtering using route maps, distribute lists, and prefix lists. (These topics are all covered in this chapter.)

Understanding why some of these problems might occur requires understanding how Cisco routers select the best path when more than one routing protocol is running and how they convert the metrics used when importing routes from one autonomous system into another. These topics are discussed in the following sections.

#### Selecting the Best Route in a Redistribution Environment

Recall from Chapter 1 that Cisco routers use the following two parameters to select the best path when they learn two or more routes to the same destination (with the same prefix length) from different routing protocols:

- **Administrative distance**— The administrative distance is used to rate a routing protocol’s *believability* (also called its *trustworthiness*). Each routing protocol is prioritized in order from most to least believable using a value called the administrative distance. This criterion is the first thing a router uses to determine which routing protocol to believe if more than one protocol provides route information for the same destination.
- **Routing metric**— The routing metric is a value representing the path between the local router and the destination network, according to the routing protocol being used. The metric is used to determine the routing protocol’s “best” path to the destination.

#### Administrative Distance

Table 4-1 lists the default administrative distance of protocols supported by Cisco (this is a copy of Table 1-6). Lower administrative distances are considered more believable (better).

Table 4-1. Default administrative distances of Routing Protocols

| Route Source                       | Default Administrative Distance |
|------------------------------------|---------------------------------|
| Connected interface                | 0                               |
| Static route out an interface[1]   | 1                               |
| Static route to a next-hop address | 1                               |
| EIGRP summary route                | 5                               |

| Route Source   | Default Administrative Distance |
|----------------|---------------------------------|
| External BGP   | 20                              |
| Internal EIGRP | 90                              |
| IGRP[2]        | 100                             |
| OSPF           | 110                             |
| IS-IS          | 115                             |
| RIPv1, RIPv2   | 120                             |
| EGP[3]         | 140                             |
| ODR            | 160                             |
| External EIGRP | 170                             |
| Internal BGP   | 200                             |
| Unreachable    | 255                             |

[1] See the Note following Table 1-6 for an explanation of the administrative distances of static routes.

[2] IGRP is no longer supported, as of Cisco IOS Release 12.3. It is included in this table for completeness.

[3] EGP is no longer supported. It is included in this table for completeness.

When using route redistribution, you might occasionally need to modify a protocol's administrative distance so that it is preferred. For example, if you want the router to select RIP-learned routes rather than OSPF-learned routes for some specific destination, you can increase the OSPF administrative distance or decrease the RIP administrative distance for the routes to that destination. Modifying the administrative distance is discussed in the section "Using Administrative Distance to Influence the Route-Selection Process," later in this chapter.

#### Seed Metrics

When a router advertises a link that is directly connected to one of its interfaces, the initial, or *seed* metric (also called the default metric) used is derived from the characteristics of that interface, and the metric increments as the routing information is passed to other routers.

For OSPF, the seed metric is based on the interface's bandwidth. For IS-IS, each interface has a default IS-IS metric of 10. For EIGRP and IGRP, the default seed metric is based on the interface bandwidth and delay. For RIP, the seed metric starts with a hop count of zero and increases in increments from router to router.

When a router is redistributing, the redistributed route must have a metric appropriate for the receiving protocol.

Because redistributed routes are learned from other sources (such as other routing protocols), a boundary router must be capable of translating the metric of the received route from the source routing protocol into the receiving routing protocol. For example, if a boundary router receives a RIP route, the route has hop count as a metric. To redistribute the route into OSPF, the router must translate the hop count into a cost metric that the other OSPF routers will understand.

The seed or default metric is defined during redistribution configuration. After the seed metric for a redistributed route is established, the metric increments normally within the autonomous system. (The exception to this rule is OSPF E2 routes, which hold their initial metric regardless of how far they are propagated across an autonomous system.)

The **default-metric** router configuration command establishes the seed metric for all redistributed routes. Cisco routers also allow the seed metric to be specified as part of the **redistribute** command, either with the **metric** option or by using a route map. These commands are discussed in detail in the section "Configuring Redistribution," later in this chapter.

When you are redistributing routing information, the seed metric should be set to a value larger than the largest metric within the receiving autonomous system (also called the largest native metric), to help prevent



suboptimal routing and routing loops. For example, if RIP routes are being redistributed into OSPF and the highest OSPF metric is 50, the redistributed RIP routes should be assigned an OSPF metric higher than 50.

#### Default Seed Metrics

Table 4-2 lists the default seed metric value for routes that are redistributed into each IP routing protocol. A metric of infinity tells the router that the route is unreachable and, therefore, should not be advertised. Therefore, when redistributing routes into RIP, IGRP, and EIGRP, you must specify a seed metric; otherwise, the redistributed routes will not be advertised.

Table 4-2. Default Seed Metrics

| Protocol That Route Is Redistributed Into | Default Seed Metric                                                                                                                                                                                                       |
|-------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RIP                                       | 0, which is interpreted as infinity.                                                                                                                                                                                      |
| IGRP/EIGRP                                | 0, which is interpreted as infinity.                                                                                                                                                                                      |
| OSPF                                      | 20 for all except BGP routes, which have a default seed metric of 1 (all default to type E2). (Note that when redistributing OSPF into OSPF, metrics associated with both intra-area and interarea routes are preserved.) |
| IS-IS                                     | 0.                                                                                                                                                                                                                        |
| BGP                                       | BGP metric is set to IGP metric value.                                                                                                                                                                                    |

#### Note

Redistribution between EIGRP and IGRP with the same autonomous system number is automatic and does not require a metric to be specified.

For OSPF, the redistributed routes have a default type 2 (E2) metric of 20, except for redistributed BGP routes, which have a default type 2 metric of 1. (Note that when redistributing OSPF into OSPF, metrics associated with both intra-area and interarea routes are preserved.)

For IS-IS, the redistributed routes have a default metric of 0. But unlike RIP, IGRP, or EIGRP, a seed metric of 0 is not treated as unreachable by IS-IS. Configuring a seed metric for redistribution into IS-IS is recommended.

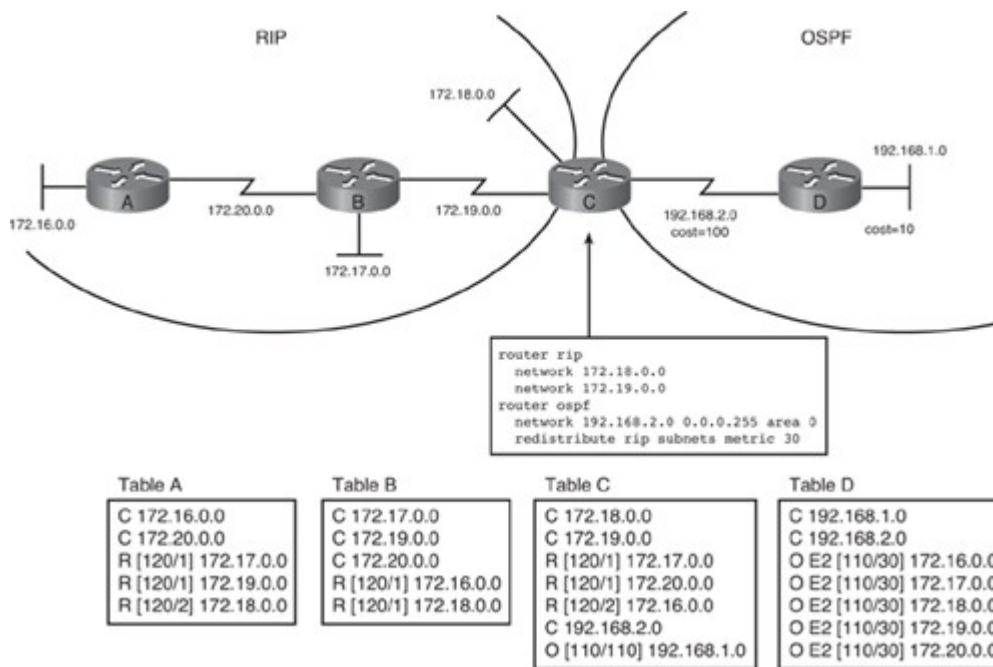
For BGP, the redistributed routes maintain the IGP routing metrics.

Figure 4-5 illustrates an example with an OSPF seed metric of 30 for redistributed RIP routes on Router C. The link cost of the Serial link to Router D is 100. The routes are redistributed as E2 routes, so the cost for networks 172.16.0.0, 172.17.0.0, and 172.18.0.0 in Router D is only the seed metric (30). Notice that the metrics of the three networks in the RIP cloud are irrelevant in the OSPF cloud because the router in the OSPF network (Router D) forwards any traffic for these three networks to the border (redistributing) router, Router C. Router C then forwards the traffic within the RIP network appropriately.

Figure 4-5. Redistribution Between OSPF and RIP.

[View full size image]





## Redistribution Techniques

This section describes one-point and multipoint redistribution techniques, and how to prevent loops in a redistribution environment.

### One-Point Redistribution

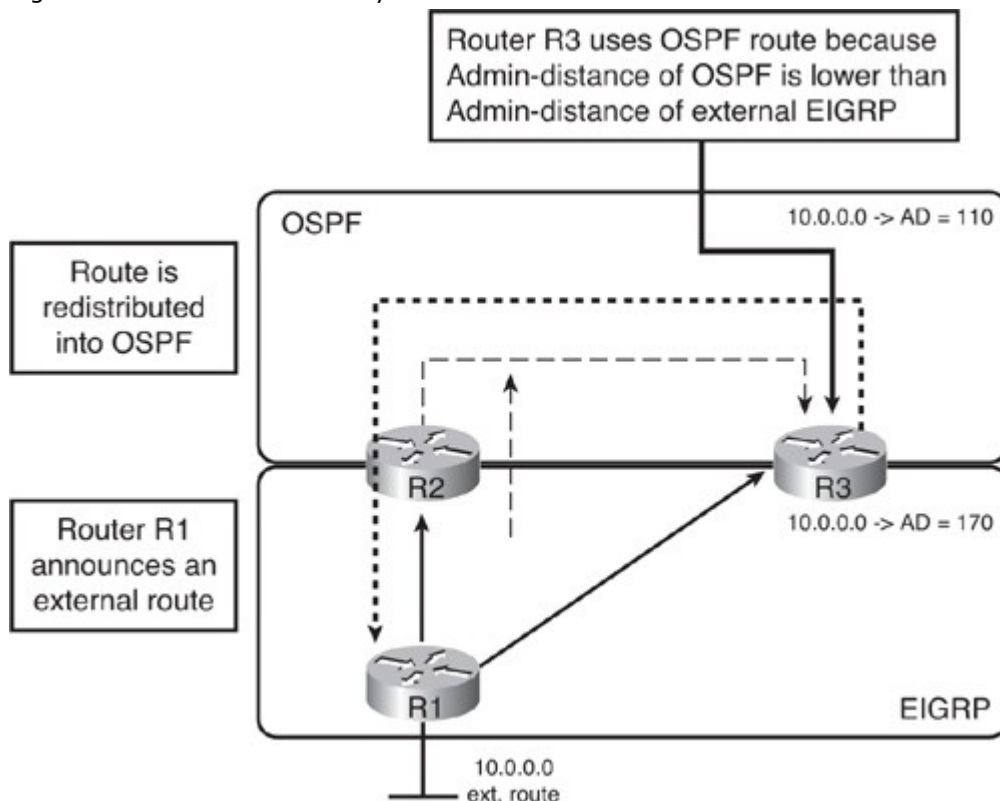
One-point redistribution has only one router redistributing between two routing protocols. The following two methods of one-point redistribution are available:

- **Two-way redistribution**— Redistributes routes between the two routing processes in both directions
- **One-way redistribution**— Redistributes only the networks learned from one routing protocol into the other routing protocol; uses a default or static route so that devices in that other part of the network can reach the first part of the network

One-point one-way or two-way redistribution is always safe (in that the network will function correctly) because, there is only one way between routing protocols so routing loops cannot be created.

However, issues can still occur. For example, Figure 4-6 illustrates a network with one-point one-way redistribution. Router R1 running EIGRP is announcing an external route to Routers R2 and R3. Both of these routers are running the two routing protocols OSPF and EIGRP, but the redistribution between EIGRP and OSPF occurs only on Router R2. Router R3 receives routing update information for the external route directly from Router R1 via EIGRP, and via OSPF from Router R2. The administrative distance of OSPF (110) is lower than administrative distance of external EIGRP routes (170), so Router R3 selects the OSPF route. Instead of sending packets directly from Router R3 to Router R1, Router R3 prefers the path via Router R2, resulting in suboptimal routing.

Figure 4-6. One-Point One-Way Redistribution Issue.



#### Multipoint Redistribution

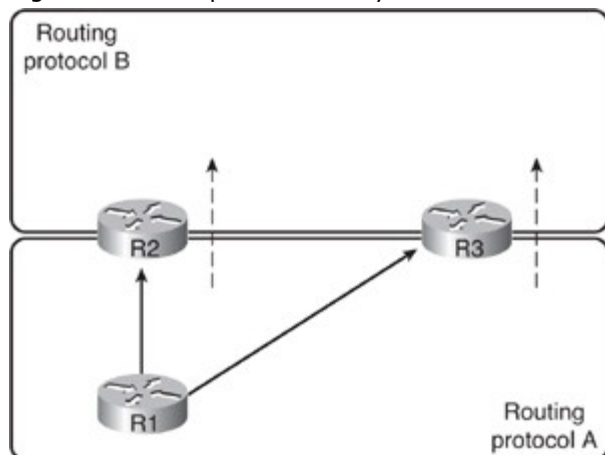
Multipoint redistribution has two separate routers running both routing protocols. Two possibilities exist:

- Multipoint one-way redistribution
- Multipoint two-way redistribution

Multipoint redistribution is likely to introduce potential routing loops. Even multipoint one-way redistribution is dangerous; multipoint two-way redistribution is highly problematic. Typical problems with multipoint redistribution involve the difference in the administrative distances of the protocols and their incompatible metrics, especially when statically assigned seed metrics are used in redistribution points.

Consider Figure 4-7, with both R2 and R3 redistributing protocol A into protocol B. Multipoint one-way redistribution only works well if

Figure 4-7. Multipoint One-Way Redistribution.

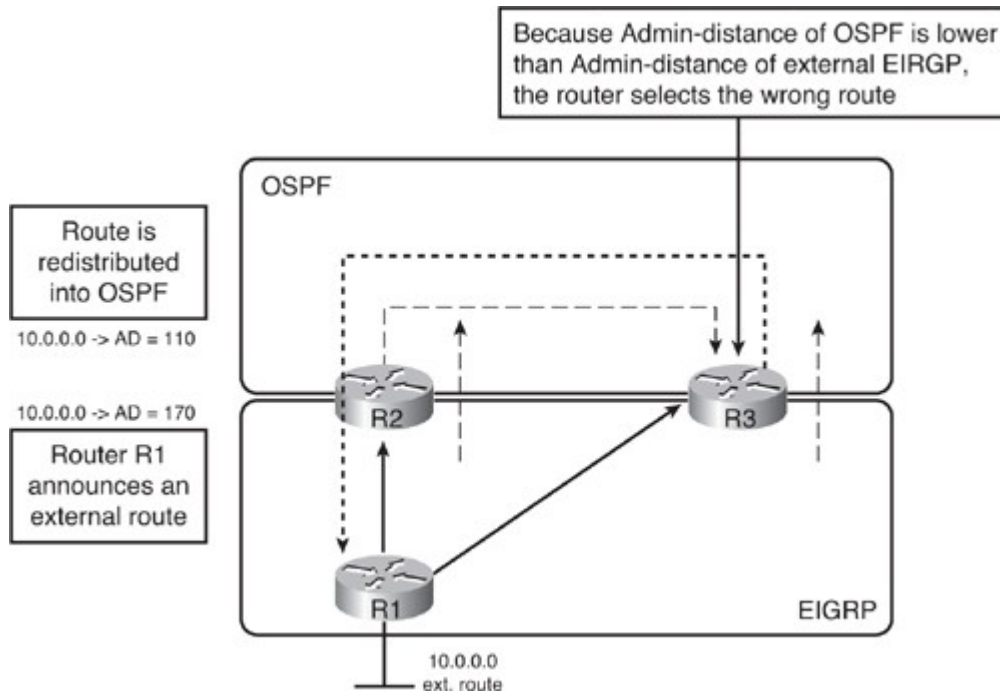


- The receiving routing protocol supports different administrative distances for internal and external routes. EIGRP, BGP and OSPF do this.

- The administrative distance of protocol B's external routes is higher than the administrative distance of protocol A's routes, so that R2 and R3 will use the appropriate routes to destinations in the protocol A side of the network.

Figure 4-8 illustrates a one-way multipoint redistribution issue. Router R1 from EIGRP is announcing routes, including the external route, to Routers R2 and R3. Both of these routers are running two routing protocols and are redistributing EIGRP into OSPF. Therefore, Routers R2 and R3 receive routing update information for the external route via EIGRP from Router R1 and via OSPF (Router R2 from Router R3, and Router R3 from Router R2). The administrative distance of OSPF (110) is lower than administrative distance of external EIGRP (170), so both Router R2 and Router R3 select the OSPF route. For example, instead of sending packets directly from Router R2 to Router R1, Router R2 prefers the path via Router R3. As a best case, this would be suboptimal routing, but because R3 sends the packet back to R2, a routing loop exists.

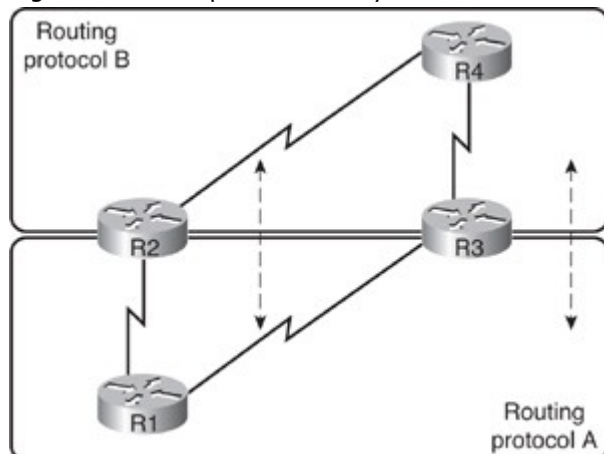
Figure 4-8. One-Way Multipoint Redistribution Issue.



Consider Figure 4-9, with both R2 and R3 redistributing protocol A into protocol B, and protocol B into protocol A. Multipoint two-way redistribution difficulties include the following:

- Suboptimal routing (because only part of the total metric is considered in routing decisions, from the redistribution point onward).
- Self-sustaining routing loops are possible.

Figure 4-9. Multipoint Two-Way Redistribution.



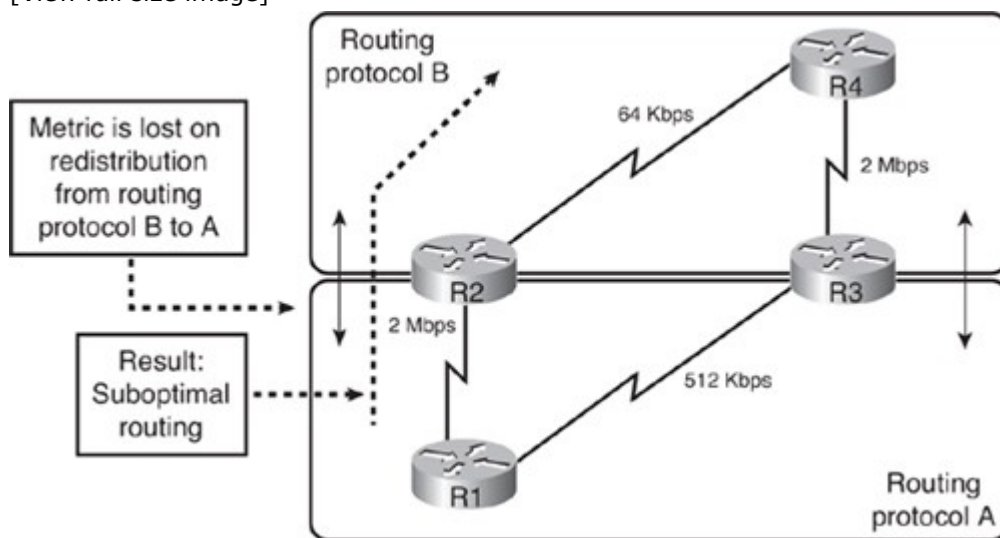
To prevent routing loops in multipoint redistribution scenario the following recommendations should be considered:

- Insert only internal routes from routing protocol A to B (and vice versa).
- Tag routes in redistribution points and filter based on these tags when configuring redistribution in the other direction.
- Propagate metrics from routing protocol A to routing protocol B properly (even though this is not sufficient to prevent loops).
- Use default routes to avoid having to do two-way redistribution.

Figure 4-10 illustrates a two-way multipoint redistribution issue where the cost of the internal links in routing protocol A is different from the cost of the internal links in routing protocol B. In the figure, it is obvious that the best path between Router R1 and Router R4 is via Router R3, but during redistribution from routing protocol B to routing protocol A, the metric is lost and Router R1 is sending the packets toward Router R4 via Router R2, resulting in suboptimal routing.

Figure 4-10. Two-Way Multipoint Redistribution Issue.

[View full size image]



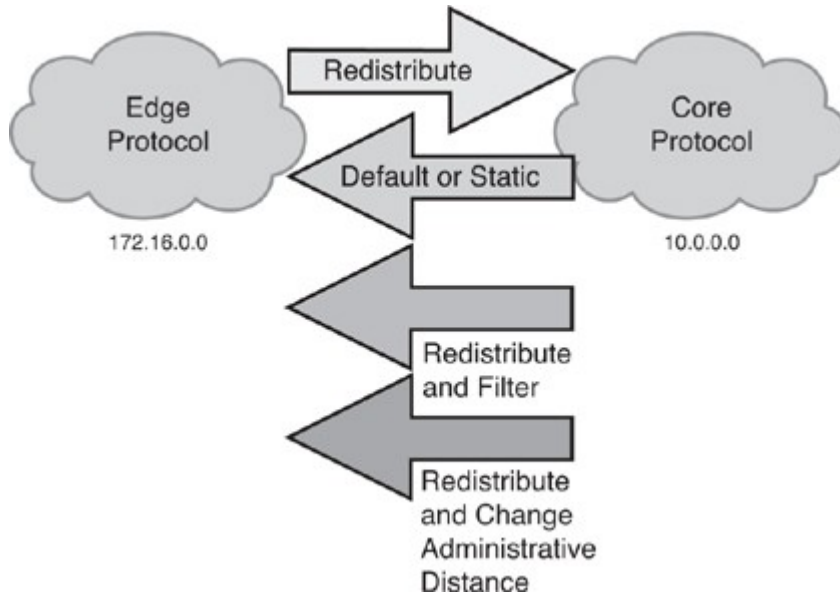
#### Preventing Routing Loops in a Redistribution Environment

The safest way to perform redistribution is to redistribute routes in only one direction, on only one boundary router within the network. (Note, however, that this results in a single point of failure in the network.)

If redistribution must be done in both directions or on multiple boundary routers, the redistribution should be tuned to avoid problems such as suboptimal routing and routing loops.

When discussing redistribution, we use the terms core routing protocol and edge routing protocol. The core routing protocol is the main routing protocol running in the network. During a transition between routing protocols, the core is the new routing protocol and the edge routing protocol is the old routing protocol. In networks that run multiple routing protocols all the time, the core is usually the more advanced routing protocol. Depending on your network design, you may use any of the following redistribution techniques, as illustrated in Figure 4-11:

Figure 4-11. Redistribution Techniques.



- Redistribute a default route from the core autonomous system into the edge autonomous system, and redistribute routes from the edge routing protocols into the core routing protocol. This technique helps prevent route feedback, suboptimal routing, and routing loops.
- Redistribute multiple static routes about the core autonomous system networks into the edge autonomous system, and redistribute routes from the edge routing protocols into the core routing protocol. This method works if there is only one redistribution point; multiple redistribution points might cause route feedback.
- Redistribute routes from the core autonomous system into the edge autonomous system with filtering to block out inappropriate routes. For example, when there are multiple boundary routers, routes redistributed from the edge autonomous system at one boundary router should not be redistributed back into the edge autonomous system from the core at another redistribution point.
- Redistribute all routes from the core autonomous system into the edge autonomous system, and from the edge autonomous system into the core autonomous system, and then modify the administrative distance associated with redistributed routes so that they are not the selected routes when multiple routes exist for the same destination.

Recall that if two routing protocols advertise routes to the same destination, information from the routing protocol with the lowest administrative distance is placed in the routing table. A route redistributed into a routing protocol by default inherits the default administrative distance of that routing protocol.

#### Implementing Route Redistribution

As shown in Example 4-1, redistribution supports all routing protocols. Static and connected routes can also be redistributed to allow the routing protocol to advertise these routes.

##### Example 4-1. Redistribution Supports All Protocols

```
RtrA(config)#router rip
RtrA(config-router)#redistribute ?
 bgp Border Gateway Protocol (BGP)
 connected Connected
```

|           |                                                    |
|-----------|----------------------------------------------------|
| eigrp     | Enhanced Interior Gateway Routing Protocol (EIGRP) |
| isis      | ISO IS-IS                                          |
| iso-igrp  | IGRP for OSI networks                              |
| metric    | Metric for redistributed routes                    |
| mobile    | Mobile routes                                      |
| odr       | On Demand stub Routes                              |
| ospf      | Open Shortest Path First (OSPF)                    |
| rip       | Routing Information Protocol (RIP)                 |
| route-map | Route map reference                                |
| static    | Static routes                                      |
| <cr>      |                                                    |

#### Note

Note that IGRP is not in this list because it is no longer supported as of Cisco IOS Release 12.3.

It is important to note that routes are redistributed *into* a routing protocol, so the **redistribute** command is configured under the routing process that is to *receive* the redistributed routes.

#### Configuring Route Redistribution

Before implementing redistribution, consider the following points:

- You can only redistribute routes from routing protocols that support the same protocol stack. For example, you can redistribute between IP Version 4 (IPv4) RIP and OSPF because they both support the TCP/IP stack. You cannot redistribute between IP Version 6 (IPv6) RIP new generation (RIPng) and IPv4 RIP because IPv6 RIPng supports only the IPv6 stack and IPv4 RIP supports only the IPv4 stack. Although there are different protocol-dependent modules of EIGRP for IPv4, IPv6, IPX, and AppleTalk, routes cannot be redistributed between them, because each protocol-dependent module supports a different protocol stack.
- The method you use to configure redistribution varies among combinations of routing protocols. For example, redistribution would occur automatically between IGRP and EIGRP processes with the same autonomous system number, but redistribution must be configured between all other routing protocols. Some routing protocols require a metric to be configured during redistribution, but others do not.

The following steps for configuring redistribution are generic enough to apply to all routing protocol combinations. However, the commands used to implement the steps vary, as identified in the following sections. It is important that you review the Cisco IOS documentation for the configuration commands that apply to the specific routing protocols you want to redistribute.

#### Note

Remember, in this chapter, the terms *core* and *edge* are generic terms used to simplify the discussion of redistribution.

- Step 1. Locate the boundary routers on which redistribution is to be configured. As discussed, selecting a single boundary router for redistribution minimizes the likelihood of routing loops caused by feedback.
- Step 2. Determine which routing protocol is the core or backbone protocol. Typically, this protocol is OSPF or EIGRP.
- Step 3. Determine which routing protocol is the edge protocol (or short-term protocol if you are migrating). Determine whether all routes from the edge protocol need to be propagated into the core. Consider methods that reduce the number of routes.
- Step 4. Select a method for injecting the required edge protocol routes into the core. Simple redistribution using summarized routes at network boundaries minimizes the number of new entries in the routing table of the core routers.
- Step 5. After you have planned the edge-to-core redistribution, consider how to inject the core routing information into the edge protocol. Your choice depends on your network.

The following sections examine the specific commands for redistributing routes into the various IP routing protocols.

#### Redistributing into RIP

Use the redistribute protocol [process-id] [match route-type] [metric metric-value] [route-map map-tag] router configuration command to redistribute routes into RIP. The parameters for this command are explained in Table 4-3.

Table 4-3. *redistribute* Command for RIP

| Parameter    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| protocol     | The source protocol from which routes are redistributed. It can be one of the following keywords: <b>bgp</b> , <b>connected</b> , <b>eigrp</b> , <b>isis</b> , <b>iso-igrp</b> , <b>mobile</b> , <b>odr</b> , <b>ospf</b> , <b>rip</b> , or <b>static</b> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| process-id   | For BGP or EIGRP, this value is an autonomous system number. For OSPF, this value is an OSPF process ID. This parameter is not required for IS-IS.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| route-type   | (Optional) A parameter used when redistributing OSPF routes into another routing protocol. It is the criterion by which OSPF routes are redistributed into other routing domains. It can be any of the following:<br><b>internal</b> —Redistributes routes that are internal to a specific autonomous system<br><b>external 1</b> —Redistributes routes that are external to the autonomous system but are imported into OSPF as a type 1 external route<br><b>external 2</b> —Redistributes routes that are external to the autonomous system but are imported into OSPF as a type 2 external route                                                                                                                                                                     |
| metric-value | (Optional) A parameter used to specify the RIP seed metric for the redistributed route. When redistributing into RIP (and all protocols other than OSPF and BGP), if this value is not specified and no value is specified using the <b>default-metric</b> router configuration command, the default metric is 0. For RIP (and all protocols other than IS-IS), the default metric of 0 is interpreted as infinity, and routes will not be redistributed. The metric for RIP is hop count.<br><br><b>Note:</b> When redistributing into RIP, the default metric is infinity except when redistributing a static route (including a default static route defined using the ip route 0.0.0.0 0.0.0.0 command) or connected route. In these cases, the default metric is 1. |
| map-tag      | (Optional) Specifies the identifier of a configured route map to be interrogated to filter the importation of routes from the source routing protocol to the current RIP routing protocol.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

Example 4-2 shows how to configure redistribution from OSPF process 1 into RIP. This example uses the router rip command to access the routing process into which routes need to be redistributed—the RIP routing process. The redistribute command is then used to specify the routing protocol to be redistributed into RIP. In this case, it is OSPF routing process number 1.

Example 4-2. Configuring Redistribution into RIP

```
RtrA(config)#router rip
RtrA(config-router)#redistribute ospf ?

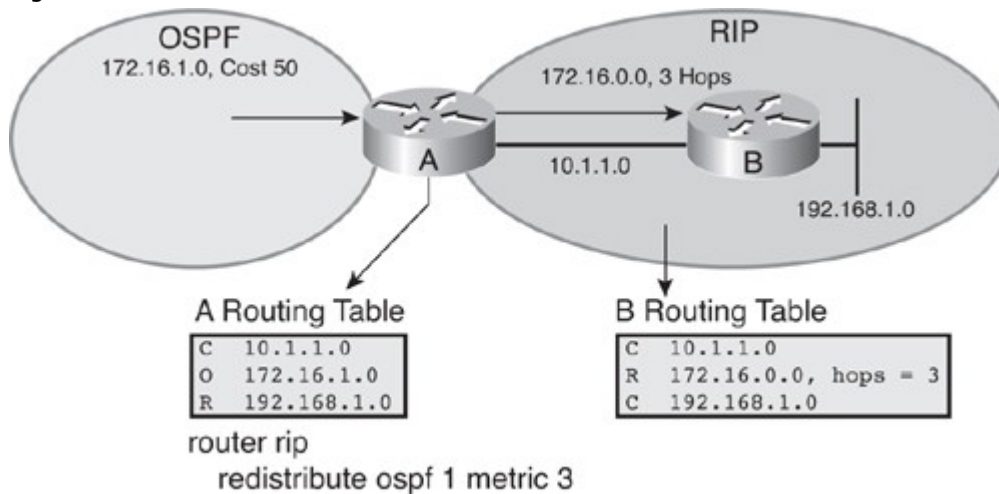
<1-65535> Process ID
RtrA(config-router)#redistribute ospf 1 ?
match Redistribution of OSPF routes
metric Metric for redistributed routes
route-map Route map reference
```



```
...
<cr>
```

Figure 4-12 provides an example of redistributing routes into RIP. On Router A, routes from OSPF process 1 are redistributed into RIP and are given a seed metric of 3. Because no route type is specified, both internal and external OSPF routes are redistributed into RIP. Notice that Router B learns about the 172.16.0.0 network from Router A via RIP; Router B's routing table has 172.16.0.0 installed as a RIP route.

Figure 4-12. Routes Redistributed into RIP.



#### Note

Notice that for RIP, the metric advertised to a router (3 in this case) is what that router uses as its metric. The sending router is assumed to have added 1 to the hop count. The receiving router does not add another hop. Notice also that the route is automatically summarized by Router A.

#### Redistributing into OSPF

Use the `redistribute protocol [process-id] [metric metric-value] [metric-type type-value] [route-map map-tag] [subnets] [tag tag-value]` router configuration command to redistribute routes into OSPF. The parameters for this command are explained in Table 4-4.

Table 4-4. `redistribute` Command for OSPF

| Parameter    | Description                                                                                                                                                                                                                                                                                                                                                              |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| protocol     | The source protocol from which routes are redistributed. It can be one of the following keywords: <b>bgp</b> , <b>connected</b> , <b>eigrp</b> , <b>isis</b> , <b>iso-igrp</b> , <b>mobile</b> , <b>odr</b> , <b>ospf</b> , <b>rip</b> , or <b>static</b> .                                                                                                              |
| process-id   | For BGP or EIGRP, this value is an autonomous system number. For OSPF, this value is an OSPF process ID. This parameter is not required for RIP or IS-IS.                                                                                                                                                                                                                |
| metric-value | (Optional) A parameter that specifies the OSPF seed metric used for the redistributed route. When redistributing into OSPF, the default metric is 20 (except for BGP routes, which have a default metric of 1). The metric for OSPF is cost. (Note that when redistributing OSPF into OSPF, metrics associated with both intra-area and interarea routes are preserved.) |
| type-value   | (Optional) An OSPF parameter that specifies the external link type associated with the external route advertised into the OSPF routing domain. This value can be <b>1</b> for type 1 external routes or <b>2</b> for type 2 external routes. The default is <b>2</b> .                                                                                                   |
| map-tag      | (Optional) Specifies the identifier of a configured route map to be                                                                                                                                                                                                                                                                                                      |

| Parameter | Description                                                                                                                                                                                                                 |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|           | interrogated to filter the importation of routes from the source routing protocol to the current OSPF routing protocol.                                                                                                     |
| subnets   | (Optional) An OSPF parameter that specifies that subnetted routes should also be redistributed. Only routes that are not subnetted are redistributed if the <b>subnets</b> keyword is not specified.                        |
| tag-value | (Optional) A 32-bit decimal value attached to each external route. The OSPF protocol itself does not use this parameter; it may be used to communicate information between OSPF Autonomous System Boundary Routers (ASBRs). |

Example 4-3 shows how to configure redistribution from EIGRP autonomous system 100 into OSPF. This example uses the `router ospf 1` command to access the OSPF routing process 1 into which routes need to be redistributed. The `redistribute` command is then used to specify the routing protocol to be redistributed into OSPF—in this case, the EIGRP routing process for autonomous system 100.

Example 4-3. Configuring Redistribution into OSPF

```
RtrA(config)#router ospf 1
RtrA(config-router)#redistribute eigrp ?

 <1-65535> Autonomous system number
RtrA(config-router)#redistribute eigrp 100 ?

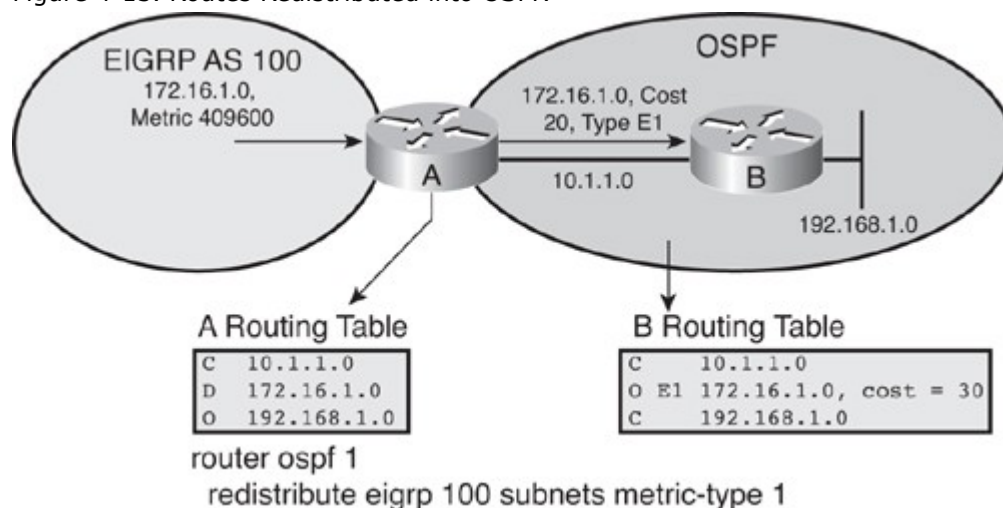
 metric Metric for redistributed routes
 metric-type OSPF/IS-IS exterior metric type for redistributed routes
 route-map Route map reference
 subnets Consider subnets for redistribution into OSPF
 tag Set tag for routes redistributed into OSPF
...
 <cr>
```

It is important to remember that when redistributing into OSPF, the default metric is usually 20, the default metric type is 2, and subnets are not redistributed by default.

Redistribution into OSPF can also be limited to a defined number of prefixes with the **redistribute maximum-prefix** *maximum* [*threshold*] [**warning-only**] router configuration command. The threshold parameter will default to logging a warning at 75 percent of the defined maximum value configured. After reaching the defined maximum number, no further routes are redistributed. If the **warning-only** parameter is configured, no limitation is placed on redistribution, and the maximum value number simply becomes a second point where another warning message is logged. This command was introduced in Cisco IOS Release 12.0(25)S and was integrated into Cisco IOS Releases 12.2(18)S and 12.3(4)T and later.

Figure 4-13 illustrates an example of redistributing EIGRP routes into OSPF. In this example, the default metric of 20 for OSPF is being used. The metric type is set to 1 (type 1 external [E1] routes), meaning that the metric increments whenever updates are passed through the network. Assuming the cost of the Ethernet link is 10, Router B's cost for the 172.16.1.0 route is  $20 + 10 = 30$ .

Figure 4-13. Routes Redistributed into OSPF.



In Figure 4-13, note that the `subnets` keyword is used so subnets are redistributed. If this keyword were omitted, no subnets would be redistributed into the OSPF domain (including the 172.16.1.0 subnet). Omitting this keyword is a common configuration error.

#### Redistributing into EIGRP

Use the `redistribute` protocol [process-id] [match route-type] [metric metric-value] [route-map map-tag] router configuration command to redistribute routes into EIGRP. The parameters for this command are explained in Table 4-5.

Table 4-5. `redistribute` Command for EIGRP

| Parameter    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| protocol     | The source protocol from which routes are redistributed. It can be one of the following keywords: <b>bgp</b> , <b>connected</b> , <b>eigrp</b> , <b>isis</b> , <b>iso-igrp</b> , <b>mobile</b> , <b>odr</b> , <b>ospf</b> , <b>rip</b> , or <b>static</b> .                                                                                                                                                                                                                                                                                                                                                                                |
| process-id   | For BGP or EIGRP, this value is an autonomous system number. For OSPF, this value is an OSPF process ID. This parameter is not required for RIP or IS-IS.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| route-type   | (Optional) A parameter used when redistributing OSPF routes into another routing protocol. It is the criterion by which OSPF routes are redistributed into other routing domains. It can be one of the following:<br><b>internal</b> —Redistributes routes that are internal to a specific autonomous system<br><b>external 1</b> —Redistributes routes that are external to the autonomous system but are imported into OSPF as a type 1 external route<br><b>external 2</b> —Redistributes routes that are external to the autonomous system but are imported into OSPF as a type 2 external route                                       |
| metric-value | (Optional) A parameter that specifies the EIGRP seed metric, in the order of bandwidth, delay, reliability, load, and maximum transmission unit (MTU), for the redistributed route. When redistributing into EIGRP (and all protocols other than OSPF and BGP), if this value is not specified and no value is specified using the <b>default-metric</b> router configuration command, the default metric is 0. For EIGRP (and all protocols other than IS-IS), the default metric of 0 is interpreted as infinity, and routes will not be redistributed. The metric for EIGRP is calculated based only on bandwidth and delay by default. |

**Note:** When redistributing routes from another routing protocol into

| Parameter | Description                                                                                                                                                                                                                                                        |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|           | EIGRP, the default metric is 0, which is interpreted infinity. When redistributing a static or connected route into EIGRP, the default metric is equal to the metric of the associated static or connected interface, so the metric does not have to be specified. |
| map-tag   | (Optional) Specifies the identifier of a configured route map that is interrogated to filter the importation of routes from the source routing protocol to the current EIGRP routing protocol.                                                                     |

Example 4-4 shows how to configure redistribution from OSPF into EIGRP autonomous system 100. This example uses the `router eigrp 100` command to access the routing process into which routes need to be redistributed—in this case, the EIGRP routing process for autonomous system 100. The `redistribute` command is then used to specify the routing protocol to be redistributed into EIGRP autonomous system 100—in this case, OSPF routing process 1.

Example 4-4. Configuring Redistribution into EIGRP

```
RtrA(config)#router eigrp 100
RtrA(config-router)#redistribute ospf ?

<1-65535> Process ID
RtrA(config-router)#redistribute ospf 1 ?

match Redistribution of OSPF routes
metric Metric for redistributed routes
route-map Route map reference
...
<cr>
```

Table 4-6 shows the five parameters that comprise metric-value when redistributing into EIGRP. Alternatively, these same five parameters can be configured using the `default-metric` router configuration command for EIGRP.

Table 4-6. *metric-value* Parameters for EIGRP

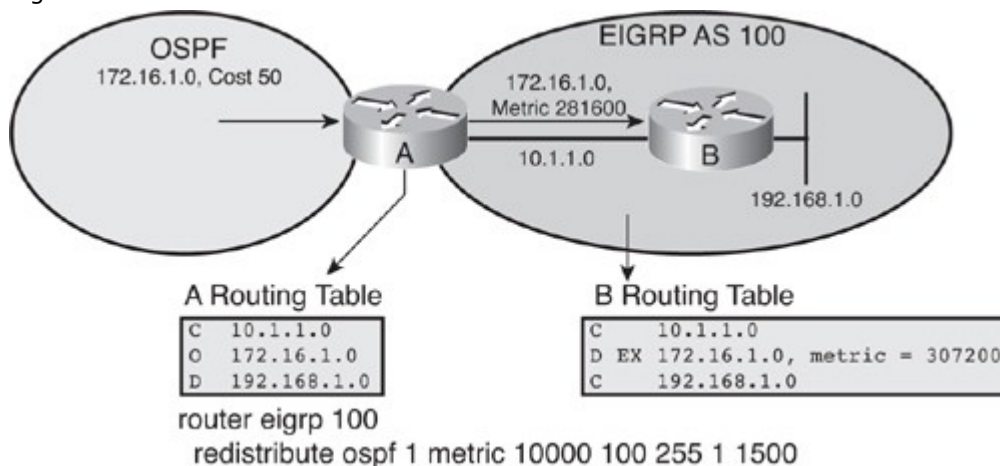
| metric-value<br>Parameter | Description                                                                                                                                                            |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bandwidth                 | The route's minimum bandwidth in kilobits per second (kbps). It can be 0 or any positive integer.                                                                      |
| delay                     | Route delay in tens of microseconds. It can be 0 or any positive integer that is a multiple of 39.1 nanoseconds.                                                       |
| reliability               | The likelihood of successful packet transmission, expressed as a number from 0 to 255, where 255 means that the route is 100 percent reliable, and 0 means unreliable. |
| loading                   | The route's effective loading, expressed as a number from 1 to 255, where 255 means that the route is 100 percent loaded.                                              |
| mtu                       | Maximum transmission unit. The maximum packet size in bytes along the route; an integer greater than or equal to 1.                                                    |

#### Note

Recall from Chapter 2 that MTU is included in the EIGRP update but is actually not used in the metric calculation.

Figure 4-14 illustrates an example of redistributing OSPF routes into EIGRP autonomous system 100. A metric is specified to ensure that routes are redistributed. The redistributed routes appear in Router B's table as external EIGRP (D EX) routes.

Figure 4-14. Routes Redistributed into EIGRP.



External EIGRP routes have a higher administrative distance than internal EIGRP (D) routes, so internal EIGRP routes are preferred over external EIGRP routes. Having different administrative distances for internal and external routes helps prevent routing loops in a network with multiple redistribution points because the routes learned within one part of the network will by default be preferred over those redistributed from another part of the network.

The metric configured in this example is interpreted as follows:

- Bandwidth in kbps = 10,000
- Delay in tens of microseconds = 100
- Reliability = 255 (maximum)
- Load = 1 (minimum)
- MTU = 1500 bytes

The default-metric Command

You can affect how routes are redistributed by changing the default metric associated with a protocol. You either specify the default metric with the **default-metric** router configuration command or use the **metric** *metric-value* parameter in the **redistribute** command.

If you use the **default-metric** command, the default metric you specify applies to all protocols being redistributed into this protocol.

If you use the **metric** parameter in the **redistribute** command, you can set a different default metric for each protocol being redistributed. A metric configured in a **redistribute** command overrides the value in the **default-metric** command for that one protocol.

When redistributing into EIGRP, use the default-metric bandwidth delay reliability loading mtu router configuration command to set the seed metric for all protocols. The parameters of this command are the same as those described earlier in Table 4-6.

When redistributing into OSPF, RIP, and BGP, use the **default-metric** *number* router configuration command for setting the seed metric. The *number* is the value of the metric, such as the number of hops for RIP.

The passive-interface Command

There are times when you must include a subnet in a routing protocols' network command, although you do not want the interface on which the subnet is connected to participate in the routing protocol. For example, you may want to advertise the subnet on the interface to other neighbors, but not run the routing protocol on that interface. The passive-interface type number [default] router configuration command prevents a routing protocol's routing updates from being sent through the specified router interface. This command is used to set either a particular interface or all router interfaces to passive; use the default option to set all router interfaces to passive. Table 4-7 describes the parameters of this command.

Table 4-7. *passive-interface* Command

| Parameter   | Description                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| type number | Specifies the type of interface and interface number that will not send routing updates (or establish neighbor relationships for link-state routing protocols and EIGRP) |
| default     | (Optional) A parameter that sets all interfaces on the router as passive by default                                                                                      |

When you use the **passive-interface** command with RIP (and IGRP), routing updates are not sent out of the specified interface. However, the router still receives routing updates on that interface.

As described in Chapter 2, when you use the *passive-interface* command with EIGRP, hello messages are not sent out of the specified interface. Neighboring router relationships do not form with other routers that can be reached through that interface (because the hello protocol is used to verify bidirectional communication between routers). Because no neighbors are found on an interface, no other EIGRP traffic is sent.

As described in Chapter 3, using the *passive-interface* command on a router running a link-state routing protocol also prevents the router from establishing neighboring router adjacencies with other routers connected to the interface specified in the command. The router does not send hello packets on the interface and therefore cannot establish neighbor adjacencies. Also with OSPF, the specified interface appears as a stub network in the OSPF domain, and OSPF routing information is neither sent nor received through the specified router interface.

#### Note

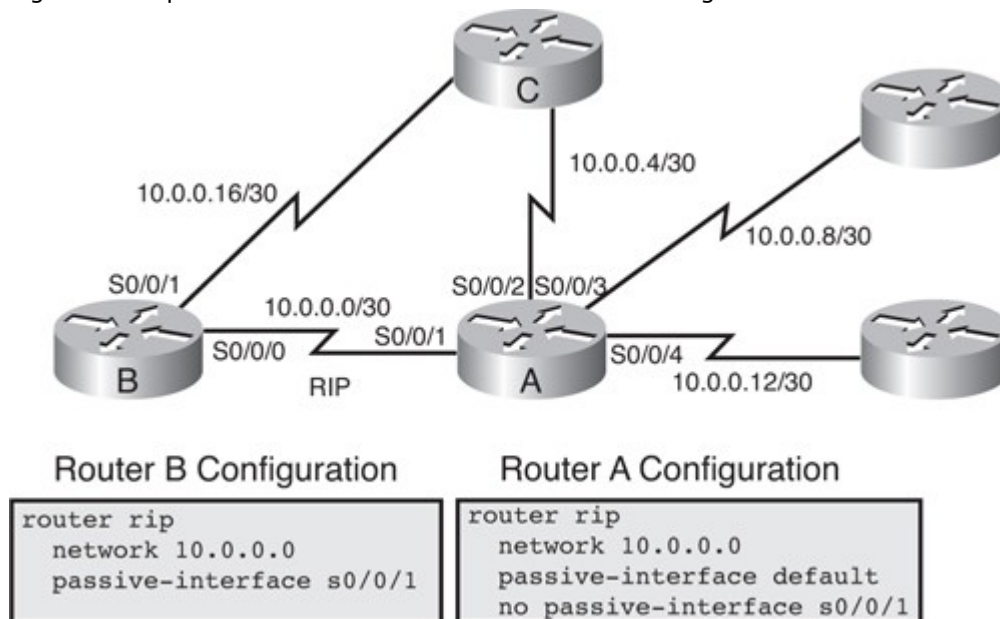
During testing with **debug** commands, it was found that in some Cisco IOS versions OSPF sends hello and database description (DBD) packets on passive interfaces but does not send link-state updates (LSUs). EIGRP does not send anything on passive interfaces.

In ISPs and large enterprise networks, many distribution routers have more than 100 interfaces. Before the introduction of the **passive-interface default** command in Cisco IOS Software Release 12.0, network administrators would have to configure the routing protocol on all interfaces and then manually set the **passive-interface** command on the interfaces where they did not require adjacencies. However, this solution meant entering many **passive-interface** commands. A single **passive-interface default** command can now be used to set all interfaces to passive by default. To enable routing on individual interfaces where you require adjacencies, use the **no passive-interface** command.

For example, in Figure 4-15, Routers A and B run RIP and have a network command that encompasses all their interfaces. However, the network administrator wants to run RIP only on the link between Router A and Router B. Router A has several interfaces, so the *passive-interface default* command is configured, and then the *no passive-interface* command is used for the one interface from where RIP updates are advertised. Router B has only two interfaces, so the *passive-interface* command is used for the one interface that does not participate in RIP routing.



Figure 4-15. passive-interface Command Restricts Routing Traffic on an Interface.



It is important to understand how this configuration affects the information exchanged between Routers A, B, and C. Unless you configure another routing protocol between Routers A and B and Router C, and redistribute between it and RIP, Router A does not tell Router C about the networks it learned from Router B via RIP (or about any of Router A's directly connected networks). Likewise, Router B does not tell Router C that it has a way to reach the networks advertised by Router A via RIP (or about any of Router B's directly connected networks). Physical redundancy is built in to this network; however, the three routers might not be able to use the redundancy effectively if they are not configured properly. For example, if the link between Routers C and A fails, Router C does not know that it has an alternative route through Router B to reach Router A.

#### Route Redistribution Example

This section shows an example of route redistribution in a network using multiple routing protocols.

Figure 4-16 shows the network of a hypothetical company. The network begins with two routing domains (autonomous systems)—one using OSPF and one using RIPv2. Router B is the boundary router; it connects directly to one router within each routing domain and runs both protocols. Router A is in the RIPv2 domain and advertises subnets 10.1.0.0, 10.2.0.0, and 10.3.0.0 to Router B. Router C is in the OSPF domain and advertises subnets 10.8.0.0, 10.9.0.0, 10.10.0.0, and 10.11.0.0 to Router B.

Figure 4-16. Sample Network Before Redistribution.

[View full size image]

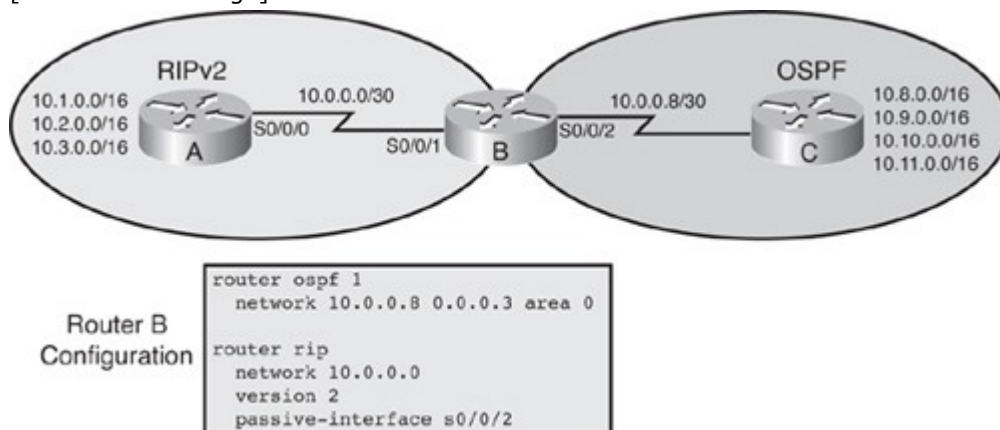


Figure 4-16 also shows the configuration of Router B, before redistribution is configured. RIPv2 is required to run on the serial 0/0/1 interface only, so the passive-interface command is configured for interface serial

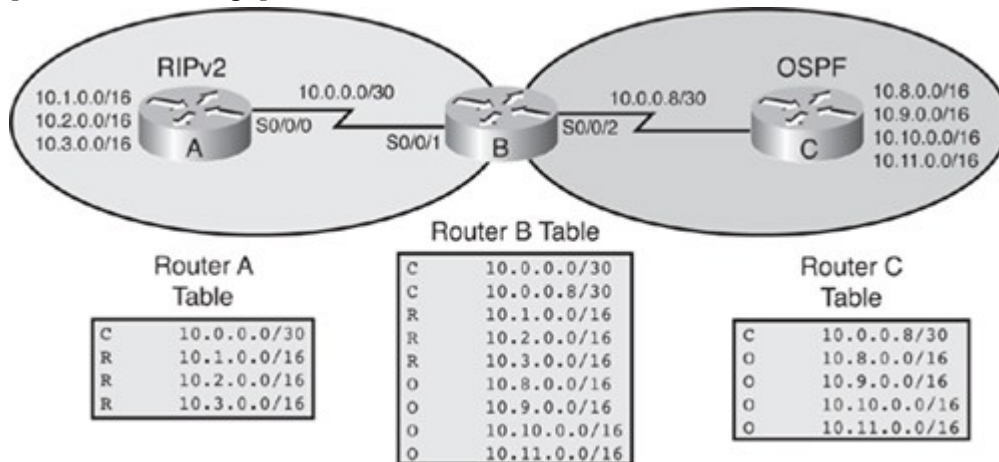


0/0/2 to prevent RIPv2 from sending route advertisements out that interface. OSPF is configured on interface serial 0/0/2.

Figure 4-17 shows the routing tables for Routers A, B, and C. Each routing domain is separate, and routers within them only recognize routes communicated from their own routing protocols. The only router with information on all the routes is Router B, the boundary router that runs both routing protocols and connects to both routing domains.

Figure 4-17. Routing Tables Before Redistribution.

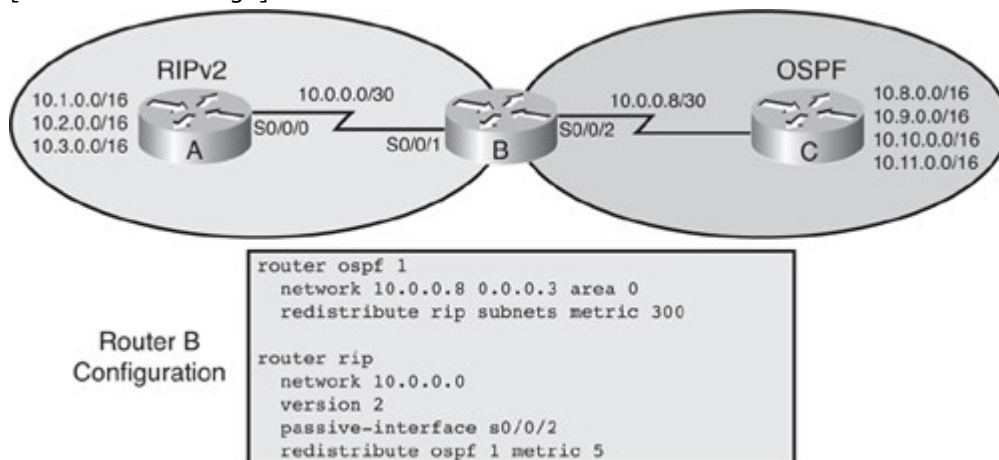
[View full size image]



The goal of redistribution in this network is for all routers to recognize all routes within the company. To accomplish this goal, RIPv2 routes are redistributed into OSPF, and OSPF routes are redistributed into RIPv2. Router B is the boundary router, so the redistribution is configured on it, as shown in Figure 4-18.

Figure 4-18. Redistribution Configured on Router B.

[View full size image]



RIPv2 is redistributed into the OSPF process, and the metric is set using the **redistribute** command. A metric value of 300 is selected because it is a worse metric than any belonging to a native OSPF route. (Although a smaller metric would only be a problem if there were multiple redistribution points between the two domains, it is considered a best practice to configure a larger metric than the native metric of the receiving domain.)

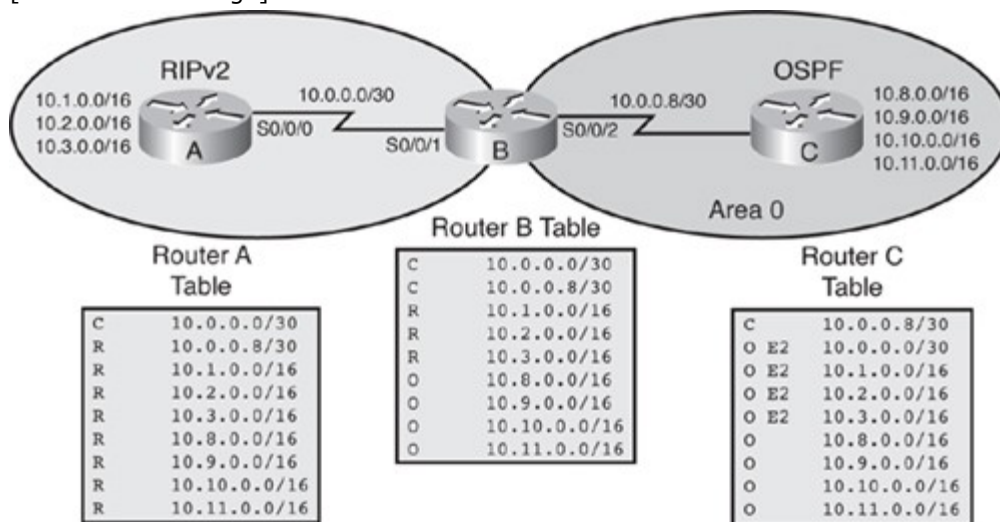
Routes from OSPF process 1 are redistributed into the RIPv2 process with a metric of 5. Again, a value of 5 is chosen because it is higher than any metric in the RIP network.

Figure 4-19 shows the routing tables of all three routers after redistribution is complete; Routers A and C now have routes to all the subnets that Router B learned from the other routing protocol. There is complete

reachability, but Routers A and C now have many more routes to keep track of than before. They also will be affected by any topology changes in the other routing domain.

Figure 4-19. Routing Tables After Redistribution.

[View full size image]



Depending on the network requirements, you can increase efficiency by summarizing the routes before redistributing them. Remember that route summarization hides information, so if routers in the other autonomous systems are required to track topology changes within the entire network, route summarization should not be performed. A more typical case is that the routers need to recognize topology changes only within their own routing domains, so performing route summarization is appropriate.

#### Note

Remember from Chapter 1 that you must be careful when configuring route summarization. If a summarized route indicates that certain subnets can be reached via a router, when in fact those subnets are discontinuous or unreachable via that router, you might experience reachability problems.

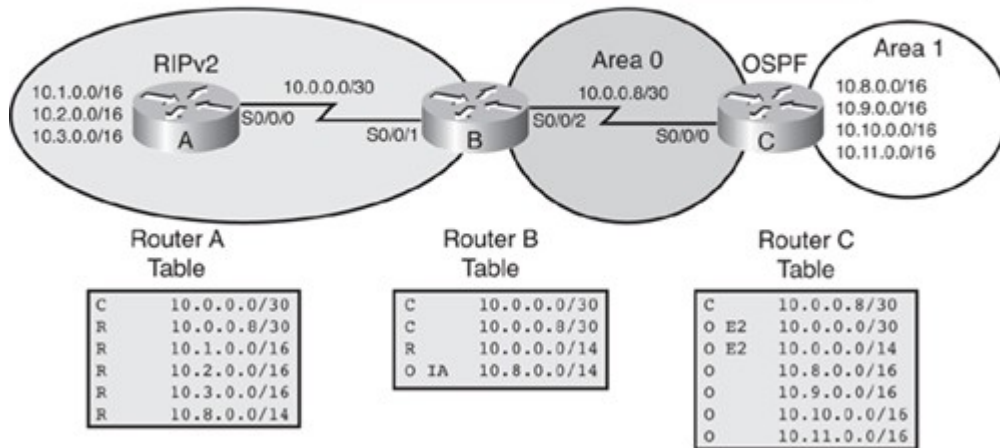
If routes are summarized before redistribution on Routers A and C, each router's routing tables are significantly smaller. Figure 4-20 shows the summarization configuration on both Routers A and C, and the routing tables on all three routers after summarization has been configured. Router B benefits the most; it now has only four routes to keep track of instead of nine. Router A has six routes instead of nine, and Router C has seven routes to keep track of instead of nine. The configurations on Routers A and C are also shown in Figure 4-20.

Figure 4-20. Routing Tables After Summarization.

[View full size image]

```
RouterA (config) #interface s0/0/0
RouterA (config-if) #ip summary-address rip 10.0.0.0 255.252.0.0
```

```
RouterC (config) #router ospf 1
RouterC (config-router) #area 1 range 10.8.0.0 255.252.0.0
```



For RIPv2 on Router A, the summarization command is configured on the interface connecting to Router B, interface S0/0/0. Interface S0/0/0 advertises the summary address instead of the individual subnets. (Note that when RIPv2 is configured, the subnet mask of the summary address must be greater than or equal to the default mask for the major classful network.) 10.0.0.0 255.252.0.0 summarizes the four subnets on Router A (including the 10.0.0.0/30 subnet).

For OSPF, summarization must be configured on an Area Border Router (ABR) or an ASBR. Therefore, OSPF area 1 is created to include the four subnets to be summarized. Router C becomes an ABR, and the summarization command is configured under the OSPF process on Router C. 10.8.0.0 255.252.0.0 summarizes the four subnets on Router C.

#### Using Administrative Distance to Influence the Route-Selection Process

Multiple sources of routing information may be active at the same time, including static routes and routing protocols that use various methods of operation and metrics. In this situation, several sources of information may supply ambiguous next-hop addresses for a particular network, so routers must identify which routing information source is the most trustworthy and reliable. When routes are redistributed between two different methods of resolving the best path, important information may be lost—namely, the relative metrics of the routes—so route selection is sometimes confusing. One approach for correcting wayward choices is to control the administrative distance to indicate route selection preference and ensure that route selection is unambiguous. This approach does not always guarantee the *best* route is selected, only that route selection will be consistent.

You should change the default administrative distance carefully and by considering the network's specific requirements.

#### Selecting Routes with Administrative Distance

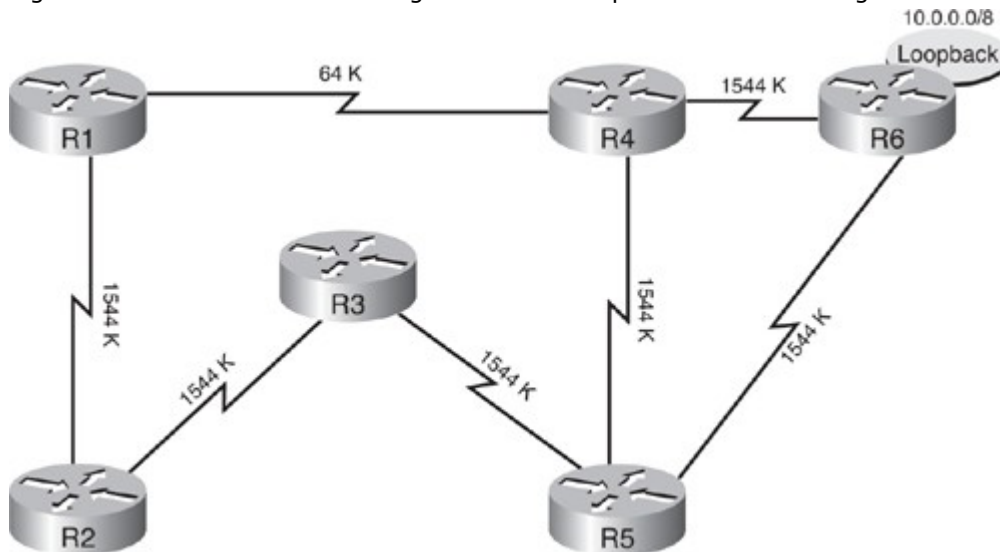
Recall that administrative distance is a way of ranking the believability or trustworthiness of the sources of routing information. Administrative distance is expressed as an integer from 0 to 255, as shown earlier in Table 4-1; lower values indicate greater believability.

Each routing protocol is prioritized in the order of most believable to least believable. Some examples of default prioritization are as follows:

- Prefer manually configured routes (static routes) to dynamically learned routes
- Prefer protocols with sophisticated metrics to protocols with more deterministic metrics
- Prefer external BGP (EBGP) to most other dynamic protocols

For example, in Figure 4-21, R1 may choose a different path to get to the 10.0.0.0/8 network on R6, depending on the routing protocol configured. If RIP, OSPF, and EIGRP are all configured on all routers in this network, the protocols make the following path decisions:

Figure 4-21. Path Selected Through a Network Depends on the Routing Protocols Configured.



- RIP, with an administrative distance of 120, chooses the R1 to R4 to R6 path based on hop count (two hops versus four hops the other way).
- OSPF, with an administrative distance of 110, by default calculates the default metric as 100 megabits per second (Mbps) divided by the interface bandwidth, where the interface bandwidth is the speed of each link in Mbps. The path R1 to R4 to R6 default metric is as follows:

$$\begin{aligned}
 & (100 \text{ Mbps} / 64 \text{ kbps}) + (100 \text{ Mbps} / 1.544 \text{ Mbps}) \\
 &= (100 \text{ Mbps} / .064 \text{ Mbps}) + (100 \text{ Mbps} / 1.544 \text{ Mbps}) \\
 &= (1562 + 64) \\
 &= 1626
 \end{aligned}$$

The R1 to R2 to R3 to R5 to R6 path default metric is this:

$$\begin{aligned}
 & (100 \text{ Mbps} / 1.544 \text{ Mbps}) + (100 \text{ Mbps} / 1.544 \text{ Mbps}) + (100 \text{ Mbps} / 1.544 \text{ Mbps}) + (100 \text{ Mbps} / 1.544 \text{ Mbps}) \\
 &= 64 + 64 + 64 + 64 \\
 &= 256
 \end{aligned}$$

Therefore, OSPF chooses the R1 to R2 to R3 to R5 to R6 path. (Although OSPF and IS-IS are both link-state routing protocols that converge quickly, OSPF is considered more trustworthy than IS-IS because OSPF bases its default metric on bandwidth and therefore is more likely to pick a faster path.)

#### Note

By default, the Cisco routers calculate the OSPF cost using the formula  $100 \text{ Mbps} / \text{bandwidth}$ . However, this formula is based on a maximum bandwidth of 100 Mbps, resulting in a cost of 1. If you have faster interfaces, you might want to recalibrate the cost of 1 to a higher bandwidth. Chapter 3 provides details about OSPF cost calculations.

- EIGRP, with an administrative distance of 90, calculates the default metric as  $\text{BW} + \text{delay}$ , where BW is  $[(10^7 / \text{least bandwidth in the path in kbps}) * 256]$ , and delay is cumulative across the path, in tens of microseconds, multiplied by 256. Assuming a uniform link delay of 100 tens of microseconds, the R1 to R4 to R6 path default metric is as follows:

$$((10^7 / 64) * 256) + (200 * 256) = 40,051,200.$$

The R1 to R2 to R3 to R5 to R6 path default metric is this:

$$((10^7 / 1544) * 256) + (400 * 256) = 1,760,431.$$

- Therefore, EIGRP chooses the R1 to R2 to R3 to R5 to R6 path. Although EIGRP and OSPF routing protocols both converge quickly and consider bandwidth, EIGRP is considered more trustworthy than OSPF because EIGRP takes more information into account in its calculation.

Because EIGRP has the lowest administrative distance of the three protocols, only the EIGRP path to 10.0.0.0/8 is put into the routing table.

#### Note

The administrative distance affects only the choice of path for identical IP routes—in other words, for routes with the same prefix and mask. For example, because OSPF does not summarize by default, and all the other protocols do, the protocols might potentially provide different routing information. In this example, if OSPF advertised a route to 10.1.0.0/16 that was not advertised by any of the other protocols (because they automatically summarized to 10.0.0.0/8), the 10.1.0.0/16 route would be in the routing tables from OSPF, and the 10.0.0.0/8 route would be in the routing tables from EIGRP. As mentioned in Chapter 1, routers use the longest prefix match in the routing table if more than one entry in the routing table matches a particular destination. In this example, packets for 10.1.1.2 would be sent via the OSPF-learned route, whereas packets for 10.2.1.3 would be sent via the EIGRP-learned route.

Typically, multiple routing protocols are run only on the boundary routers in a network, not on *all* routers, so this situation should not be common.

#### Modifying Administrative Distance

In some cases, you will find that a router selects a suboptimal path as a result of believing a routing protocol that actually has a poorer route, because that protocol has a better administrative distance. One way to make sure that routes from the desired routing protocol are selected is to assign a higher administrative distance to the routes from the undesired routing protocol.

#### Note

Routes with a distance of 255 are not installed in the routing table.

For all protocols use the distance administrative-distance [address wildcard-mask [ip-standard- list] [ip-extended-list]] router configuration command, as explained in Table 4-8, to change the default administrative distances.

Table 4-8. *distance* Command

| Parameter                                       | Description                                                                                                                                                                                                                                                                                                                             |
|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| administrative-distance                         | Sets the administrative distance, an integer from 10 to 255. (The values 0 to 9 are reserved for internal use and should not be used, even though values from 1 to 9 can be configured.)                                                                                                                                                |
| address                                         | (Optional) Specifies the IP address. This allows filtering of networks according to the IP address of the router supplying the routing information.                                                                                                                                                                                     |
| wildcard-mask                                   | (Optional) Specifies the wildcard mask used to interpret the IP address. A bit set to 1 in the <i>wildcard-mask</i> argument instructs the software to ignore the corresponding bit in the address value. Use an address/mask of 0.0.0.0 255.255.255.255 to match any IP address (any source router supplying the routing information). |
| <i>ip-standard-list</i> <i>ip-extended-list</i> | (Optional) The number or name of a standard or extended access list to be applied to the incoming routing updates. Allows filtering of the networks being advertised.                                                                                                                                                                   |

#### Note

The *ip-standard-list* and *ip-extended-list* parameters were added in Cisco IOS Release 12.0.

Alternatively, for EIGRP, the `distance eigrp internal-distance external-distance` router configuration command can be used, as explained in Table 4-9. By default, natively learned routes have an administrative distance of 90, but external routes have an administrative distance of 170.

Table 4-9. *distance eigrp* Command

| Parameter         | Description                                                                                                                                                                                                                                           |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| internal-distance | Specifies the administrative distance for EIGRP internal routes. Internal routes are those that are learned from another entity within the same EIGRP autonomous system. The distance can be a value from 1 to 255. The default is 90.                |
| external-distance | Specifies the administrative distance for EIGRP external routes. External routes are those for which the best path is learned from a neighbor external to the EIGRP autonomous system. The distance can be a value from 1 to 255. The default is 170. |

In Example 4-5, the `distance eigrp 80 130` command sets the administrative distance for internal EIGRP routes to 80 and for external EIGRP routes to 130. The `distance 90 192.168.7.0 0.0.0.255` command sets the administrative distance to 90 for all routes learned from routers on the Class C network 192.168.7.0. The `distance 120 172.16.1.3 0.0.0.0` command sets the administrative distance to 120 for all routes from the router with the address 172.16.1.3.

Example 4-5. Modifying Administrative Distance for EIGRP Example

```
router eigrp 100
 network 192.168.7.0
 network 172.16.0.0
 distance eigrp 80 130
 distance 90 192.168.7.0 0.0.0.255
 distance 120 172.16.1.3 0.0.0.0
```

Alternatively for OSPF, the `distance ospf {[intra-area dist1] [inter-area dist2] [external dist3]}` router configuration command can be used to define the OSPF administrative distances based on route type, as explained in Table 4-10.

Table 4-10. *distance ospf* Command

| Parameter | Description                                                                                                                                                                     |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dist1     | (Optional) Specifies the administrative distance for all OSPF routes within an area. Acceptable values are from 1 to 255. The default is 110.                                   |
| dist2     | (Optional) Specifies the administrative distance for all OSPF routes from one area to another area. Acceptable values are from 1 to 255. The default is 110.                    |
| dist3     | (Optional) Specifies the administrative distance for all routes from other routing domains, learned by redistribution. Acceptable values are from 1 to 255. The default is 110. |

In Example 4-6, the `distance ospf external 100 inter-area 100 intra-area 100` command sets the administrative distance for external, interarea, and intra-area OSPF routes to 100 (default values are 110). The `distance 90 10.0.0.0 0.0.0.255`, `distance 110 10.11.0.0 0.0.0.255`, and `distance 130 10.11.12.0 0.0.0.255` commands set the administrative distance to 90, 110, and 130, respectively, for all routes learned from routers with specific addresses; notice that the router's addresses are specified from least to the most specific. For example, routes from a router with address 10.10.0.1 will have an administrative distance of 90, and routes from a router with address 10.11.12.1 will have an administrative distance of 130.



#### Example 4-6. Modifying Administrative Distance for OSPF Example

```
router ospf 10
 network 192.168.7.0 0.0.0.255 area 0
 network 172.16.0.0 0.0.255.255 area 0
 distance ospf external 100 inter-area 100 intra-area 100
 distance 90 10.0.0.0 0.0.0.255
 distance 110 10.11.0.0 0.0.0.255
 distance 130 10.11.12.0 0.0.0.255
```

Alternatively, for BGP, the distance bgp external-distance internal-distance local-distance router configuration command can be used to change the administrative distances, as explained in Table 4-11. (Using the distance command for BGP sets the administrative distance of EBGp routes only.)

Table 4-11. *distance bgp* Command

| Parameter         | Description                                                                                                                                                                                                                                                                                                          |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| external-distance | Specifies the administrative distance for BGP external routes. External routes are routes for which the best path is learned from a neighbor external to the autonomous system. Acceptable values are from 1 to 255. The default is 20.                                                                              |
| internal-distance | Specifies the administrative distance for BGP internal routes. Internal routes are learned from another BGP entity within the same autonomous system. Acceptable values are from 1 to 255. The default is 200.                                                                                                       |
| local-distance    | Specifies the administrative distance for BGP local routes. Local routes are networks that are listed with a <b>network</b> router configuration command, often as back doors, for that router or for networks that are redistributed from another process. Acceptable values are from 1 to 255. The default is 200. |

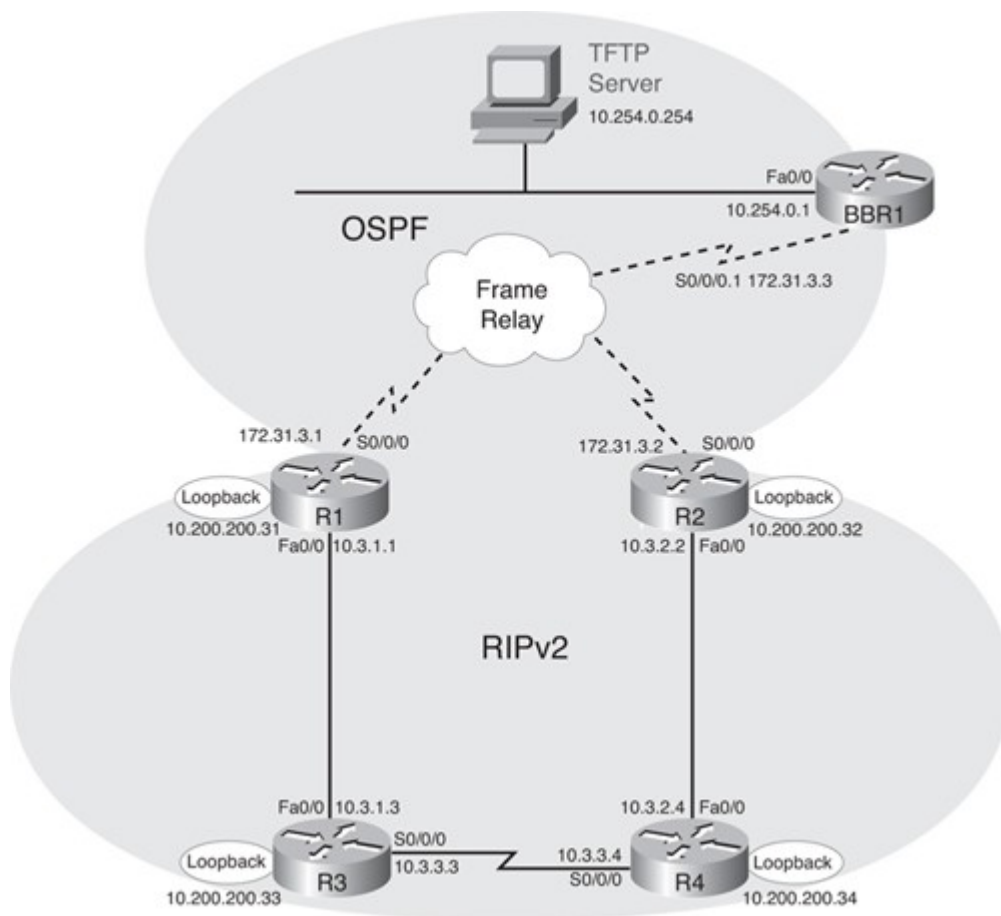
#### Redistribution Using Administrative Distance Example

Figure 4-22 illustrates a network using multiple routing protocols, RIPv2 and OSPF. The purpose of this example is to show how a path-selection problem can occur, where the problem appears, and one possible way to resolve it, using administrative distance.

Figure 4-22. Sample Redistribution Network Topology.

[View full size image]





Recall that OSPF is by default considered more believable than RIPv2; OSPF has an administrative distance of 110 and RIPv2 has an administrative distance of 120. For example, if a boundary router (R1 or R2) learns about network 10.3.3.0 via RIPv2 and also via OSPF, the OSPF route is inserted into the routing table. This route is used because OSPF has a lower administrative distance than RIPv2, even though the path via OSPF might be the longer (worse) path.

Example 4-7 and Example 4-8 illustrate the configurations for the R1 and R2 routers; RIPv2 is redistributed into OSPF and OSPF is redistributed into RIPv2 on both routers.

Example 4-7. Configuration of Redistribution on Router R1 in Figure 4-22

```
hostname R1
!
router ospf 1
 redistribute rip metric 10000 metric-type 1 subnets
 network 172.31.0.0 0.0.255.255 area 0
!
router rip
 version 2
 redistribute ospf 1 metric 5
 network 10.0.0.0
 no auto-summary
```

Example 4-8. Configuration of Redistribution on Router R2 in Figure 4-22

```
hostname R2
!
router ospf 1
```

```
redistribute rip metric 10000 metric-type 1 subnets
network 172.31.0.0 0.0.255.255 area 0
!
router rip
version 2
redistribute ospf 1 metric 5
network 10.0.0.0
no auto-summary
```

The RIPv2 routes redistributed into OSPF have an OSPF seed metric of 10,000 to make these routes less preferred than native OSPF routes and to protect against route feedback. The **redistribute** command also sets the metric type to 1 (external type 1) so that the route metrics continue to accrue. The routers also redistribute subnet information.

The OSPF routes redistributed into RIPv2 have a RIP seed metric of five hops to also protect against route feedback.

The first edge router on which redistribution is configured, R1 in this case, has a routing table that contains both OSPF and RIPv2 routes, as you would expect. The routers in the OSPF domain learn about the routes from the RIPv2 domain via redistribution; they then advertise these RIPv2 routes via OSPF routes to their neighboring routers. Thus, the second edge router, R2, receives information about the RIPv2 domain routes (also called the native RIPv2 routes) from both OSPF and RIPv2. R2 prefers the OSPF routes because OSPF has a lower administrative distance.

Example 4-9 displays the routing table on the R2 router after redistribution has occurred. This output confirms that even though the R2 router learns RIPv2 and OSPF routes, it lists only OSPF routes in the routing table, because they have a lower administrative distance.

Example 4-9. Routing Table on Router R2 in Figure 4-22 with Redistribution Configured

```
R2#show ip route
<output omitted>
Gateway of last resort is not set

 172.31.0.0/24 is subnetted, 1 subnet
C 172.31.3.0/24 is directly connected, Serial0/0/0
 10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
O E1 10.3.1.0/24 [110/10781] via 172.31.3.1, 00:09:47, Serial0/0/0
O E1 10.3.3.0/24 [110/10781] via 172.31.3.1, 00:04:51, Serial0/0/0
C 10.3.2.0/24 is directly connected, FastEthernet0/0
O E1 10.200.200.31/32 [110/10781] via 172.31.3.1, 00:09:48, Serial0/0/0
O E1 10.200.200.34/32 [110/10781] via 172.31.3.1, 00:04:52, Serial0/0/0
C 10.200.200.32/32 is directly connected, Loopback0
O E1 10.200.200.33/32 [110/10781] via 172.31.3.1, 00:04:52, Serial0/0/0
O E2 10.254.0.0/24 [110/50] via 172.31.3.3, 00:09:48, Serial0/0/0
R2#
```

See Figure 4-22 to trace some of the routes. The redistribution has resulted in suboptimal paths to many of the networks. For instance, 10.200.200.34 is a loopback interface on Router R4. R4 is directly attached to R2. However, from R2 the OSPF path to that loopback interface goes through R1, and then R3, and then R4 before it reaches its destination. The OSPF path taken is actually a longer (worse) path than the more direct RIPv2 path.

You can change the administrative distance of the redistributed RIPv2 routes to ensure that the boundary routers select the native RIPv2 routes. Example 4-10 and Example 4-11 show the configurations on the R1 and R2 routers. The distance command changes the administrative distance of the OSPF routes to the networks that match access list 64 to 125 (from 110). Table 4-12 describes some of the command parameters used in the example configurations.

Table 4-12. distance Command Parameters Used in Example 4-10 and Example 4-11

| Parameter                  | Description                                                                                                                              |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| 125                        | Defines the administrative distance that specified routes are assigned                                                                   |
| 0.0.0.0<br>255.255.255.255 | Defines the source address of the router supplying the routing information—in this case, any router                                      |
| 64                         | Defines the access list to be used to filter incoming routing updates to determine which will have their administrative distance changed |

Example 4-10. Configuration to Change the Administrative Distance on Router R1 in Figure 4-22

```
hostname R1
!
router ospf 1
redistribute rip metric 10000 metric-type 1 subnets
network 172.31.0.0 0.0.255.255 area 0
distance 125 0.0.0.0 255.255.255.255 64
!
router rip
version 2
redistribute ospf 1 metric 5
network 10.0.0.0
no auto-summary
!
access-list 64 permit 10.3.1.0
access-list 64 permit 10.3.3.0
access-list 64 permit 10.3.2.0
access-list 64 permit 10.200.200.31
access-list 64 permit 10.200.200.32
access-list 64 permit 10.200.200.33
access-list 64 permit 10.200.200.34
```

Example 4-11. Configuration to Change the Administrative Distance on Router R2 in Figure 4-22

```
hostname R2
!
router ospf 1
redistribute rip metric 10000 metric-type 1 subnets
network 172.31.0.0 0.0.255.255 area 0
distance 125 0.0.0.0 255.255.255.255 64
!
router rip
version 2
redistribute ospf 1 metric 5
network 10.0.0.0
no auto-summary
!
access-list 64 permit 10.3.1.0
access-list 64 permit 10.3.3.0
access-list 64 permit 10.3.2.0
```

```
access-list 64 permit 10.200.200.31
access-list 64 permit 10.200.200.32
access-list 64 permit 10.200.200.33
access-list 64 permit 10.200.200.34
```

Access list 64 is used to match all the native RIPv2 routes. The access-list 64 permit 10.3.1.0 command configures a standard access list to permit the 10.3.1.0 network; similar access list statements permit the other internal native RIPv2 networks. Table 4-13 describes some of the command parameters used in the examples.

Table 4-13. access-list Command Parameters Used in Example 4-10 and Example 4-11

| Parameter | Description                                                                                                            |
|-----------|------------------------------------------------------------------------------------------------------------------------|
| 64        | The access list number                                                                                                 |
| permit    | Allows all networks that match the address to be permitted—in this case, to have their administrative distance changed |
| 10.3.1.0  | A network to be permitted—in this case, to have its administrative distance changed                                    |

Both R1 and R2 are configured to assign an administrative distance of 125 to routes that they learn from OSPF and that match access list 64. Access list 64 has **permit** statements for the internal native RIPv2 networks 10.3.1.0, 10.3.2.0, and 10.3.3.0 and the loopback networks 10.200.200.31, 10.200.200.32, 10.200.200.33, and 10.200.200.34. Therefore, when either of these routers learns about these networks from both RIPv2 and OSPF, it selects the routes learned from RIPv2—with a lower administrative distance of 120—over the same routes learned from OSPF (via redistribution from the other boundary router)—with an administrative distance of 125—and puts only the RIPv2 routes in the routing tables.

Notice in this example that the **distance** command is part of the OSPF routing process configuration because the administrative distance should be changed for these routes when they are learned by OSPF, not by RIPv2.

You need to configure the **distance** command on both redistributing routers because either one can have suboptimal routes, depending on which redistributing router sends the OSPF updates about the RIPv2 networks to the other redistributing router first.

Example 4-12 shows that Router R2 now retains the more direct paths to the internal networks by learning them from RIPv2.

Example 4-12. Routing Table on Router R2 in Figure 4-22 with the Administrative Distance Changed

```
R2#show ip route
<output omitted>
Gateway of last resort is not set

 172.31.0.0/24 is subnetted, 1 subnet
C 172.31.3.0/24 is directly connected, Serial0/0/0
 10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
R 10.3.1.0/24 [120/2] via 10.3.2.4, 00:00:03, FastEthernet0/0
R 10.3.3.0/24 [120/1] via 10.3.2.4, 00:00:03, FastEthernet0/0
C 10.3.2.0/24 is directly connected, FastEthernet0/0
R 10.200.200.31/32 [120/3] via 10.3.2.4, 00:00:04, FastEthernet0/0
R 10.200.200.34/32 [120/1] via 10.3.2.4, 00:00:04, FastEthernet0/0
C 10.200.200.32/32 is directly connected, Loopback0
R 10.200.200.33/32 [120/2] via 10.3.2.4, 00:00:04, FastEthernet0/0
O E2 10.254.0.0/24 [110/50] via 172.31.3.3, 00:00:04, Serial0/0/0
```

R2#

However, some routing information is lost with this configuration. For example, depending on the actual bandwidths, R2's OSPF path for the 10.3.1.0 network might have been better than its RIP path, so it might have made sense not to include 10.3.1.0 in the access list for R2.

This example illustrates the importance of knowing your network before implementing redistribution, and closely examining which routes the routers select after redistribution is enabled. You should pay particular attention to routers that can select from several possible redundant paths to a network, because they may select suboptimal paths. The most important feature of using administrative distance to control route preference is that no path information is lost. In this example, the OSPF information is still in the OSPF database, so if the primary path (via the RIPv2 routes) is lost, the OSPF path reasserts itself, and the router maintains connectivity with those networks.

#### Verifying Redistribution Operation

The best way to verify redistribution operation is as follows:

- Know your network topology, particularly where redundant routes exist.
- Study the routing tables on a variety of routers in the internetwork using the **show ip route** [*ip-address*] EXEC command. For example, check the routing table on the boundary router and on some of the internal routers in each autonomous system.
- Examine the topology table of each configured routing protocol to ensure that all appropriate prefixes are being learned.
- Perform a trace using the **traceroute** [*ip-address*] EXEC command on some of the routes that go across the autonomous systems to verify that the shortest path is being used for routing. Be sure to run traces to networks for which redundant routes exist.
- If you encounter routing problems, use the **traceroute** and **debug** commands to observe the routing update traffic on the boundary routers and on the internal routers.

#### Caution

Use caution when executing **debug** commands because they may consume a lot of router resources and could cause problems in a busy production network. Debugging output takes priority over other network traffic. Too much debug output might severely reduce the performance of the router or even render it unusable in the worst case.

#### Controlling Routing Update Traffic

Routing updates compete with user data for bandwidth and router resources, yet routing updates are critical because they carry the information that routers need to make sound routing decisions. To ensure that the network operates efficiently, you must control and tune routing updates. Information about networks must be sent where it is needed and filtered from where it is not needed. No one type of route filter is appropriate for every situation. Therefore, the more techniques you have at your disposal, the better your chance of having a smooth, well-run network.

This section discusses controlling the updates sent and received by dynamic routing protocols and controlling the routes redistributed into routing protocols. In many cases, you do not want to prevent all routing information from being advertised; you might want to block the advertisement of only certain routes. For example, you could use such a solution to prevent routing loops when implementing two-way route redistribution with dual redistribution points. The following are some ways to control or prevent dynamic routing updates from being generated:

- **Passive interface**— A passive interface prevents routing updates for the specified protocol from being sent through an interface.
- **Default routes**— A default route instructs the router that if it does not have a route for a given destination, it should send the packet to the default route. Therefore, no dynamic routing updates about the remote destinations are necessary.
- **Static routes**— A static route allows routes to remote destinations to be manually configured on the router. Therefore, no dynamic routing updates about the remote destinations are necessary.
- **Route maps**— Route maps are complex access lists that allow conditions to be tested against a packet or route, and then actions taken to modify attributes of the packet or route.
- **Distribute lists**— A distribute list allows an access list to be applied to routing updates.
- **Prefix lists**— A prefix list is a specialized access list designed to filter routes.

Passive interfaces were discussed earlier in the “The passive-interface Command” section. Static and default routes were discussed in Chapter 1; specifics related to controlling routing updates are explored in the next section, which is followed by a discussion of route maps, distribute lists, and prefix lists.

### Static and Default Routes

Static routes are routes that you manually configure on a router. Static routes are used most often to do the following:

- Define specific routes to use when two autonomous systems must exchange routing information, rather than having entire routing tables exchanged.
- Define routes to destinations over a WAN link to eliminate the need for a dynamic routing protocol—that is, when you do not want routing updates to enable or cross the link.

When configuring static routes, keep in mind the following considerations:

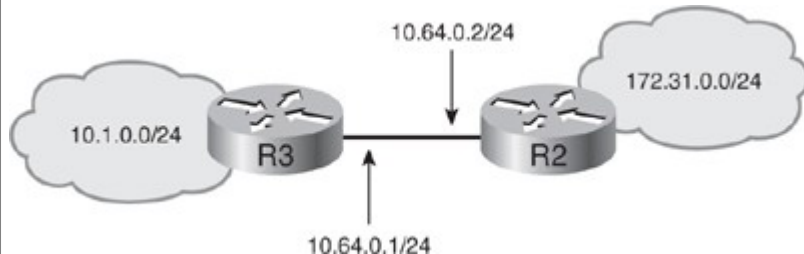
- When using static routes instead of dynamic routing updates, all participating routers must have static routes defined so that they can reach remote networks. Static route entries must be defined for all routes for which a router is responsible. To reduce the number of static route entries, you can define a default static route—for example, **ip route 0.0.0.0 0.0.0.0 S0/0/1**.
- If you want a router to advertise a static route in a routing protocol, you might need to redistribute it.

You can configure default routes for routing protocols on Cisco routers. For example, when you create a default route on a router running RIP, the router advertises an address of 0.0.0.0. When a router receives this default route, it forwards any packets destined for a destination that does not appear in its routing table to the default route you configured.

#### ip default-network and Other IP Commands

You can also configure a default route by using the `ip default-network network-number global` configuration command. Figure 4-23, Example 4-13, and Example 4-14 demonstrate the use of this command on a router running RIP. With the `ip default-network` command, you designate an actual network currently available in the routing table as the default path to use.

Figure 4-23. Using the *ip default-network* Command.



Example 4-13. Configuration on Router R2 in Figure 4-23

```
R2#show run
<output omitted>
router rip
 network 10.0.0.0
 network 172.31.0.0
!
ip classless
ip default-network 10.0.0.0
```

Example 4-14. Routing Table on R3 in Figure 4-23

```
R3#show ip route
<output omitted>
Gateway of last resort is 10.64.0.2 to network 0.0.0.0
<Output Omitted>
R 172.31.0.0/16 [120/1] via 10.64.0.2, 00:00:16, FastEthernet0/0
R* 0.0.0.0/0 [120/1] via 10.64.0.2, 00:00:05, FastEthernet0/0
```

In Example 4-13, the R2 router has a directly connected interface onto the network specified in

the `ip default-network network-number` command. RIP generates (sources) a default route, which appears as a 0.0.0.0 0.0.0.0 route to its RIP neighbor routers, as shown in Example 4-14 for R3.

The **`ip default-network`** command is used to distribute default route information to other routers. For RIP, this command provides no functionality for the router on which it is configured. Other protocols behave differently than RIP with the **`ip default-network`** and **`ip route 0.0.0.0 0.0.0.0`** commands.

For example, EIGRP does not redistribute the 0.0.0.0 0.0.0.0 default route by default. However, if the `network 0.0.0.0` command is added to the EIGRP configuration, it redistributes a default route as a result of the `ip route 0.0.0.0 0.0.0.0 interface` command (but not as a result of the `ip route 0.0.0.0 0.0.0.0 address` or `ip default-network` commands). See Chapter 2 and Cisco IOS documentation for further information.

The **`ip default-network`** command is used in networks with routers that do not know how to get to the outside world. This command is configured on the router that connects to the outside world and goes through a different major network to reach the outside world. If your environment is all one major network address, you probably would not want to use the **`ip default-network`** command, but rather a static route to 0.0.0.0 via a border router.

The **`ip route 0.0.0.0 0.0.0.0`** command is used on routers with IP routing enabled and that point to the outside world, for example for Internet connectivity. This route is advertised as the “gateway of last resort” if running RIP. The router that is directly connected to the border of the outside world is the preferred router, with the static route pointing to 0.0.0.0.

(Note that a command with a similar name, the **`ip default-gateway`** command, is used on routers or communication servers that have IP routing turned off. The router or communication server acts just like a host on the network.)

## Using Route Maps

Route maps provide another technique to manipulate and control routing protocol updates. Route maps may be used for a variety of purposes. After describing route map applications and operation, this section explores the use of route maps as a tool to filter and manipulate routing updates. All the IP routing protocols can use route maps for redistribution filtering.

### Route Map Applications

Network administrators use route maps for a variety of purposes. Several of the more common applications for route maps are as follows:

- **Route filtering during redistribution**— Redistribution nearly always requires some amount of route filtering. Although distribute lists can be used for this purpose, route maps offer the added benefit of manipulating routing metrics through the use of **`set`** commands.
- **Policy-based routing (PBR)**— Route maps can be used to match source and destination addresses, protocol types, and end-user applications. When a match occurs, a **`set`** command can be used to define the interface or next-hop address to which the packet should be sent. PBR allows the operator to define routing policy other than basic destination-based routing using the routing table.
- **Network Address Translation (NAT)**— Route maps can better control which private addresses are translated to public addresses. Using a route map with NAT also provides more detailed **`show`** commands that describe the address-translation process.
- **BGP**— Route maps are the primary tools for implementing BGP policy. Network administrators assign route maps to specific BGP sessions (neighbors) to control which routes are allowed to flow in and out of the BGP process. In addition to filtering, route maps provide sophisticated manipulation of BGP path attributes. Route maps for BGP are discussed in Chapter 6, “Implementing a BGP Solution for ISP Connectivity.”

### Understanding Route Maps

Route maps are complex access lists that allow some conditions to be tested against the packet or route in question using **`match`** commands. If the conditions match, some actions can be taken to modify attributes of the packet or route. These actions are specified by **`set`** commands. This is a big difference between route maps and access lists: Route maps can modify the packet or route by using **`set`** commands.

A collection of **`route-map`** statements that have the same route map name is considered one route map. Within a route map, each **`route-map`** statement is numbered and therefore can be edited individually.



The statements in a route map correspond to the lines of an access list. Specifying the match conditions in a route map is similar in concept to specifying the source and destination addresses and masks in an access list.

The route-map map-tag [permit | deny] [sequence-number] global configuration command can be used to define a route map. The parameters of this command are explained in detail in Table 4-14.

Table 4-14. *route-map* Command

| Parameter                   | Description                                                                                                                                                                                    |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| map-tag                     | Name of the route map.                                                                                                                                                                         |
| <b>permit</b>   <b>deny</b> | (Optional) A parameter that specifies the action to be taken if the route map match conditions are met. The meaning of <b>permit</b> or <b>deny</b> is dependent on how the route map is used. |
| sequence-number             | (Optional) A sequence number that indicates the position that a new <b>route-map</b> statement will have in the list of <b>route-map</b> statements already configured with the same name.     |

The default for the **route-map** command is **permit**, with a *sequence-number* of 10.

If you leave out the sequence number when configuring all statements for the same route map name, the router will assume that you are editing (and possibly adding to) the first statement, which defaults to sequence number 10. It is important to remember that route map sequence numbers do not automatically increment!

When the *sequence-number* parameter of the **route-map** command is not used, the following occurs:

- If no other entry is already defined with the supplied **route-map** tag, an entry is created, with the *sequence-number* set to 10.
- If only one entry is already defined with the supplied **route-map** tag, that entry is the default entry for the **route-map** command, and the *sequence-number* of the entry is unchanged. (The router assumes you are editing the one entry that is already defined.)
- If more than one entry is already defined with the supplied **route-map** tag, an error message is displayed, indicating that the *sequence-number* is required.
- If the **no route-map map-tag** command is specified (without the *sequence-number* parameter), then the whole route map is deleted.

A route map may be made up of multiple **route-map** statements (with different sequence numbers). The statements are processed top-down, similar to an access list. The first match found for a route is applied. The sequence number is also used for inserting or deleting specific **route-map** statements in a specific place in the route map.

The sequence number specifies the order in which conditions are checked. For example, if two statements in a route map are named MYMAP, one with sequence 10 and the other with sequence 20, sequence 10 is checked first. If the match conditions in sequence 10 are not met, sequence 20 is checked.

Like an access list, an implicit **deny any** appears at the end of a route map. The consequences of this **deny** depend on how the route map is being used.

The **match condition** route map configuration commands are used to define the conditions to be checked.

The **set condition** route map configuration commands are used to define the actions to be followed if there is a match and the action to be taken is permit. (The consequences of a deny action depend on how the route map is being used.)

A **route-map** statement without any **match** statements will be considered matched.

A single **match** statement may contain multiple conditions. Only one condition in the same **match** statement must be true for that **match** statement to be considered a match (this is a logical OR operation).

A **route-map** statement may contain multiple **match** statements. All **match** statements within a **route-map** statement must be considered true for the **route-map** statement to be considered matched. (This is a logical AND operation.)

For example, IP standard or extended access lists or prefix lists can be used to establish match criteria using the **match ip address** {*access-list-number* | *name*} [...*access-list-number* | *name*] | **prefix-list** *prefix-list-name* [...*prefix-list-name*] route map configuration command. If multiple access lists or prefix lists are specified, matching any one results in a match. A standard IP access list can be used to specify match

criteria for a packet's source address. Extended access lists can be used to specify match criteria based on source and destination addresses, application, protocol type, type of service (ToS), and precedence. Another way to explain how a route map works is to use a simple example and see how it would be interpreted. Example 4-15 shows a sample route map-like configuration. (Note that on a router, all the conditions and actions shown would be replaced with specific conditions and actions, depending on the exact match and set commands used.)

Example 4-15. Demonstration of the *route-map* Command

```
route-map demo permit 10
 match x y z
 match a
 set b
 set c
route-map demo permit 20
 match q
 set r
route-map demo permit 30
```

The route map named demo in Example 4-15 is interpreted as follows:

```
If {(x or y or z) and (a) match} then {set b and c}
Else
If q matches then set r
Else
Set nothing
```

#### Configuring Route Maps to Control Routing Updates

The redistribute commands discussed in the "Configuring Redistribution" section all have a route-map option with a map-tag parameter. This parameter refers to a route map configured with the route-map map-tag [permit | deny] [sequence-number] global configuration command, as described earlier in Table 4-14.

It is important to understand what the **permit** and **deny** mean when redistributing. When used with a **redistribute** command, a **route-map** statement with **permit** indicates that the matched route is to be redistributed, while a **route-map** statement with **deny** indicates that the matched route is not to be redistributed.

The match condition route map configuration commands are used to define the conditions to be checked. Table 4-15 lists some of the variety of matchcommands that can be configured. Some of these commands are used for BGP policy, some for PBR, and some for redistribution filtering.

Table 4-15. *match* Commands

| Command                                                                                                                                                                                        | Description                                                                                                                                                                                                        |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>match ip address</b> { <i>access-list-number</i>   <i>name</i> } [... <i>access-list-number</i>   <i>name</i> ]   <b>prefix-list</b> <i>prefix-list-name</i> [... <i>prefix-list-name</i> ] | Matches any routes that have a network number that is permitted by a standard or extended access list or prefix list. Multiple access lists or prefix lists can be specified; matching any one results in a match. |
| <b>match length</b> <i>min max</i>                                                                                                                                                             | Matches based on a packet's Layer 3 length.                                                                                                                                                                        |
| <b>match interface</b> <i>type number</i>                                                                                                                                                      | Matches any routes that have the next hop out of one of the interfaces specified.                                                                                                                                  |
| <b>match ip next-hop</b> { <i>access-list-number</i>   <i>access-list-name</i> } [... <i>access-list-number</i>   ... <i>access-list-name</i> ]                                                | Matches any routes that have a next-hop router address permitted by one of the access lists specified.                                                                                                             |
| <b>match ip route-source</b> { <i>access-list-number</i>   <i>access-list-name</i> } [... <i>access-list-number</i>   ... <i>access-list-name</i> ]                                            | Matches routes that have been advertised by routers and access servers that have an address permitted by one of the access lists specified.                                                                        |

| Command                                                                                                        | Description                                    |
|----------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| <b>match metric</b> <i>metric-value</i>                                                                        | Matches routes that have the metric specified. |
| <b>match route-type</b> [ <b>external</b>   <b>internal</b>   <b>level-1</b>   <b>level-2</b>   <b>local</b> ] | Matches routes of the specified type.          |
| <b>match community</b> { <i>list-number</i>   <i>list-name</i> }                                               | Matches a BGP community.                       |
| <b>match tag</b> <i>tag-value</i>                                                                              | Matches based on the tag of a route.           |

The set condition route map configuration commands change or add characteristics, such as metrics, to any routes that have met a match criterion and the action to be taken is permit. (The consequences of a deny action depend on how the route map is being used.) Table 4-16 lists some of the variety of set commands that are available. Not all the set commands listed here are used for redistribution purposes; the table includes commands for BGP and PBR.

Table 4-16. *set* Commands

| Command                                                                                      | Description                                                                                                                                                                                                                                                                          |
|----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>set metric</b> <i>metric-value</i>                                                        | Sets the metric value for a routing protocol.                                                                                                                                                                                                                                        |
| <b>set metric-type</b> [ <b>type-1</b>   <b>type-2</b>   <b>internal</b>   <b>external</b> ] | Sets the metric type for the destination routing protocol.                                                                                                                                                                                                                           |
| <b>set default interface</b> <i>type number</i> [... <i>type number</i> ]                    | Indicates where to send output packets that pass a match clause of a route map for policy routing and for which the Cisco IOS Software has no explicit route to the destination.                                                                                                     |
| <b>set interface</b> <i>type number</i> [... <i>type number</i> ]                            | Indicates where to send output packets that pass a match clause of a route map for policy routing.                                                                                                                                                                                   |
| <b>set ip default next-hop</b> <i>ip-address</i> [... <i>ip-address</i> ]                    | Indicates where to send output packets that pass a match clause of a route map for policy routing and for which the Cisco IOS Software has no explicit route to the destination.                                                                                                     |
| set ip default next-hop verify-availability                                                  | Forces the router to check the CDP database to determine if an entry is available for the next hop that is specified by the <b>set ip default next-hop</b> command. This command is used to prevent traffic from being “black holed” if the configured next hop becomes unavailable. |
| <b>set ip next-hop</b> <i>ip-address</i> [... <i>ip-address</i> ]                            | Indicates where to send output packets that pass a match clause of a route map for policy routing.                                                                                                                                                                                   |
| set ip next-hop verify-availability                                                          | Forces the router to check the Cisco Discovery Protocol (CDP) database or use object tracking to determine if the next hop that is specified for policy based routing is available.                                                                                                  |
| set ip vrf                                                                                   | Indicates where to forward packets that pass a match clause of a route map for policy routing when the next hop must be under a specified virtual routing and forwarding (VRF) name.                                                                                                 |
| set next-hop                                                                                 | Specifies the address of the next hop.                                                                                                                                                                                                                                               |
| <b>set level</b> [ <b>level-1</b>   <b>level-2</b>   <b>stub-area</b>   <b>backbone</b> ]    | Indicates at what level or type of area to import routes into (for IS-IS and OSPF routes).                                                                                                                                                                                           |
| <b>set as-path</b> { <b>tag</b>   <b>prepend</b> <i>as-path-string</i> }                     | Modifies an autonomous system path for BGP routes.                                                                                                                                                                                                                                   |

| Command                                                                                                            | Description                                                            |
|--------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| set automatic-tag                                                                                                  | Automatically computes the BGP tag value.                              |
| <b>set community</b> { <i>community-number</i> [ <b>additive</b> ] [ <i>well-known-community</i> ]   <b>none</b> } | Sets the BGP community attribute.                                      |
| <b>set local-preference</b> <i>bgp-path-attributes</i>                                                             | Specifies a local preference value for the BGP autonomous system path. |
| <b>set weight</b> <i>bgp-weight</i>                                                                                | Specifies the BGP weight value.                                        |
| <b>set origin</b> <i>bgp-origin-code</i>                                                                           | Specifies the BGP origin code.                                         |
| set tag                                                                                                            | Specifies the tag value for destination routing protocol.              |

### Configuring Route Maps for Policy Based Routing

PBR is described in detail in Chapter 5, "Implementing Path Control." This section briefly introduces how route maps are used for PBR.

You might enable policy routing if you want your packets to take a route other than the obvious shortest path. The following should be documented in an implementation plan when planning route map configuration for PBR:

- Define the route map, including defining the conditions to match (the **match** statements), and defining the action to be taken when there is a match (the **set** statements).
- Define which interface the route map will be attached to. PBR is applied to *incoming* packets. Enabling PBR causes the router to evaluate all packets incoming on the interface using a route map configured for that purpose.

To define the conditions for policy routing packets, apply the `ip policy route-map map-tag interface` configuration command. The `ip policy route-map MyRouteMap` command in Example 4-16 declares that a route map named `MyRouteMap` is to be used for the policy routing on an interface. The route map has one statement; it is a permit. The `match` command specifies the match criteria—the conditions under which policy routing is allowed for the interface, based on the destination IP address of the packet. The `set` command specifies the set actions, which are the particular policy routing actions to perform if the criteria enforced by the `match` commands are met. In this route map, the `route-map MyRouteMap` command includes `amatch` command that matches IP address 172.21.16.18, and a `set` command that sets the next hop to 172.30.3.20 when the condition under `match` command is met. Packets from 172.21.16.18 will be sent to 172.30.3.20.

Example 4-16. Using a Route Map for Policy-Based Routing

```
interface serial 0/0/0
 ip policy route-map MyRouteMap
 !
route-map MyRouteMap permit 10
 match ip address 1
 set ip next-hop 172.30.3.20
 !

access-list 1 permit 172.21.16.18 0.0.0.0
```

### Configuring Route Redistribution Using Route Maps

Use route maps when you want detailed control over how routes are redistributed between routing protocols. The **route-map** *map-tag* parameter in the **redistribute** commands specifies the route map used.

The following should be documented in an implementation plan when planning route map configuration for redistribution:

- Define the route map, including defining the conditions to match (the **match** statements), and defining the action to be taken when there is a match (the **set** statements).
- Define where the route map will be used to control redistribution.

## Using Route Maps with Redistribution

Example 4-17 illustrates a route map being used to redistribute RIPv1 into OSPF 10. The route map, called `redis-rip`, is used in the `redistribute rip route-map redis-rip subnets` command under the OSPF process.

Example 4-17. Redistributing RIPv1 into OSPF Using a Route Map

```
router ospf 10
 redistribute rip route-map redis-rip subnets

route-map redis-rip permit 10
 match ip address 23 29
 set metric 500
 set metric-type type-1

route-map redis-rip deny 20
 match ip address 37

route-map redis-rip permit 30
 set metric 5000
 set metric-type type-2

access-list 23 permit 10.1.0.0 0.0.255.255
access-list 29 permit 172.16.1.0 0.0.0.255
access-list 37 permit 10.0.0.0 0.255.255.255
```

Sequence number 10 of the route map is looking for an IP address match in access list 23 or access list 29. Routes 10.1.0.0/16 and 172.16.1.0/24 match these lists. If a match is found, the router redistributes the route into OSPF with a cost metric of 500 and sets the new OSPF route to external type 1.

If there is no match to sequence number 10, sequence number 20 is checked. If there is a match in access list 37 (10.0.0.0/8), that route is not redistributed into OSPF, because sequence number 20 specifies **deny**.

If there is no match to sequence number 20, sequence number 30 is checked. Because sequence number 30 is a **permit** and there is no match criterion, all remaining routes are redistributed into OSPF with a cost metric of 5000 and an external metric of type 2.

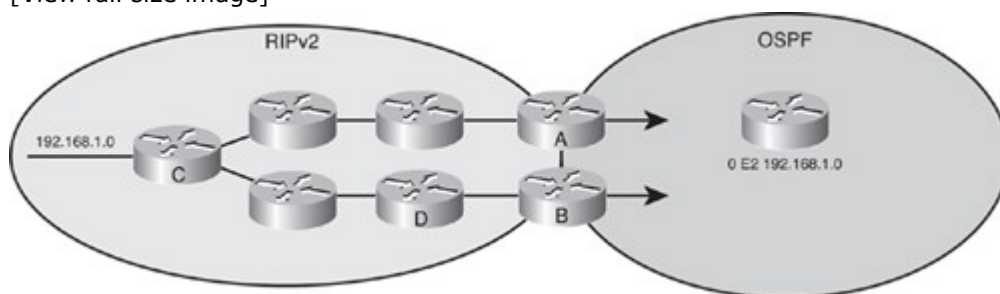
Note that the decision of whether to redistribute a route or not is based on the **deny** or **permit** in the **route-map** command, and not the **deny** or **permit** in the ACLs referenced by the **route-map** command. The ACLs are used only to match routes in the **route-map** statements.

## Using Route Maps to Avoid Route Feedback

As discussed, there is a possibility that routing feedback might cause suboptimal routing or a routing loop when routes are redistributed by more than one router. Figure 4-24 illustrates a network in which two-way multipoint redistribution (redistribution in both directions on both routers) is configured on Routers A and B. To prevent redistribution feedback loops, route maps are configured on both routers.

Figure 4-24. Route Maps Can Help Avoid Route Feedback Loops.

[View full size image]



The potential for routing feedback becomes apparent if you follow the advertisements for a specific network before route maps are configured. For example, RIPv2 on Router C advertises network 192.168.1.0. Routers

A and B redistribute the network into OSPF. OSPF then advertises the route to its neighbor OSPF routers as an OSPF external route. The route passes through the OSPF autonomous system and eventually makes its way back to the other edge router. Router B (or A) then redistributes 192.168.1.0 from OSPF back into the original RIPv2 network. This is a routing feedback loop.

To prevent the routing feedback loop, a route map called OSPF\_into\_RIP has been applied to Routers A and B when redistributing OSPF routes into RIP, as shown in Example 4-18. The route-map statement with sequence number 10 refers to access list 1, which matches the original RIPv2 network 192.168.1.0/16. (The access list should include all networks in the RIPv2 domain.) This route-map statement is a deny, so the 192.168.1.0 route is denied from being redistributed back into RIPv2. If the route does not match sequence number 10, the router then checks sequence number 20, which is an empty permit statement. This statement matches all routes, so all other routes are redistributed into RIP.

Example 4-18. Partial Configuration on Routers A and B in Figure 4-24

```
access-list 1 permit 192.168.1.0 0.0.0.255

route-map OSPF_into_RIP deny 10
 match ip address 1

route-map OSPF_into_RIP permit 20

router rip
 redistribute ospf 10 route-map OSPF_into_RIP

router ospf 10
 redistribute rip subnets
```

#### Using Route Maps with Tags

In Example 4-19, EIGRP 7 is configured on a router. The redistribute rip route-map rip\_to\_eigrp metric 10000 1000 255 1 1500 command configures redistribution of RIP routes into the EIGRP 7 routing process, with the route map named rip\_to\_eigrp used to influence the redistribution of RIP routes into EIGRP. The route-map statement 20 matches routes with a tag of 88. These routes are denied from being redistributed into EIGRP 7, because the route-map statement is a deny. Routes without a tag of 88 reach the second route-map statement and are tagged with 77, with the set tag 77 command. These routes are allowed to be distributed into EIGRP 7 because the route-map statement is a permit.

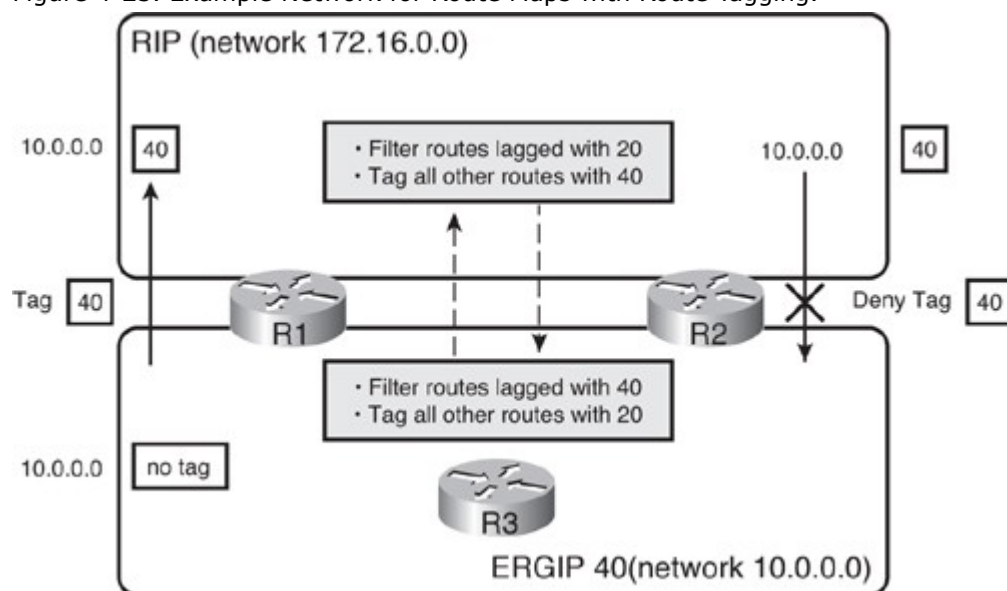
Example 4-19. Using Route Maps with Tags

```
route-map rip_to_eigrp deny 10
 match tag 88
route-map rip_to_eigrp permit 20
 set tag 77
router eigrp 7
 redistribute rip route-map rip_to_eigrp metric 10000 1000 255 1 1500
```

#### Using Route Maps with Redistribution and Tags

Figure 4-25 illustrates a network with RIP running in one part and EIGRP running in the other part; multipoint two-way redistribution is configured on Routers R1 and R2. To avoid routing loops, route maps with route tagging are used. The partial configuration for Router R1 is provided in Example 4-20. Router R2 has a similar configuration.

Figure 4-25. Example Network for Route Maps with Route Tagging.



Example 4-20. Partial Configuration of Router R1 in Figure 4-25

```
router eigrp 100
 redistribute rip metric 10000 100 255 1 1500 route-map intoeigrp
<output omitted>
!
router rip
 redistribute eigrp 100 metric 3 route-map intorip
<output omitted>
!
route-map intoeigrp deny 10
 match tag 40
!
route-map intoeigrp permit 20
 set tag 20
!
route-map intorip deny 10
 match tag 20
!
route-map intorip permit 20
 set tag 40
!
```

In this network, routes redistributed into RIP are tagged with the value 40, and routes redistributed into EIGRP are tagged with the value 20. At the same time, the route map filters tagged updates. Updates with the tag 20 are not allowed to go back into RIP and updates with the tag 40 are not allowed to go back into EIGRP.

The route maps each have two statements. When a route matches the tag in the **match** command in the first statement, the route is denied (it is not redistributed). All other routes are permitted (they are redistributed) by the second statement, and the **set** command tags the route appropriately.

Example 4-21 illustrates show ip route output from Routers R1 and R2, for networks 10.0.0.0 and 172.16.0.0. Notice that the route tags do not appear on the routes within the R1 and R2 routers, because these routers learn about all routes from both RIP and EIGRP directly. However, Example 4-22 illustrates show ip route output from Router R3, an internal router in the EIGRP network. Notice that



Router R3 does see network 172.16.0.0 with a tag of 20; this tag is carried with the route as R3 advertises it to other routers in the EIGRP network, including R1 and R2. When Routers R1 and R2 see the tag of 20, they do not redistribute the 172.16.0.0 route back into RIP.

Example 4-21. Output on Routers R1 and R2 in Figure 4-25

```
R1#show ip route 10.0.0.0
Routing entry for 10.0.0.0/8
 Known via "eigrp 100", distance 90, metric 20514560, type internal
 Redistributing via rip, eigrp 100
 Advertised by rip metric 3 route-map intorip
<output omitted>

R2#show ip route 172.16.0.0
Routing entry for 172.16.0.0/16
 Known via "rip", distance 120, metric 1
 Redistributing via rip, eigrp 100
 Advertised by eigrp 100 metric 10000 100 255 1 1500 route-map intoeigrp
<output omitted>
```

Example 4-22. Output from Router R3 in Figure 4-25

```
R3>show ip route 172.16.0.0
Routing entry for 172.16.0.0/16
 Known via "eigrp 100", distance 170, metric 20537600
 Tag 20, type external
<output omitted>
```

#### Using Distribute Lists

Another way to control routing updates is to use a distribute list. A distribute list allows an access list to be applied to routing updates.

As mentioned, access lists are usually associated with interfaces and are usually used to control *user* traffic (data plane traffic) rather than routing protocol traffic (or other control plane traffic). However, routers can have many interfaces, and route information can also be obtained through route redistribution, which does not involve a specific interface. In addition, access lists do not affect traffic originated by the router, so applying one on an interface has no effect on outgoing routing advertisements. However, when you configure an access list and use it with a distribute list, routing updates can be controlled, no matter what their source is.

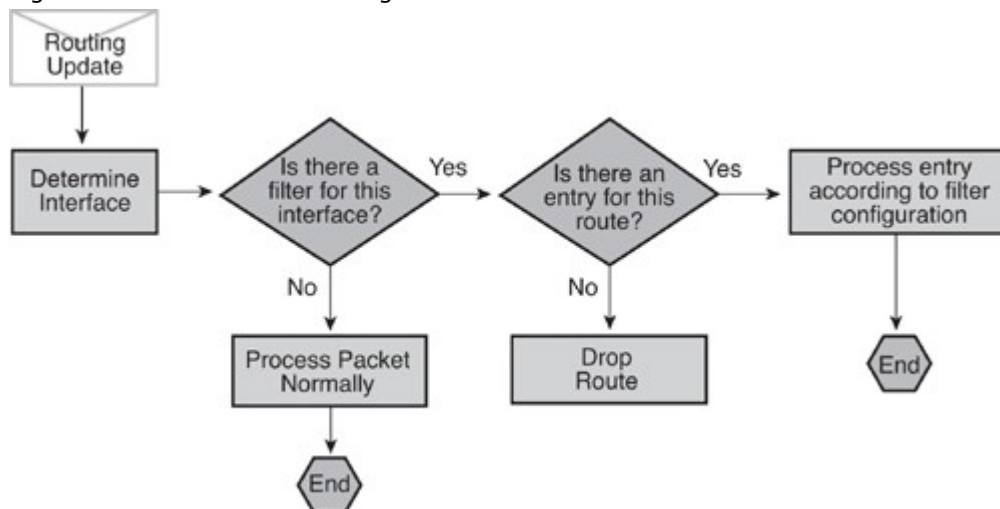
Access lists are configured in global configuration mode; the associated distribute list is configured under the routing protocol process. The access list should permit the networks that you want advertised or redistributed and deny the networks that you want to remain hidden. The router then applies the access list to routing updates for that protocol. Options in the **distribute-list** command allow updates to be filtered based on factors including the following:

- Incoming interface
- Outgoing interface
- Redistribution from another routing protocol

Using a distribute list gives the administrator great flexibility in determining just which routes will be permitted and which will be denied.

Figure 4-26 shows the general process that a router uses when filtering routing updates using a distribute list that is based on the incoming or outgoing interface. The process includes the following steps:

Figure 4-26. Route Filters Using a Distribute List.



1. The router receives a routing update or prepares to send an update about one or more networks.
2. The router looks at the interface involved with the action: the interface on which an incoming update has arrived, or, for an update that must be advertised, the interface out of which it should be advertised.
3. The router determines if a filter (distribute list) is associated with the interface.
4. If a filter (distribute list) is not associated with the interface, the packet is processed normally.
5. If a filter (distribute list) is associated with the interface, the router scans the access list referenced by the distribute list for a match for the given routing update.
6. If there is a match in the access list, the route entry is processed as configured. It is either permitted or denied by the matching access list statement.
7. If no match is found in the access list, the implicit **deny any** at the end of the access list causes the route entry to be dropped.

#### Configuring Distribute Lists to Control Routing Updates

You can filter routing update traffic for any protocol by defining an access list and applying it to a specific routing protocol using the **distribute-list** command. A distribute list enables the filtering of routing updates coming into or out of a specific interface from neighboring routers using the same routing protocol. A distribute list also allows the filtering of routes redistributed from other routing protocols or sources.

#### Planning a Distribute List Deployment

The following should be documented in an implementation plan when planning to configure distribute lists:

- Define the traffic filtering requirements that will be used to permit or deny routes, using an access list or a route map.
- Define a distribute list to use the access list or route map, and whether it will be applied to the inbound or outbound updates.

#### Configuring a Distribute List

To configure a distribute list, follow this procedure:

- Step 1. Identify the network addresses of the routes you want to filter, and create an access list.
- Step 2. Determine whether you want to filter traffic on an incoming interface, traffic on an outgoing interface, or routes being redistributed from another routing source.
- Step 3. Use the `distribute-list {access-list-number | name} out [interface-name | routing-process [routing-process parameter]]` router configuration command to assign the access list to filter outgoing routing updates. Table 4-17 explains the parameters of this command.

Table 4-17. *distribute-list out* Command

| Parameter                        | Description                                       |
|----------------------------------|---------------------------------------------------|
| <i>access-list-number   name</i> | Specifies the standard access list number or name |

| Parameter                 | Description                                                                                                                                               |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| out                       | Applies the access list to outgoing routing updates                                                                                                       |
| interface-name            | (Optional) Specifies the name of the interface out of which updates are filtered                                                                          |
| routing-process           | (Optional) Specifies the name of the routing process, or the keyword static or connected, that is being redistributed and from which updates are filtered |
| routing-process parameter | (Optional) Specifies a routing process parameter, such as the autonomous system number of the routing process                                             |

#### Note

Because OSPF routers must maintain LSDB synchronization within an area, the **distribute-list out** command cannot be used with OSPF to block outbound LSAs on an interface. For OSPF, this command works only on the routes being redistributed by ASBRs into OSPF; in other words, the command can be applied to external type 2 and external type 1 routes, but not to intra-area or interarea routes.

- Step 4. Alternatively, use the `distribute-list [access-list-number | name] [route-map map-tag] in [interface-type interface-number]` router configuration command to assign the access list to filter routing updates coming in through an interface. (This command allows the use of a route map instead of an access list for OSPF and EIGRP.) Table 4-18 explains the parameters of this command.
- Table 4-18. *distribute-list in* Command

| Parameter                        | Description                                                                                                                                                                                                       |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>access-list-number   name</i> | Specifies the standard access list number or name.                                                                                                                                                                |
| map-tag                          | (Optional) Specifies the name of the route map that defines which networks are to be installed in the routing table and which are to be filtered from the routing table. This argument is supported by OSPF only. |
| in                               | Applies the access list to incoming routing updates.                                                                                                                                                              |
| interface-type interface-number  | (Optional) Specifies the interface type and number from which updates are filtered.                                                                                                                               |

#### Note

The **distribute-list in** command prevents most routing protocols from placing the filtered routes in their database. However, OSPF routes cannot be filtered from entering the OSPF LSDB. Thus, when this command is used with OSPF, the routes are still placed in the LSDB; they are only filtered from entering the routing table.

It is important to understand the differences between these commands:

- The **distribute-list out** command filters updates going *out* of the interface or routing protocol specified in the command, *into* the routing process under which it is configured.
- The **distribute-list in** command filters updates going *into* the interface specified in the command, *into* the routing process under which it is configured.

#### IP Route Filtering with Distribute List Configuration Example

Figure 4-27 shows the topology of a network in which network 10.0.0.0 must be hidden from the devices in network 192.168.5.0.

Figure 4-27. Network 10.0.0.0 Needs to Be Hidden from Network 192.168.5.0.



Example 4-23 is the configuration of Router B in Figure 4-27. In this example, the distribute-list out command applies access list 7 to packets going out interface Serial 0/0/0. The access list allows only routing information about network 172.16.0.0 to be distributed out Router B's Serial 0/0/0 interface. The implicit deny any at the end of the access list prevents updates about any other networks from being advertised. As a result, network 10.0.0.0 is hidden.

Example 4-23. Filtering Out Network 10.0.0.0 on Router B in Figure 4-27

```
router eigrp 1
 network 172.16.0.0
 network 192.168.5.0
 distribute-list 7 out Serial0/0/0
!
access-list 7 permit 172.16.0.0 0.0.255.255
```

#### Note

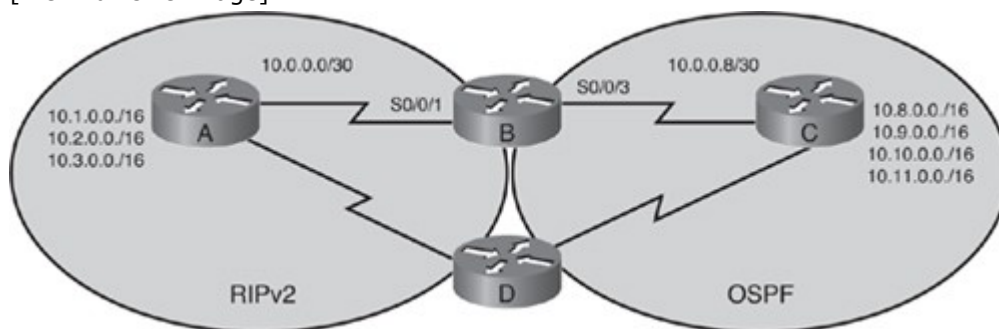
Another way to achieve the filtering of network 10.0.0.0 in this example is to deny network 10.0.0.0 and permit any other networks. This method would be particularly efficient if the routing information contained multiple networks but only network 10.0.0.0 needed filtering.

#### Controlling Redistribution with Distribute Lists

With mutual redistribution, using a distribute list helps prevent route feedback and routing loops. Route feedback occurs when routes originally learned from one routing protocol are redistributed back into that protocol. Figure 4-28 illustrates an example in which redistribution is configured both ways between RIPv2 and OSPF (two-way redistribution). The configuration on Router B is shown in Example 4-24. (Router D's configuration would be similar to Router B's configuration.)

Figure 4-28. Router B Controls Redistribution.

[View full size image]



Example 4-24. Configuration of Router B in Figure 4-28

```
router ospf 1
 network 10.0.0.8 0.0.0.3 area 0
 redistribute rip subnets
 distribute-list 2 out rip
```

```
router rip
 network 10.0.0.0
 version 2
 passive-interface Serial0/0/3
 redistribute ospf 1 metric 5
 distribute-list 3 out ospf 1

access-list 2 deny 10.8.0.0 0.3.255.255
access-list 2 permit any

access-list 3 permit 10.8.0.0 0.3.255.255
```

Router B redistributes the networks it learns via RIPv2 (in this case, 10.1.0.0 to 10.3.0.0) into OSPF. Without filtering, route feedback could occur if OSPF on Router D, the other redistribution point, redistributes those same networks back into RIP.

Therefore, the configuration in Example 4-24 includes a distribute list configuration that prevents route feedback. Access list 2 denies the routes originating in the OSPF network (10.8.0.0 through 10.11.0.0) and permits all others; the distribute list configured under OSPF refers to this access list. The result is that networks 10.8.0.0 to 10.11.0.0, originated by OSPF, are not redistributed back into OSPF from RIPv2. All other routes are redistributed into OSPF. Redistribution from OSPF into RIPv2 is filtered with access list 3; it permits routes 10.8.0.0 through 10.11.0.0 to be redistributed into RIPv2.

A distribute list hides network information, which could be considered a drawback in some circumstances. For example, in a network with redundant paths, a distribute list might permit routing updates for only specific paths, to avoid routing loops. In this case, other routers in the network might not know about the other ways to reach the filtered networks, so if the primary path goes down, the backup paths are not used because the rest of the network does not know they exist. When redundant paths exist, you should use other techniques.

#### Using Prefix Lists

As described in the previous section, to restrict the routing information that the Cisco IOS Software learns or advertises, you can filter routing updates to and from particular neighbors by defining either an access list with a distribute list, or a prefix list, and then apply it to the updates.

Using distribute lists as route filters has several drawbacks, including the following:

- A subnet mask cannot be easily matched.
- Access lists are evaluated sequentially for every IP prefix in the routing update.
- An extended access list can be cumbersome to configure.

The following sections detail the characteristics of prefix lists and how to filter with them. The configuration and verification of prefix lists is also covered.

#### Prefix List Characteristics

As discussed, access lists were originally designed to do packet filtering. Prefix lists can be used as an alternative to access lists in many route filtering commands. The advantages of using prefix lists include the following:

- A significant performance improvement over access lists in loading and route lookup of large lists. The router transforms the prefix list into a tree structure, with each branch of the tree representing a test, allowing the Cisco IOS Software to determine whether to permit or deny much faster.
- Support for incremental modifications. Compared to a traditional access list in which one **no** command erases the whole access list, prefix list entries can be modified incrementally. You can assign a sequence number to each line of a prefix list; the router uses this number to sort the entries in the list. If you initially sequence the lines with some gaps between the sequence numbers, you can easily insert lines later. You can also remove individual lines without removing the entire list.

Note

ACLs can now also be edited incrementally.

- A more user-friendly command-line interface. The command-line interface for using extended access lists to filter updates is difficult to understand and use.

- Greater flexibility. For example, routers match network numbers in a routing update against the prefix list using as many bits as indicated. For example, you can specify a prefix list to match 10.0.0.0/16, which will match 10.0.0.0 routes but not 10.1.0.0 routes. Optionally, the prefix list can also specify the size of the subnet mask, or that the subnet mask must be in a specified range.

Prefix lists have several similarities to access lists. A prefix list can consist of any number of lines, each of which indicates a test and a result. When a router evaluates a route against the prefix list, the first line that matches results in either a permit or deny. If none of the lines in the list match, the result is "implicitly deny," just as it is in an access list.

#### Filtering with Prefix Lists

Filtering by prefix list involves matching the prefixes of routes with those listed in the prefix list, similar to using access lists.

Whether a prefix is permitted or denied is based on the following rules:

- An empty prefix list permits all prefixes.
- If a prefix is permitted, the route is used. If a prefix is denied, the route is not used.
- Prefix lists consist of statements with sequence numbers. The router begins the search for a match at the top of the prefix list, which is the statement with the lowest sequence number.
- When a match occurs, the router does not need to go through the rest of the prefix list. For efficiency, you might want to put the most common matches (permits or denies) near the top of the list by specifying a lower sequence number.
- An implicit **deny** is assumed if a given prefix does not match any entries in a prefix list.

#### Configuring Prefix Lists

The `ip prefix-list {list-name | list-number} [seq seq-value] {deny | permit} network/length [ge ge-value] [le le-value]` global configuration command is used to create a prefix list, as described in Table 4-19.

Table 4-19. *ip prefix-list* Command Description

| Parameter            | Description                                                                                                                                                                                                                  |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| list-name            | The name of the prefix list that will be created (case sensitive).                                                                                                                                                           |
| list-number          | The number of the prefix list that will be created.                                                                                                                                                                          |
| <b>seq</b> seq-value | A 32-bit sequence number of the <b>prefix-list</b> statement, used to determine the order in which the statements are processed when filtering. Default sequence numbers are in increments of 5 (5, 10, 15, and so on).      |
| deny   permit        | The action taken when a match is found.                                                                                                                                                                                      |
| network/length       | The prefix to be matched and the length of the prefix. The network is a 32-bit address. The length is a decimal number.                                                                                                      |
| <b>ge</b> ge-value   | The range of the prefix length to be matched for prefixes that are more specific than <i>network/length</i> . The range is assumed to be from <i>ge-value</i> to 32 if only the <b>ge</b> attribute is specified.            |
| <b>le</b> le-value   | The range of the prefix length to be matched for prefixes that are more specific than <i>network/length</i> . The range is assumed to be from <i>length</i> to <i>le-value</i> if only the <b>le</b> attribute is specified. |

The **ge** and **le** keywords are optional. They can be used to specify the range of the prefix length to be matched for prefixes that are more specific than *network/length*. The value range is as follows  
 $length < ge-value < le-value \leq 32$

An exact match is assumed when neither **ge** nor **le** is specified.

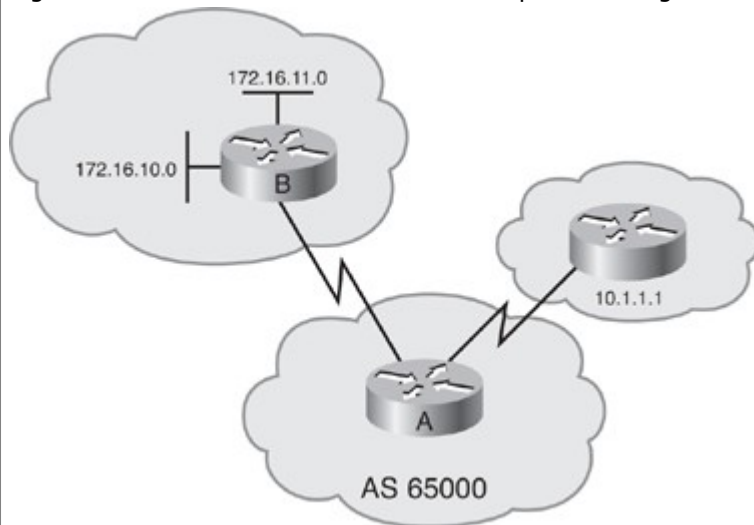
Prefix list entries can be reconfigured incrementally. In other words, an entry can be deleted or added individually.

|                                |
|--------------------------------|
| ip prefix-list Command Options |
|--------------------------------|

The use of the **ge** and **le** options in the **ip prefix-list** command can be confusing. The following are results of some testing done to understand these keywords.

Three routers were used in this testing: Router B, Router A, and its neighbor 10.1.1.1, as illustrated in Figure 4-29.

Figure 4-29. Network Used in Prefix List Option Testing.



Before configuring the prefix list, Router A learns the following routes (from Router B):

172.16.0.0 subnetted:

172.16.10.0/24

172.16.11.0/24

In these scenarios some BGP commands, as described in Chapter 6, are used. At this point the workings of these commands is not important. The important facts for this testing are that Router A initially has two /24 routes and is aggregating (summarizing) those routes to 172.16.0.0/16 so that the summary /16 route and the two /24 routes are available to be advertised. The prefix list determines which routes neighbor 10.1.1.1 learns from Router A.

Five scenarios were tested:

- **Scenario 1**— In this scenario, the following is configured on Router A:  

```
router bgp 65000
 aggregate-address 172.16.0.0 255.255.0.0
 neighbor 10.1.1.1 prefix-list tenonly out
ip prefix-list tenonly permit 172.16.10.0/8 le 24
```

 When you view the router's configuration with the **show run** command, you see that the router automatically changes the last line of this configuration to the following:  

```
ip prefix-list tenonly permit 172.0.0.0/8 le 24
```

 This is because only the first 8 bits in the address are considered significant when a prefix length of /8 is used. In this case, neighbor 10.1.1.1 learns about 172.16.0.0/16, 172.16.10.0/24, and 172.16.11.0/24. These are the routes that match the first 8 bits of 172.0.0.0 and have a prefix length between 8 and 24.
- **Scenario 2**— In this scenario, the following is configured on Router A:  

```
router bgp 65000
 aggregate-address 172.16.0.0 255.255.0.0
 neighbor 10.1.1.1 prefix-list tenonly out
ip prefix-list tenonly permit 172.0.0.0/8 le 16
```

 Neighbor 10.1.1.1 learns only about 172.16.0.0/16. This is the only route that matches the first 8 bits of 172.0.0.0 and has a prefix length between 8 and 16.
- **Scenario 3**— In this scenario, the following is configured on Router A:



```
router bgp 65000
 aggregate-address 172.16.0.0 255.255.0.0
 neighbor 10.1.1.1 prefix-list tenonly out
ip prefix-list tenonly permit 172.0.0.0/8 ge 17
Neighbor 10.1.1.1 learns only about 172.16.10.0/24 and 172.16.11.0/24. (In other words, Router A ignores the /8 parameter and treats the command as if it has the parameters ge 17 le 32.)
```

- **Scenario 4**— In this scenario, the following is configured on Router A:

```
router bgp 65000
 aggregate-address 172.16.0.0 255.255.0.0
 neighbor 10.1.1.1 prefix-list tenonly out
ip prefix-list tenonly permit 172.0.0.0/8 ge 16 le 24
Neighbor 10.1.1.1 learns about 172.16.0.0/16, 172.16.10.0/24, and 172.16.11.0/24. (In other words, Router A ignores the /8 parameter and treats the command as if it has the parameters ge 16 le 24.)
```

- **Scenario 5**— In this scenario, the following is configured on router A:

```
router bgp 65000
 aggregate-address 172.16.0.0 255.255.0.0
 neighbor 10.1.1.1 prefix-list tenonly out
ip prefix-list tenonly permit 172.0.0.0/8 ge 17 le 24
```

Neighbor 10.1.1.1 learns about 172.16.10.0/24 and 172.16.11.0/24. (In other words, Router A ignores the /8 parameter and treats the command as if it has the parameters **ge 17 le 24**.)

The **no ip prefix-list list-name** global configuration command, where *list-name* is the name of a prefix list, is used to delete a prefix list.

The **[no] ip prefix-list list-name description text** global configuration command can be used to add or delete a text description for a prefix list.

#### Prefix List Sequence Numbers

Prefix list sequence numbers are generated automatically, unless you disable this automatic generation. If you do so, you must specify the sequence number for each entry using the *seq-value* argument of the **ip prefix-list** command.

A prefix list is an ordered list. The sequence number is significant when a given prefix is matched by multiple entries of a prefix list, in which case the one with the smallest sequence number is considered the real match.

The evaluation of a prefix list starts with the lowest sequence number and continues down the list until a match is found. When an IP address match is found, the **permit** or **deny** statement is applied to that network and the remainder of the list is not evaluated.

#### Tip

For best performance, the most frequently processed prefix list statements should be configured with the lowest sequence numbers. The **seqseq-value** keyword and argument can be used for resequencing.

Regardless of whether you use the default sequence numbers to configure a prefix list, you do not need to specify a sequence number when removing a configuration entry.

By default, a prefix list's entries have sequence values of 5, 10, 15, and so on. In the absence of a specified sequence value, a new entry is assigned a sequence number equal to the current maximum sequence number plus 5. For example, if the first configured sequence number is 3, subsequent entries will be 8, 13, and 18.

Prefix list **show** commands include the sequence numbers in their output.

The **no ip prefix-list sequence-number** global configuration command is used to disable the automatic generation of sequence numbers of prefix list entries. Use the **ip prefix-list sequence-number** global configuration command to reenab the automatic generation of sequence numbers.

#### Prefix List Examples

Consider the prefix list: **ip prefix-list MyList permit 192.168.0.0/16**. Which of the following routes would this prefix list match: 192.168.0.0/16, 192.168.0.0/20, 192.168.2.0/24?

Only the first route, 192.168.0.0/16, would match because that is the only one that matches both the address and the mask.

For the next example, consider the two prefix lists:

- ip prefix-list List1 permit 192.168.0.0/16 le 20
- ip prefix-list List2 permit 192.168.0.0/16 ge 18

Which of the following routes would these prefix lists match: 192.168.0.0/16, 192.168.0.0/20, 192.168.2.0/24?

The following routes match List 1: 192.168.0.0/16, 192.168.0.0/20. The route 192.168.2.0/24 is not matched because, even though the IP address falls within the specified address range, the subnet mask is too long.

The following routes match List 2: 192.168.0.0/20, 192.168.2.0/24. The route 192.168.0.0/16 is not matched because the subnet mask is too short.

Another example is **ip prefix-list NextList1 0.0.0.0/0**. The all-0s prefix means match all prefixes. In this case, without any **le** or **ge** parameter, the /0 mask must also be matched. Only a default route matches this prefix list.

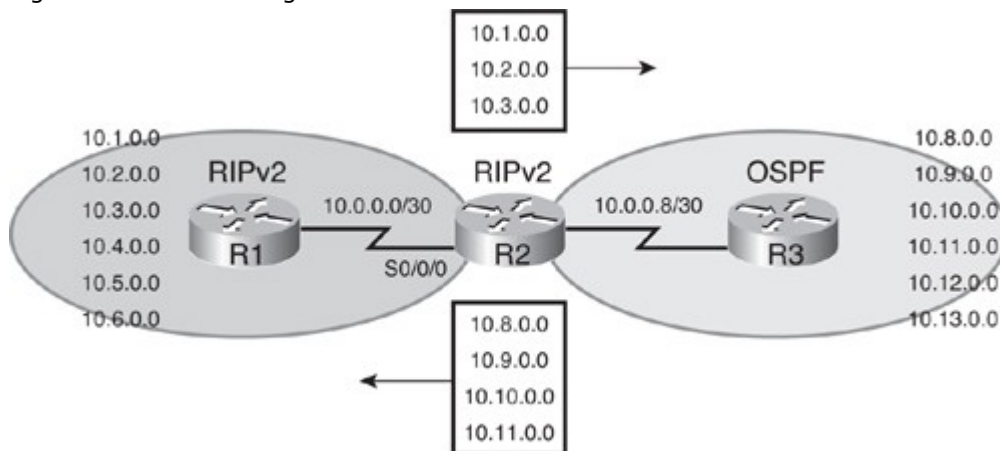
A modification of the above is **ip prefix-list NextList20.0.0.0/0 ge 32**. In this case any route with a /32 prefix matches this list.

A similar example is **ip prefix-list NextList3 0.0.0.0/0 le 32**. The **le 32** parameter means that any prefix length between 0 and 32, inclusive, matches. Thus, this prefix list matches all routes.

Another similar example is **ip prefix-list NextList4 0.0.0.0/1 le 24**. Any route with a prefix between /1 and /24 matches this list.

For the next example, consider Figure 4-30 and Example 4-25. Two prefix lists are configured, each one used in a route map.

Figure 4-30. Controlling Redistribution with Prefix Lists.



Example 4-25. Configuration of Router R2 in Figure 4-30

```
router ospf 1
 network 10.0.0.8 0.0.0.0 area 0
 redistribute rip route-map intoOSPF subnets

router rip
 network 10.0.0.0
 version 2
 passive-interface s0/0/0
 redistribute ospf 1 route-map intoRIP metric 5

route-map intoOSPF permit 10
 match ip address prefix-list PFX1
```

```

route-map intoRIP permit 10
 match ip address prefix-list PFX2

ip prefix-list PFX1 permit 10.0.0.0/14
ip prefix-list PFX2 permit 10.8.0.0/14

```

The *intoOSPF* route map uses prefix list PFX1. This permits 10.0.0.0/14, which includes 10.0.0.0 through 10.3.0.0. These routes are redistributed from RIP into OSPF. All other networks are not permitted by prefix list PFX1 and are therefore denied (not redistributed).

The *intoRIP* route map uses prefix list PFX2. This permits 10.8.0.0/14, which includes 10.8.0.0 through 10.11.0.0, some of the routes originated by OSPF. These routes are redistributed from OSPF into RIP. All other networks are not permitted by prefix list PFX2 and are therefore denied (not redistributed).

Note that the decision of whether to redistribute a route or not is based on the **deny** or **permit** in the **route-map** command, and not the **deny** or **permit** in the prefix list referenced by the **route-map** command. The prefix list is used only to match routes in the **route-map** statements.

In a network with redundant paths, the goal of using a prefix lists and route maps may be to prevent routing loops. As with distribute lists, prefix lists permit routing updates that enable only the desired paths to be advertised. Therefore, other routers in the network do not know about the other ways to reach the filtered networks.

#### Verifying Prefix Lists

The EXEC commands related to prefix lists are described in Table 4-20. Use the `show ip prefix-list?` command to see all the show commands available for prefix lists.

Table 4-20. Commands Used to Verify Prefix Lists

| Command                                                                        | Description                                                                                                                                                                               |
|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>show ip prefix-list [detail   summary]</code>                            | Displays information on all prefix lists. Specifying the <b>detail</b> keyword includes the description and the hit count (the number of times the entry matches a route) in the display. |
| <code>show ip prefix-list [detail   summary] prefix-list-name</code>           | Displays a table showing the entries in a specific prefix list.                                                                                                                           |
| <code>show ip prefix-list prefix-list-name [network/length]</code>             | Displays the policy associated with a specific <i>network/length</i> in a prefix list.                                                                                                    |
| <code>show ip prefix-list prefix-list-name [seq sequence-number]</code>        | Displays the prefix list entry with a given sequence number.                                                                                                                              |
| <code>show ip prefix-list prefix-list-name [network/length] longer</code>      | Displays all entries of a prefix list that are more specific than the given network and length.                                                                                           |
| <code>show ip prefix-list prefix-list-name [network/length] first-match</code> | Displays the entry of a prefix list that matches the network and length of the given prefix.                                                                                              |
| <code>clear ip prefix-list prefix-list-name [network/length]</code>            | Resets the hit count shown on prefix list entries.                                                                                                                                        |

In the sample output of the `show ip prefix-list detail` command shown in Example 4-26, Router A has a prefix list called *superonly* that has only one entry (sequence number 5). The hit count of 0 means that no routes match this entry.

Example 4-26. `show ip prefix-list detail` Command Output

```

Code View: Scroll / Show All
RtrA #show ip prefix-list detail
Prefix-list with the last deletion/insertion: superonly ip prefix-list superonly:

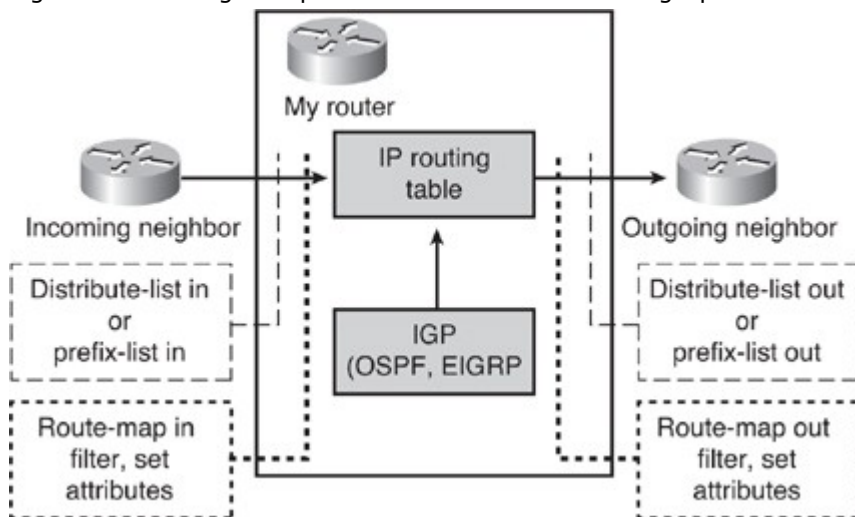
```

Description: only permit supernet  
count: 1, range entries: 0, sequences: 5 - 5, refcount: 1  
seq 5 permit 172.0.0.0/8 (hit count: 0, refcount: 1)

#### Using Multiple Methods to Control Routing Updates

You can apply a combination of prefix lists, distribute lists, and route maps to incoming or outgoing information, or both, as illustrated in Figure 4-31. The incoming prefix list, the incoming distribute list, and the incoming route map must all permit the routes that are received from a neighbor before they will be accepted into the IP routing table. Similarly, outgoing routes must pass the outgoing distribute list, the outgoing prefix list, and the outgoing route map before they will be transmitted to the neighbor.

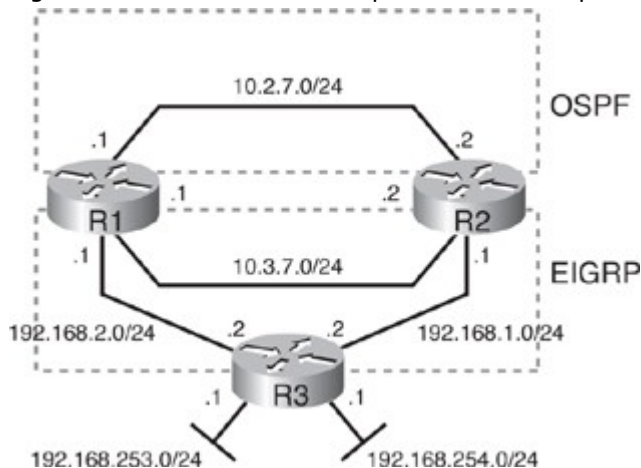
Figure 4-31. Using Multiple Methods to Control Routing Updates.



#### Comprehensive Example of Controlling Routing Updates

This section provides a comprehensive example of how routing updates can be controlled, using multiple methods. Figure 4-32 shows the network used in this example. Both R1 and R2 are redistributing OSPF into EIGRP and EIGRP into OSPF. Notice that two links exist between R1 and R2, one in OSPF and one in EIGRP.

Figure 4-32. Network for Comprehensive Example.



R3's loopback interface addresses are 192.168.254.1 and 192.168.253.1. These interfaces are not running EIGRP. R3 is injecting these routes into the EIGRP network. Router R1 translates those EIGRP advertisements into OSPF, with a specified seed metric. These OSPF advertisements are translated back into EIGRP by Router R2. However, as we shall see, the resulting metric is better than the one advertised by R3, so R1 determines

that the best path to R3's loopback address is via R2. We shall also see that there is a routing loop in this network.

In this example, prefix lists, route tagging, and route filtering are used to fix the routing loop issue, and the administrative distance is changed to fix the suboptimal routing issue.

A prefix list is used to identify specific prefixes. As discussed, other tools could be used to identify routes, but prefix lists are flexible and allow both the prefix and mask to be matched.

When Router R3 injects 192.168.254.0 and 192.168.253.0 into EIGRP, all routers in the EIGRP network learn the routes. In R1, the routes are redistributed into OSPF and will be tagged. When R2 is redistributing routes from OSPF into EIGRP, it will examine the routes for the tag; if the route is tagged, it will not be redistributed into EIGRP because the tag indicates that it originated in EIGRP. Routes being redistributed by R2 into OSPF and then by R1 into EIGRP will be handled in a similar manner.

This example goes through the following steps:

- Step 1. Verify the scenario.
  - Step 2. Identify specific routes with prefix lists.
  - Step 3. Configure route maps to set tags according to the prefix list.
  - Step 4. Add the route map to the **redistribute** command.
  - Step 5. Verify the configuration.
  - Step 6. Display routing table changes and trace to key destinations to verify operations.
- First, the scenario is verified. The focus is on the 192.168.254.0 route, but the results for the 192.168.253.0 route are similar. Example 4-27 shows R1's routing table entry for the 192.168.254.0 route.

Example 4-27. Router R1's Route to Router R3's Loopback Interface

```
Code View: Scroll / Show All
R1#show ip route 192.168.254.0
Routing entry for 192.168.254.0 255.255.255.0
 Known via "eigrp 1", distance 170, metric 28160, type external
 Redistributing via eigrp 1, ospf 1
 Advertised by ospf 1 metric 4 subnets
 Last update from 10.3.7.2 on FastEthernet0/1.2, 01:38:46 ago
 Routing Descriptor Blocks:
 * 10.3.7.2, from 10.3.7.2, 01:38:46 ago, via FastEthernet0/1.2
 Route metric is 28160, traffic share count is 1
 Total delay is 100 microseconds, minimum bandwidth is 100000 Kbit
 Reliability 255/255, minimum MTU 500 bytes
 Loading 1/255, Hops 1

R1#
```

Notice in Example 4-27 that Router R1 is learning about this route from Router R2, 10.3.7.2, instead of directly from Router R3. This is suboptimal routing.

Example 4-28 illustrates the results of a trace to 192.168.254.1, Router R3's loopback interface, on Router R1. Notice the routing loop between R1 and R2. Thus, this network has a routing loop and suboptimal routing.

Example 4-28. *trace* Command Output on Router R1

```
Code View: Scroll / Show All
R1#trace 192.168.254.1 ttl 0 255

Type escape sequence to abort.
```

Tracing the route to 192.168.254.1

```
0 10.3.7.2 0 msec 0 msec 4 msec
1 10.3.7.2 0 msec 0 msec 4 msec
2 10.2.7.1 0 msec 0 msec 0 msec
3 10.3.7.2 4 msec 0 msec 4 msec
4 10.2.7.1 0 msec 0 msec 0 msec
5 10.3.7.2 0 msec 0 msec 4 msec
6 10.2.7.1 0 msec 0 msec 4 msec
7 10.3.7.2 0 msec 0 msec 4 msec
8 10.2.7.1 0 msec 0 msec 4 msec
9 10.3.7.2 0 msec 0 msec 0 msec
10 10.2.7.1 0 msec 4 msec 0 msec
 <output omitted>
250 10.2.7.1 8 msec 12 msec 8 msec
251 10.3.7.2 8 msec 12 msec 8 msec
252 10.2.7.1 12 msec 8 msec 12 msec
253 10.3.7.2 8 msec 12 msec 8 msec
254 10.2.7.1 8 msec 12 msec 8 msec
255 10.3.7.2 12 msec 8 msec 12 msec
R1#
```

While investigating the cause of these issues, some output on the routers is examined. Example 4-29 illustrates output from the `show ip protocols` command on Router R1, confirming that Router R1 is redistributing OSPF into EIGRP and redistributing EIGRP into OSPF.

Example 4-29. *show ip protocols* Command Output on Router R1

Code View: Scroll / Show All

R1#**show ip protocols**

Routing Protocol is "eigrp 1"

```
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 metric weight K1=1, K2=0, K3=1, K4=0, K5=0
EIGRP maximum hopcount 100
EIGRP maximum metric variance 1
Redistributing: eigrp 1, ospf 1
EIGRP NSF-aware route hold timer is 240s
```

<output omitted>

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 10.0.7.1
It is an autonomous system boundary router
Redistributing External Routes from,
 eigrp 1 with metric mapped to 4, includes subnets in redistribution
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
Maximum path: 4
```

Routing for Networks:

10.2.0.0 0.0.255.255 area 0

172.31.8.0 0.0.0.255 area 0

R1's EIGRP topology table is examined next, and the results are displayed in Example 4-30. Notice the metric for the first path (28160), via R2, is much lower than the metric for the second path (2297856), via R3. This is why R1 chooses the path via R2 instead of via R3. However, the path via R2 is the redistributed route from OSPF, and the route was redistributed into OSPF by R1. The result is the routing loop and suboptimal routing.

Example 4-30. *show ip eigrp topology* Command Output on Router R1

Code View: Scroll / Show All

R1#**show ip eigrp topology 192.168.254.0**

IP-EIGRP (AS-1): Topology entry for 192.168.254.0 255.255.255.0

State is Passive, Query origin flag is 1, 1 Successor(s), FD is 28160

Routing Descriptor Blocks:

10.3.7.2 (FastEthernet0/1.2), from 10.3.7.2, Send flag is 0x0

Composite metric is (28160/256), Route is External

Vector metric:

Minimum bandwidth is 100000 Kbit

Total delay is 100 microseconds

Reliability is 255/255

Load is 1/255

Minimum MTU is 500

Hop count is 1

External data:

Originating router is 10.0.7.2

AS number of route is 1

External protocol is OSPF, external metric is 4

Administrator tag is 0 (0x00000000)

192.168.2.2 (Serial0/0/0), from 192.168.2.2, Send flag is 0x0

Composite metric is (2297856/128256), Route is External

Vector metric:

Minimum bandwidth is 1544 Kbit

Total delay is 25000 microseconds

Reliability is 255/255

Load is 1/255

Hop count is 1

External data:

<output omitted>

This is confirmed by examining R2's routing table entry for 192.168.254.0. The results are shown in Example 4-31. Notice that this route is known by OSPF. Also notice that this route is being redistributed into EIGRP, and a seed metric of 10000000 0 255 1 500 is specified. It seems that R1 views this seed metric as better than R3's metric for the 192.168.254.0 network.

Example 4-31. *show ip route* Command Output on Router R2

Code View: Scroll / Show All

R2#**show ip route 192.168.254.0**



Routing entry for 192.168.254.0 255.255.255.0

Known via "ospf 1", distance 110, metric 4, type extern 2, forward metric 1

Redistributing via eigrp 1

Advertised by eigrp 1 metric 10000000 0 255 1 500 match external 1 & 2

Last update from 10.2.7.1 on Fast Ethernet0/1.1, 00:46:19 ago

Routing Descriptor Blocks:

\* 10.2.7.1, from 10.0.8.1, 00:46:19 ago, via FastEthernet0/1.1

Route metric is 4, traffic share count is 1

R2#

To fix these problems, a prefix list is first configured, as shown in Example 4-32 for R1 (a similar configuration is used on R2). In the prefix list, le 32 is used to identify any mask with a length less than or equal to 32 bits, which includes the prefix and any smaller subnet of the prefix. The prefix list is two lines, one for each network we want to identify.

Example 4-32. Prefix List Configuration on Router R1

```
R1(config)#ip prefix-list EXTERNAL seq 5 permit 192.168.253.0/24 le
32
R1(config)#ip prefix-list EXTERNAL seq 10 permit 192.168.254.0/24 le
32
R1(config)#
```

Route maps are configured next, as shown in Example 4-33. These route maps will be used to control redistribution. The first route map, SETTAG, uses the prefix list to match routes. Routes that match the prefix list are tagged with a value of 1000. The first statement in this route map is a permit; it permits routes that match the prefix list and sets the tag to 1000. All other routes, those that don't match the prefix list, fall to the second statement, which simply permits the routes (to be redistributed).

The second route map, MATCHTAG, also has two statements. Routes that have a tag that matches 1000 are denied (from being redistributed). These routes are the ones that were previously redistributed, so we do not want them to be redistributed again. All other routes, those that do not have a tag that matches 1000, fall to the second statement, which simply permits the routes (to be redistributed).

Example 4-33. Route Map Configuration on Router R1

```
R1(config)#route-map SETTAG permit 10
R1(config-route-map)#match ip address prefix-list
EXTERNAL
R1(config-route-map)#set tag 1000
R1(config-route-map)#exit
R1(config)#route-map SETTAG permit 20
R1(config-route-map)#exit
R1(config)#route-map MATCHTAG deny 10
R1(config-route-map)#match tag 1000
R1(config-route-map)#!
R1(config-route-map)#route-map MATCHTAG permit 20
R1(config-route-map)#
```

Next redistribution is configured to use the route maps, as shown in Example 4-34. The SETTAG route map is used when redistributing into OSPF. The MATCHTAG route map is used when redistributing into EIGRP.

Example 4-34. Redistribution Configuration on Router R1

Code View: Scroll / Show All

```
R1(config-router)#router ospf 1
R1(config-router)#redistribute eigrp 1 metric 4 subnets route-map SETTAG
R1(config-router)#router eigrp 1
R1(config-router)#redistribute ospf 1 metric 100 100 125 125 1500 match
external
1 external 2 route-map MATCHTAG
R1(config-router)#
```

Example 4-35 illustrates the similar configuration of R2.

Example 4-35. Configuration on Router R2

Code View: Scroll / Show All

```
R2(config)#ip prefix-list EXTERNAL seq 5 permit 192.168.253.0/24 le 32
R2(config)#ip prefix-list EXTERNAL seq 10 permit 192.168.254.0/24 le 32
R2(config)#
R2(config)#route-map SETTAG permit 10
R2(config-route-map)# match ip address prefix-list EXTERNAL
R2(config-route-map)# set tag 1000
R2(config-route-map)#route-map SETTAG permit 20
R2(config-route-map)#
R2(config-route-map)#route-map MATCHTAG deny 10
R2(config-route-map)#match tag 1000
R2(config-route-map)#route-map MATCHTAG permit 20
R2(config-route-map)#
R2(config-route-map)#router ospf 1
R2(config-router)#redistribute eigrp 1 metric 4 subnets route-map SETTAG
R2(config-router)#
R2(config-router)#router eigrp 1
R2(config-router)#redistribute ospf 1 metric 10000000 0 255 1 500 match external
1
external 2 route-map MATCHTAG
R2(config-router)#
```

The next step is to verify that traffic is now going the best way, and that the routing loop is no longer there. Example 4-36 illustrates output from the `show ip route` command on R1, illustrating that 192.168.254.0 is now reachable via EIGRP, from R3 (192.168.2.2). This output also shows that this route is being advertised into OSPF and that the route map SETTAG is used. (Note that displaying a specific entry in the routing table such as in this example is the only way to see that a route map is being used for redistribution.)

Example 4-36. `show ip route` Command Output on Router R1

Code View: Scroll / Show All

```
R1#show ip route 192.168.254.0
Routing entry for 192.168.254.0 255.255.255.0
 Known via "eigrp 1", distance 170, metric 2297856, type external
 Redistributing via eigrp 1, ospf 1
```

```
Advertised by ospf 1 metric 4 subnets route-map SETTAG
Last update from 192.168.2.2 on Serial0/0/0, 00:03:18 ago
```

```
Routing Descriptor Blocks:
```

```
* 192.168.2.2, from 192.168.2.2, 00:03:18 ago, via Serial0/0/0
 Route metric is 2297856, traffic share count is 1
 Total delay is 25000 microseconds, minimum bandwidth is 1544 Kbit
 Reliability is 255/255, minimum MTU 1500 bytes
 Loading 1/255, Hops 1
```

```
R1#
```

Example 4-37 illustrates the result of a trace to 192.168.254.1, confirming that the routing loop is no longer there.

Example 4-37. *trace* Command Output on Router R1

```
R1#trace 192.168.254.1
```

```
Type escape sequence to abort.
```

```
Tracing the route to 192.168.254.1
```

```
 1 192.168.2.2 12 msec * 12 msec
```

```
R1#
```

Other commands can also be used to verify this configuration. Example 4-38 verifies R1's route to 192.168.254.0 is via R3.

Example 4-38. *show ip route* Command Output on Router R1

```
R1#show ip route | section 254
```

```
D EX 192.168.254.0 255.255.255.0
```

```
 [170/2297856] via 192.168.2.2, 00:07:05, Serial0/0/0
```

```
R1#
```

Note

The **section** keyword causes only those lines that match a given expression (such as the text "254"), and any lines associated with that expression, to be displayed in the command output.

Example 4-39 illustrates R1's EIGRP topology table. From this output we can see that R1 now learns about this route from only R3. The route from R2 is no longer there.

Example 4-39. *show ip eigrp topology* Command Output on Router R1

```
R1#show ip eigrp topology 192.168.254.0
```

```
IP-EIGRP (AS 1): Topology entry for 192.168.254.0 255.255.255.0
```

```
 State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2297856
```

```
Routing Descriptor Blocks:
```

```
 192.168.2.2 (Serial0/0/0), from 192.168.2.2, Send flag is 0x0
```

```
 Composite metric is (2297856/128256), Route is External
```

```
Vector metric:
```

```
 Minimum bandwidth is 1544 Kbit
```

```
 Total delay is 25000 microseconds
```

```
 Reliability is 255/255
```

```
Load is 1/255
Minimum MTU is 1500
Hop count is 1
External data:
 Originating router is 3.3.3.3
 AS number of route is 0
 External protocol is Connected, external metric is 0
 Administrator tag is 0 (0x00000000)
```

R1#

Examining the OSPF tables verifies that the tagged route is being learned in the OSPF network. Example 4-40 illustrates a portion of R1's OSPF database, using the `show ip ospf database external` command. Notice that R1 is learning the 192.168.254.0 route via OSPF, and that this route is tagged with a value of 1000.

Example 4-40. *show ip ospf database external* Command Output on Router R1

```
R1#show ip ospf database external 192.168.254.0
```

```
OSPF Router with ID (1.1.1.1) (Process ID 1)
```

```
Type-5 AS External Link States
```

```
LS age: 539
Options: (No TOS-capability, DC)
LS Type: AS External Link
Link State ID: 192.168.254.0 (External Network Number)
Advertising Router: 1.1.1.1
LS Seq Number: 80000001
Checksum: 0x1642
Length: 36
Network Mask: 255.255.255.0
 Metric Type: 2 (Larger than any link state path)
 TOS: 0
 Metric: 4
 Forward Address: 0.0.0.0
 External Route Tag: 1000
```

R1#

To verify that the route map is really tagging the traffic, we might try the **show route-map** command. This command does show the route maps configured on the router. However, the match counters in this command only work when the route map is used for policy based routing, not when it is used for redistribution.

The `show ip prefix-list detail` command, illustrated in Example 4-41 is more useful in this case. Notice the hit counts for each sequence number in the prefix list in the command output, indicating how many times the prefix list matched the networks.

Example 4-41. *show ip prefix-list detail* Command Output on Router R1

```
R1#show ip prefix-list detail EXTERNAL
```

```
ip prefix-list EXTERNAL
```

```
count: 2, range entries: 2, sequences: 5 - 10, refcount: 3
seq 5 permit 192.168.253.0/24 le 32 (hit count: 2, refcount: 2)
seq 10 permit 192.168.254.0/24 le 32 (hit count: 2, refcount: 1)
```

R1#

To test the prefix list counters, the OSPF process is first cleared, forcing it to restart. After the OSPF neighbor relationship goes down and then back up, the router recalculates and retags the routes. Example 4-42 shows the clearing process and the resulting prefix list detail. Notice that the hit count on both networks has increased by 1, indicating a match. This prefix list match is used by the route map to set the route tag.

#### Caution

Clearing the OSPF process is not recommended in production networks unless absolutely necessary, because it resets all OSPF relationships.

#### Example 4-42. Resetting the OSPF Process on Router R1

Code View: Scroll / Show All

```
R1#clear ip ospf process
```

```
Reset ALL OSPF processes? [no]: yes
```

```
R1#
```

```
4d21h: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on FastEthernet0/1.1 from FULL to DOWN, Neighbor Down: Interface down or detached
```

```
R1#
```

```
R1#
```

```
4d21h: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on FastEthernet0/1.1 from LOADING to FULL, Loading Done
```

```
R1#
```

```
R1#show ip prefix-list detail EXTERNAL
```

```
ip prefix-list EXTERNAL
```

```
count: 2, range entries: 2, sequences: 5 – 10, refcount: 3
```

```
seq 5 permit 192.168.253.0/24 le 32 (hit count: 3, refcount: 2)
```

```
seq 10 permit 192.168.254.0/24 le 32 (hit count: 3, refcount: 1)
```

```
R1#
```

An alternative, simpler, configuration for the R1 and R2 routers is illustrated in Example 4-43. This configuration uses only one route map, TAGS, applied to both OSPF and EIGRP redistribution. The route map denies (does not redistribute) routes that are tagged, and it permits (redistributes) all other routes after setting their tag.

#### Example 4-43. Alternative Configuration for R1 and R2

```
route-map TAGS deny 10
```

```
match tag 1000
```

```
route-map TAGS permit 20
```

```
set tag 1000
```

```
<output omitted>
```

```
router ospf 1
```

```
redistribute eigrp 1 metric 4 route-map TAGS
```

```
<output omitted>
```

```
router eigrp 1
```

```
redistribute ospf 1 metric 1000000 0 255 1 800 route-map TAGS
```

```
<output omitted>
```

The routing loop problem is solved. However suboptimal routing still exists in this network, as shown in Example 4-44, illustrating R2's routing table. Notice that R2 is using the OSPF route, via R1 to reach R3's external networks. The trace output shown in Example 4-45 confirms the route R2 is taking.

Example 4-44. R2's Routing Table

R2#**show ip route 192.168.254.0**

Routing entry for 192.168.254.0 255.255.255.0

Known via "ospf 1", distance 110, metric 4

Tag 1000, type extern 2, forward metric 1

Redistributing via eigrp 1

Last update from 10.2.7.1 on FastEthernet0/1.1, 00:23:52 ago

Routing Descriptor Blocks:

\* 10.2.7.1, from 10.0.7.1, 00:23:52 ago, via FastEthernet0/1.1

Route metric is 4, traffic share count is 1

Route tag 1000

R2#

Example 4-45. R2 Reaches R3's External Routes Via R1

R2#**trace 192.168.254.1**

Type escape sequence to abort.

Tracing the route to 192.168.254.1

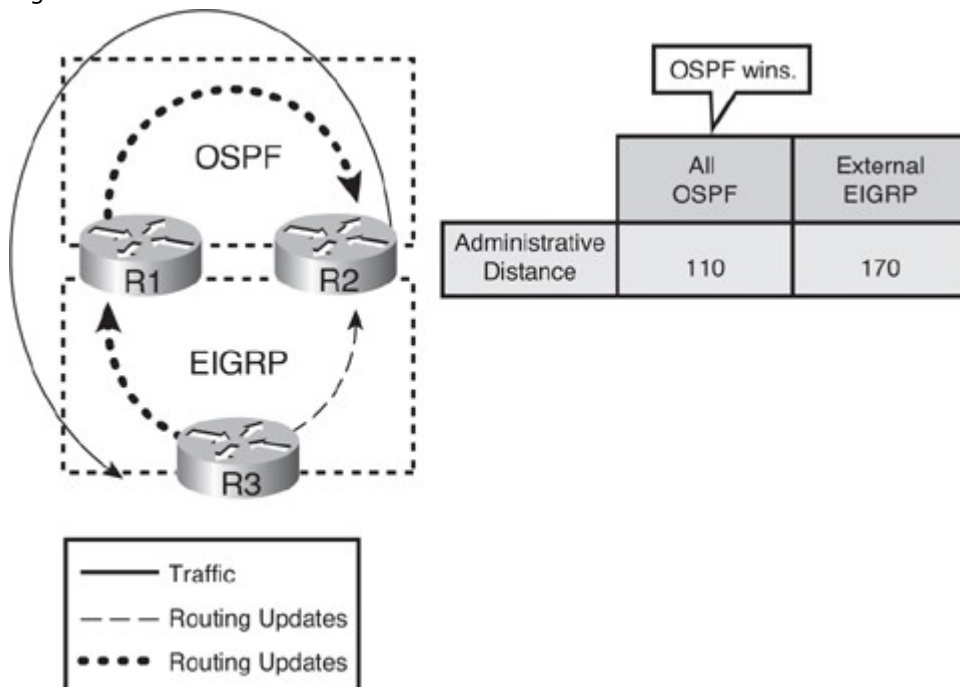
1 10.2.7.1 4 msec 0 msec 0 msec

2 192.168.2.2 16 msec \* 12 msec

R2#

Example 4-46 compares R1's and R2's routing table entry for the route to the R3 loopback network. Notice the administrative distance of the routes. R2 learns the route from R1 with an administrative distance of 110 (from OSPF). R2 also learns the route from R3, just as R1 does, with an administrative distance of 170 (from external EIGRP). R2 chooses the route with the lowest administrative distance, and therefore chooses the route via R1. This process is illustrated in Figure 4-33.

Figure 4-33. R2 Chooses the Route with the Lowest Administrative Distance.



Example 4-46. R1's and R2's Routing Table Entry for R3's Network

```
R1#show ip route | section 254
D EX 192.168.254.0 255.255.255.0
 [170/2297856] via 192.168.2.2, 00:18:07, Serial0/0/0
R1#

R2#show ip route | section 254
O E2 192.168.254.0 255.255.255.0
 [110/4] via 10.2.7.1, 00:17:32, FastEthernet0/1.1
R2#
```

To fix this suboptimal routing, the administrative distance of the external routes in the OSPF network is changed to something higher than the administrative distance of EIGRP external routes (170); 171 is used here. R2's configuration is changed first, and before changing the configuration, debugging is enabled so that the process used by the router can be examined. Example 4-47 illustrates the resulting output when the change is made on R2. Notice the routes being deleted and then be added again because of the smaller administrative distance.

Example 4-47. Changing the Administrative Distance of External OSPF Routes on R2

```
Code View: Scroll / Show All
R2#debug ip routing
IP routing debugging is on
R2#
R2#config t
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#router ospf 1
R2(config-router)#distance ospf external 171
R2(config-router)#
lw2d: RT: delete route to 10.1.1.0 255.255.255.0
lw2d: RT: NET-RED 10.1.1.0 255.255.255.0
lw2d: RT: delete route to 10.0.7.1 255.255.255.255
lw2d: RT: NET-RED 10.0.7.1 255.255.255.255
lw2d: RT: delete route to 192.168.254.0 255.255.255.0
lw2d: RT: NET-RED 192.168.254.0 255.255.255.0
lw2d: RT: delete route to 192.168.253.0 255.255.255.0
lw2d: RT: NET-RED 192.168.253.0 255.255.255.0
lw2d: RT: SET_LAST_RDB for 10.0.7.1 255.255.255.255
 NEW rdb: via 10.2.7.1

lw2d: RT: add 10.0.7.1 255.255.255.255 via 10.2.7.1, ospf metric [110/2]
lw2d: RT: NET-RED 10.0.7.1 255.255.255.255
lw2d: RT: SET_LAST_RDB for 10.1.1.0 255.255.255.0
 NEW rdb: via 10.2.7.1

lw2d: RT: add 10.1.1.0 255.255.255.0 via 10.2.7.1, ospf metric [171/4]
lw2d: RT: NET-RED 10.1.1.0 255.255.255.0
lw2d: RT: SET_LAST_RDB for 192.168.253.0 255.255.255.0
 NEW rdb: via 10.2.7.1

lw2d: RT: add 192.168.253.0 255.255.255.0 via 10.2.7.1, ospf metric [171/4]
lw2d: RT: NET-RED 192.168.253.0 255.255.255.0
```



```
lw2d: RT: SET_LAST_RDB for 192.168.254.0 255.255.255.0
 NEW rdb: via 10.2.7.1

lw2d: RT: add 192.168.254.0 255.255.255.0 via 10.2.7.1, ospf metric [171/4]
lw2d: RT: NET-RED 192.168.253.0 255.255.255.0
lw2d: RT: closer admin distance for 192.168.254.0, flushing 1 routes
lw2d: RT: NET-RED 192.168.254.0 255.255.255.0
lw2d: RT: SET_LAST_RDB for 192.168.254.0 255.255.255.0
 NEW rdb: via 192.168.1.2
```

Example 4-48 illustrates the resulting routing table entry on R2; R2 is now learning directly from R3. This is confirmed by the trace output shown in Example 4-49.

Example 4-48. R2's Routing Table Entry After the Administrative Distance Changed

```
R2(config-router)#do show ip route | section 254
D EX 192.168.254.0 255.255.255.0
 [170/2297856] via 192.168.1.2, 00:00:51, Serial0/0/0
R2(config-router)#
```

Example 4-49. R2 Now Goes Directly to R3

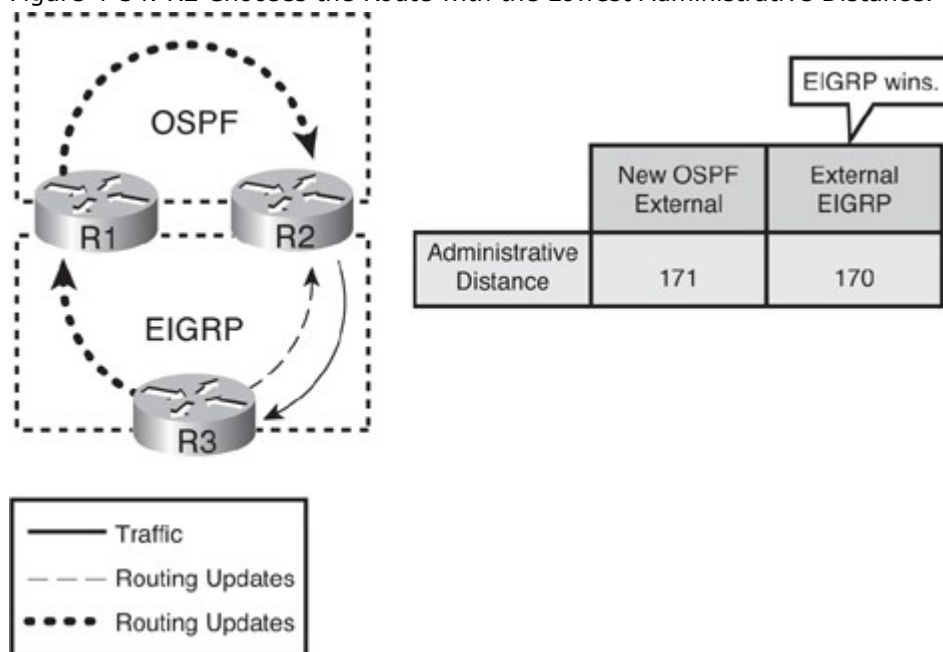
```
R2(config-router)#do trace
192.168.254.1

Type escape sequence to abort.
Tracing the route to 192.168.254.1

 1 192.168.1.2 12 msec * 12 msec
R2(config-router)#
```

Figure 4-34 shows the resulting traffic flow.

Figure 4-34. R2 Chooses the Route with the Lowest Administrative Distance.



The same configuration is also needed on R1. The configuration and resulting trace output is shown in Example 4-50.

Example 4-50. Configuration on R1 to Change the Administrative Distance

```
R1(config)#router ospf 1
R1(config-router)#distance ospf external 171
R1(config-router)#do trace 192.168.254.1
```

Type escape sequence to abort.

Tracing the route to 192.168.254.1

```
 1 192.168.2.2 16 msec * 12 msec
```

```
R1(config-router)#
```

This comprehensive example illustrated how routing updates can be controlled, using multiple methods, including prefix lists, route tagging, route filtering, and changing the administrative distance.

## Summary

In this chapter, you learned about manipulating routing updates. The chapter focused on the following topics:

- Network performance issues, including excessive routing updates, the presence of any route maps or filters, and the number of routing protocols running in the same autonomous system. Solutions to these issues include design changes, using passive interfaces, and route filtering (access lists, route maps, distribute lists, and prefix lists).
- Reasons for using more than one routing protocol (migration, host system needs, mixed-vendor environment, political and geographical borders, MPLS VPNs), how routing information can be exchanged between them (referred to as redistribution), and how Cisco routers operate in a multiple routing protocol environment.
- How route redistribution is always performed *outbound*. The router doing redistribution does not change its routing table.
- Issues arising when redistributing routes, including routing loops, incompatible routing information, and inconsistent convergence times.
- The roles that the administrative distance and the routing metric play in route selection. Lower administrative distances are considered more believable (better). Lower metrics are considered

better, within the same routing protocol. When redistributing, a router assigns a seed metric to redistributed routes, using the **default-metric** router configuration command, or specified as part of the **redistribute** command either with the **metric** option or by using a route map.

- The redistribution techniques, one-point and multipoint:
  - The two methods of one-point route redistribution: one way and two way. Suboptimal routing is a possible issue with these techniques.
  - The two methods of multipoint route redistribution: one way and two way. Multipoint redistribution is likely to introduce potential routing loops.
  - To prevent routing issues use one of the following options: Redistribute a default route from the core autonomous system into the edge autonomous system, and redistribute routes from the edge routing protocols into the core routing protocol; redistribute multiple static routes about the core autonomous system networks into the edge autonomous system, and redistribute routes from the edge routing protocols into the core routing protocol; redistribute routes from the core autonomous system into the edge autonomous system with filtering to block out inappropriate routes; redistribute all routes from the core autonomous system into the edge autonomous system, and from the edge autonomous system into the core autonomous system, and then modify the administrative distance associated with redistributed routes so that they are not the selected routes when multiple routes exist for the same destination.
- Configuration of redistribution between various IP routing protocols:
  - To redistribute into RIP, use the **redistribute protocol [process-id] [match route-type] [metric metric-value] [route-map map-tag]** router configuration command.
  - To redistribute into OSPF, use the **redistribute protocol [process-id] [metric metric-value] [metric-type type-value] [route-map map-tag] [subnets] [tag tag-value]** router configuration command.
  - To redistribute into EIGRP, use the **redistribute protocol [process-id] [match route-type] [metric metric-value] [route-map map-tag]** router configuration command.
- Using the **passive-interface type number [default]** router configuration command to prevent a routing protocol's routing updates from being sent through the specified router interface.
- How to manipulate the administrative distance of routes to influence the route-selection process:
  - Use the **distance administrative-distance [address wildcard-mask [ip-standard- list] [ip-extended-list]]** router configuration command for all protocols.
  - Alternatively, for EIGRP, use the **distance eigrp internal-distance external-distance** router configuration command.
  - Alternatively for OSPF, use the **distance ospf {[intra-area dist1] [inter-area dist2] [external dist3]}** router configuration command.
  - Alternatively, for BGP, use the **distance bgp external-distance internal-distance local-distance** router configuration command.
- Using the **show ip route [ip-address]** and **traceroute [ip-address]** commands to verify route redistribution.
- Using the **ip route 0.0.0.0 0.0.0.0 interface** or **ip default-network network-number** global configuration command to configure default routes.
- Using route maps for route filtering during redistribution, PBR, NAT, and BGP.
- The characteristics of route maps, configured using the **route-map map-tag [permit | deny] [sequence-number]** global configuration command:
  - Route maps allow some conditions to be tested against the packet or route in question using **match** commands. If the conditions match, some actions can be taken to modify attributes of the packet or route; these actions are specified by **set** commands.
  - A collection of **route-map** statements that have the same route map name is considered one route map.
  - Within a route map, each **route-map** statement is numbered and therefore can be edited individually.
  - The default for the **route-map** command is **permit**, with a *sequence-number* of 10.
  - Only one condition listed on the same **match** statement must match for the entire statement to be considered a match. However, all **match** statements within a **route-map** statement must match for the route map to be considered matched.

- When used with a **redistribute** command, a **route-map** statement with **permit** indicates that the matched route is to be redistributed, while a **route-map** statement with **deny** indicates that the matched route is not to be redistributed.
- Configuring route maps for PBR, using the **ip policy route-map map-tag** interface configuration command.
- Table 4-15 lists some of the variety of match criteria that can be defined. Table 4-16 lists some of the variety of set commands that are available.
- Distribute lists, allowing an access list to be applied to routing updates:
  - The **distribute-list {access-list-number | name} out [interface-name | routing-process [routing-process parameter]]** router configuration command assigns the access list to filter outgoing routing updates. This command filters updates going *out of* the interface or routing protocol specified in the command, *into* the routing process under which it is configured.
  - The **distribute-list {access-list-number | name} [route-map map-tag] in [interface-type interface-number]** router configuration command assigns the access list to filter routing updates coming in through an interface. This command filters updates going *into* the interface specified in the command, *into* the routing process under which it is configured.
- Prefix lists, an alternative to distribute lists, with improvements in performance, support for incremental modifications, a more user-friendly command-line interface, and greater flexibility. Prefix lists are configured with the **ip prefix-list {list-name | list-number} [seq seq-value] {deny | permit} network/length [ge ge-value] [le le-value]** global configuration command. The *ge-value* and *le-value* range is  $length < ge-value < le-value \leq 32$ . An exact match is assumed when neither **ge** nor **le** is specified.
- Whether a prefix in a prefix list is permitted or denied is based on the following rules:
  - An empty prefix list permits all prefixes.
  - If a prefix is permitted, the route is used. If a prefix is denied, the route is not used.
  - Prefix lists consist of statements with sequence numbers. The router begins the search for a match at the top of the prefix list, which is the statement with the lowest sequence number.
  - When a match occurs, the router does not need to go through the rest of the prefix list. For efficiency, you might want to put the most common matches (permits or denies) near the top of the list by specifying a lower sequence number.
  - An implicit **deny** is assumed if a given prefix does not match any entries in a prefix list.
- Prefix list sequence numbers:
  - Sequence numbers are generated automatically, unless you disable this automatic generation.
  - A prefix list is an ordered list. The sequence number is significant when a given prefix is matched by multiple entries of a prefix list, in which case the one with the smallest sequence number is considered the real match.
  - The evaluation of a prefix list starts with the lowest sequence number and continues down the list until a match is found, in which case the **permit** or **deny** statement is applied to that network and the remainder of the list is not evaluated.
- Verifying prefix lists can involve commands listed in Table 4-20.
- How multiple methods can be combined to effectively control routing updates.

## References

For additional information, see these resources:

- "Cisco IOS Software Releases 12.4 Mainline" support page: [http://www.cisco.com/en/US/products/ps6350/tsd\\_products\\_support\\_series\\_home.html](http://www.cisco.com/en/US/products/ps6350/tsd_products_support_series_home.html)
- The Cisco IOS IP Routing Protocols Command Reference at [http://www.cisco.com/en/US/docs/ios/iproute/command/reference/irp\\_book.html](http://www.cisco.com/en/US/docs/ios/iproute/command/reference/irp_book.html)

## Review Questions

Answer the following questions, and then see Appendix A, "Answers to Review Questions," for the answers.

1. List common network performance issues related to routing.
2. What methods can be used to filter routes?

3. List some reasons why you might use multiple routing protocols in a network.
4. What is redistribution?
5. How does redistributing between two routing protocols change the routing table on the router that is doing the redistribution?
6. What are some issues that arise with redistribution?
7. What may be the cause of a routing loop in a network that has redundant paths between two routing processes?
8. What two parameters do routers use to select the best path when they learn two or more routes to the same destination from different routing protocols?
9. Fill in the default administrative distances for the following routing protocols.
 

| Routing Protocols                  | Default Administrative Distance Value |
|------------------------------------|---------------------------------------|
| Connected interface                |                                       |
| Static route out an interface      |                                       |
| Static route to a next-hop address |                                       |
| EIGRP summary route                |                                       |
| External BGP                       |                                       |
| Internal EIGRP                     |                                       |
| IGRP                               |                                       |
| OSPF                               |                                       |
| IS-IS                              |                                       |
| RIPv1 and RIPv2                    |                                       |
| EGP                                |                                       |
| ODR                                |                                       |
| External EIGRP                     |                                       |
| Internal BGP                       |                                       |
| Unknown                            |                                       |
10. When configuring a default metric for redistributed routes, should the metric be set to a value *larger* or *smaller* than the largest metric within the receiving autonomous system?
11. Fill in the default seed metrics for the following protocols.
 

| Protocol That the Route Is Redistributed Into | Default Seed Metric |
|-----------------------------------------------|---------------------|
| RIP                                           |                     |
| IGRP/EIGRP                                    |                     |
| OSPF                                          |                     |
| IS-IS                                         |                     |
| BGP                                           |                     |
12. What is the safest way to perform redistribution between two routing protocols?
13. Can redistribution be configured between IPv6 RIPv2 and IP RIP? Between IPv6 EIGRP and IP EIGRP? Between IP EIGRP and OSPF?
14. When configuring redistribution into RIP, what is the *metric-value* parameter?
15. Router A is running RIPv2 and OSPF. In the RIPv2 domain, it learns about the 10.1.0.0/16 and

10.3.0.0/16 routes. In the OSPF domain, it learns about the 10.5.0.0/16 and 172.16.1.0/24 routes. What is the result of the following configuration on Router A?

```
router ospf 1
 redistribute rip metric 20
```

16. What are the five components of the EIGRP routing metric?
17. What happens if you use the **metric** parameter in a **redistribute** command and you use the **default-metric** command?
18. What does the **passive-interface default** command do?
19. Suppose you have a dialup WAN connection between site A and site B. What can you do to prevent excess routing update traffic from crossing the link but still have the boundary routers know the networks that are at the remote sites?
20. What does the following command do?  
distance 150 0.0.0.0 255.255.255.255 3
21. What are some applications of route maps?
22. A **route-map** statement has multiple **match** commands. How many of these **match** statements must be considered true for the **route-map** statement to be considered matched?
23. A single **match** statement has multiple conditions. How many of these conditions in the **match** statement must be true for that **match** statement to be considered a match?
24. What is the *map-tag* parameter in a **route-map** command?
25. What command is used to define a route map for policy-based routing?
26. What commands would be used to configure the use of a route map called TESTING when redistributing OSPF 10 traffic into RIP?
27. What does the following configuration do?  
route-map test deny 10  
match tag 80  
route-map test permit 20  
router rip  
redistribute ospf 10 route-map test
28. A distribute list allows routing updates to be filtered based on what?
29. What is the difference between the **distribute-list out** and **distribute-list in** commands?
30. What command is used to configure filtering of the routing update traffic from an interface? At what prompt is this command entered?
31. In a prefix list, what does **permit** mean, and what does **deny** mean?
32. When configuring a prefix list, what is formula that defines the range of the *length*, *le-value*, and *ge-value* parameters?
33. What is the difference in the default values of the sequence number parameter when configuring a prefix list versus a route map?
34. What command can be used to discover the path that a packet takes through a network?

## Chapter 5. Implementing Path Control

This chapter discusses. It covers the following topics:

- Understanding Path Control
- Implementing Path Control Using Offset Lists
- Implementing Path Control Using Cisco IOS IP SLAs
- Implementing Path Control Using Policy-Based Routing

- Advanced Path Control Tools

This chapter starts by discussing path control fundamentals. Three tools for path control are detailed: offset lists, Cisco IOS IP service level agreements (SLAs), and policy-based routing (PBR). The chapter concludes with a discussion of advanced path control tools.

#### Understanding Path Control

This section introduces path control performance issues and introduces the tools available to control path selection.

#### Assessing Path Control Network Performance

This chapter is concerned with controlling the path that traffic takes through a network. In some cases, there might be only one way for traffic to go. However, many networks include redundant paths, by having redundant devices or redundant links. In these cases, the network administrator may want to control which way certain traffic flows.

The choice of routing protocol or routing protocols used in a network is one factor in defining how paths are selected; for example, different administrative distances, metrics, and convergence times may result in different paths being selected. As described in Chapter 4, "Manipulating Routing Updates," when multiple routing protocols are implemented, inefficient routing may result. For example, two-way multipoint redistribution requires careful planning and implementation to ensure that traffic travels the optimal way, and that there are no routing loops.

When a network includes redundancy, other considerations include the following:

- **Resiliency**— Having redundancy does not guarantee resiliency, the ability to maintain an acceptable level of service when faults occur. For example, having redundant links between two sites does not automatically result in the backup link being used if the primary link fails. Configuration is necessary to implement failover, and to use the backup link for load sharing if that is desired. (Even if failover is configured correctly, the redundant link may not operate when needed; for example, if it uses the same physical infrastructure as the primary link.)
- **Availability**— The time required for a routing protocol to learn about a backup path when a primary link fails is the *convergence time*. If the convergence time is relatively long, some applications may time out. Thus, using a fast-converging routing protocol, and tuning parameters to ensure that it does converge fast, is crucial for high-availability networks.
- **Adaptability**— The network can also be configured to adapt to changing conditions. For example, a redundant path could be brought up and used when the primary path becomes congested, not just when it fails.
- **Performance**— Network performance can be improved by tuning routers to load share across multiple links, making more efficient use of the bandwidth. For example, route advertisements for specific prefixes can be advertised on one link to change the balance of bandwidth use relative to other links.
- **Support for network and application services**— More advanced path control solutions involve adjusting routing for specific services, such as security, optimization, and quality of service (QoS). For example, to optimize traffic via a Cisco Wide Area Application Services (WAAS) Central Manager, traffic must be directed to flow through the Cisco WAAS device.

#### Note

Cisco WAAS is a WAN optimization and application acceleration solution that optimizes application and video delivery over a WAN, and is illustrated briefly in the "Cisco Wide Area Application Services" section, later in this chapter.

- **Predictability**— The path control solution implemented should derive from an overall strategy, so that the results are deterministic and predictable. For example, traffic is bidirectional by nature; for every packet that goes out, a reply typically must come back. When configuring a routing protocol to deploy a path control strategy, consider both upstream and downstream traffic. For example, changing or tuning downstream advertisements toward a server farm could adversely affect upstream traffic flows from the server farm.
- **Asymmetric traffic**— Asymmetric traffic, traffic that flows one on path in one direction and on a different path in the opposite direction, occurs in many networks that have redundant paths. Asymmetry, far from being a negative trait, is often *desirable* network trait, because it uses available bandwidth effectively, such as on an Internet connection on which downstream traffic may require higher bandwidth than upstream traffic. Border Gateway Protocol (BGP) includes a good set of tools



to control traffic in both directions on an Internet connection. However, in most routing protocols, there are no specific tools to control traffic direction.

In a part of a network that includes devices or services such as stateful firewalls, Network Address Translation (NAT) devices, and voice traffic, which require symmetrical routing, traffic symmetry must be enforced or the services must be tuned to accommodate asymmetry. For example, asymmetry in voice networks may introduce jitter and QoS issues. In other areas of the network, though, it might be inefficient and undesirable to try to engineer artificial symmetry.

Optimal routing in terms of network utilization within specific requirements is typically a design goal. Those requirements should be considered within the context of the applications in use, the user experience, and a comprehensive set of performance parameters. These parameters include delay, bandwidth utilization, jitter, availability, and overall application performance. Even if the routing table on the routers includes the necessary prefixes, applications might still fail if the performance requirements are not met.

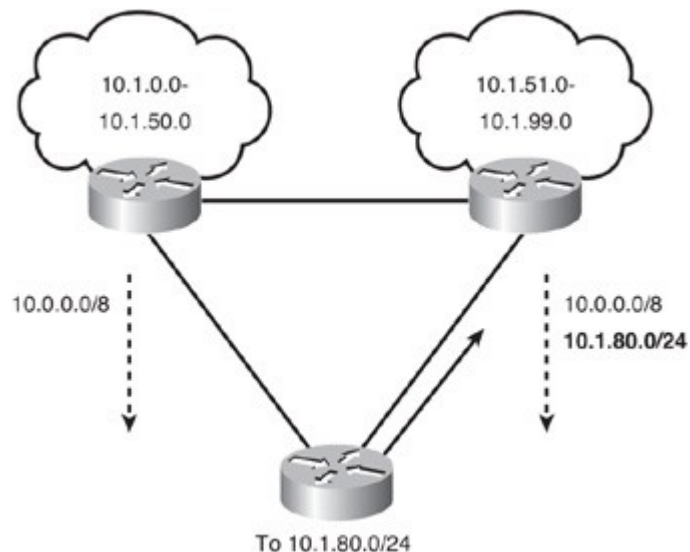
#### Path Control Tools

Unfortunately there is not a “one-command” solution to implement path control. Instead, many tools are available.

Path control tools include the following:

- A good addressing design: A good design should include summarizable address blocks and classless interdomain routing (CIDR) that align with the physical topology. These aspects are key to a stable network. As discussed in Chapter 1, “Routing Services,” summarization hides addressing details, isolates routing issues, and defines failure domains. Controlling summarization in strategic areas of the network affects path control. For example, in the network in Figure 5-1, the 10.0.0.0/8 summary is advertised from both routers, and the more specific route for 10.1.80.0/24 is advertised from the router on the right, providing direct access to that subnet. The resulting traffic flows are deterministic and more resilient.

Figure 5-1. Advertising Summaries and More-Specific Routes Affects Traffic Flow.



- Redistribution and other routing protocol characteristics— The capabilities of the routing protocol used can help implement a path control strategy more effectively, as summarized in Table 5-1. For example, Enhanced Interior Gateway Routing Protocol (EIGRP) automatically summarizes on network boundaries, and Open Shortest Path First (OSPF) can summarize only on Area Border Routers (ABRs) and Autonomous System Boundary Routers (ASBRs). Metrics can be changed and external routes can be tagged during redistribution between protocols. When multiple routing protocols are used, routes must be redistributed between them carefully, as detailed in Chapter 4.

Table 5-1. Routing Protocol Characteristics

| Characteristic | OSPF                                                          | EIGRP                                  |
|----------------|---------------------------------------------------------------|----------------------------------------|
| Route marking  | Tags for external routes can be added at distribution points. | Tags for all routes can be configured. |

| Characteristic              | OSPF                                                         | EIGRP                                                                             |
|-----------------------------|--------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Metric                      | Can be changed for external routes at redistribution points. | Can be set using route maps.                                                      |
| Next hop                    | Can be changed for external routes at redistribution points. | Can be set for all routes under various conditions.                               |
| Filtering                   | Summary information can be filtered at ABRs and ASBRs.       | Can be configured anywhere for any routes.                                        |
| Route summarization         | Can be configured only on ABRs and ASBRs.                    | Can be configured anywhere for any routes. Autosummarization is on by default.[1] |
| Unequal-cost load balancing | Not available.                                               | Available, with <b>variance</b> command.                                          |

[1] As mentioned in Chapter 1, the Cisco IOS documentation for EIGRP says that automatic summarization is now disabled by default. However, testing has confirmed it is still on, at least in some versions of the Cisco IOS. Thus, it would be prudent to confirm the autosummary configuration or to configure it explicitly.

- Passive interfaces— As also described in Chapter 4, passive interfaces prevent a routing protocol's routing updates from being sent through the specified router interface.

Other tools include the following:

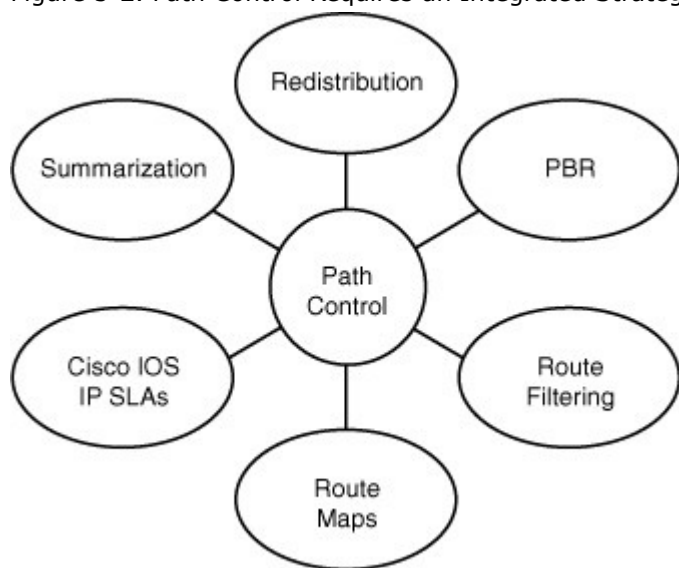
- Distribute lists
- Prefix lists
- Administrative distance
- Route maps
- Route tagging
- Offset lists
- Cisco IOS IP SLAs
- PBR

The first five of these tools were covered in Chapter 4; the others are the focus of the rest of this chapter.  
Note

Three other tools are covered in the "Advanced Path Control Tools" section, at the end of the chapter.

You can use all of these tools as part of an integrated strategy to implement path control, as illustrated in Figure 5-2. It is important to have a strategy before implementing specific path control tools and technologies.

Figure 5-2. Path Control Requires an Integrated Strategy.



For example, filters allow specific control of routing updates and provide security mechanisms to hide specific destinations. In contrast, PBR can bypass the routing table and define a path based on static or dynamic information, forcing traffic to specific destinations such as security appliances, NAT devices, and WAN optimization elements.

As another example, by controlling and filtering routing updates in one direction, you can affect traffic flowing in the opposite direction and prevent that traffic from reaching those destinations

By tagging routes by using route maps, you can define priorities for specific destinations along multiple paths, allowing those paths to be used in a deterministic order. For example, on an Internet connection when multiple exit points exist out of a network, route maps can be used to tag and define priorities for specific destinations.

#### Implementing Path Control Using Offset Lists

This section introduces offset lists and how to configure and verify path control using offset lists.

##### Using Offset Lists to Control Path Selection

An offset list is the mechanism for increasing incoming and outgoing metrics to routes learned via EIGRP or Routing Information Protocol (RIP). (Offset lists are only used for distance vector routing protocols.) Optionally, an offset list can be limited by specifying either an access list or an interface.

##### Configuring Path Control Using Offset Lists

To add an offset to incoming and outgoing metrics to routes learned via EIGRP or RIP, use the offset-list {access-list-number | access-list-name} {in |out} offset [interface-type interface-number] router configuration command, as explained in Table 5-2.

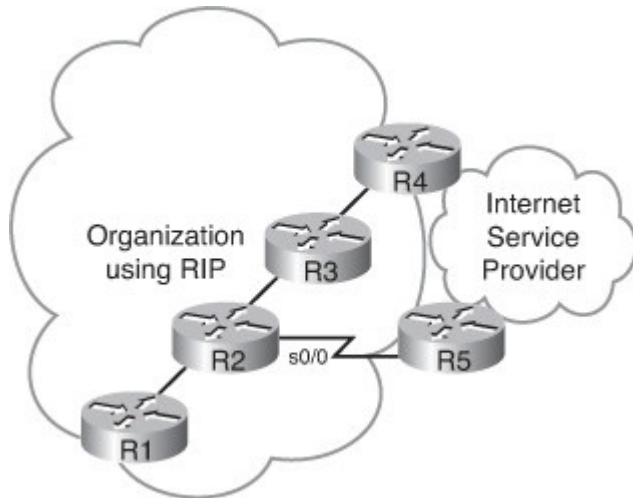
Table 5-2. *offset-list* Command

| Parameter                                           | Description                                                                                                                                              |
|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>access-list-number</i>   <i>access-list-name</i> | Standard access list number or name to be applied. Access list number 0 indicates all access lists. If the <i>offset</i> value is 0, no action is taken. |
| <i>in</i>                                           | Applies the access list to incoming metrics.                                                                                                             |
| <i>out</i>                                          | Applies the access list to outgoing metrics.                                                                                                             |
| <i>offset</i>                                       | Positive offset to be applied to metrics for networks matching the access list. If the offset is 0, no action is taken.                                  |
| <i>interface-type interface-number</i>              | (Optional) Interface type and number to which the offset list is applied.                                                                                |

The offset value is added to the routing metric. An offset list that specifies an interface type and interface number is considered to be an extended list and takes precedence over an offset list that is not extended. Therefore, if an entry passes the extended offset list and a normal offset list, the offset of the extended offset list is added to the metric.

Figure 5-3 illustrates an example network in which an organization is using RIP and is connected to the Internet service provider (ISP) via edge Routers R4 and R5. A subset of routes is received from each of the edge routers. The metric between Routers R2 and R5 is smaller than the metric between Routers R2 and R4, because it is only one hop. However, this is very slow link. An offset list can be used on Router R2 so that it prefers the path toward the edge Router R4 for a specific set of destinations.

Figure 5-3. An Offset List Can Be Used to Prefer a Faster Path.



A partial configuration of Router R2 is shown in Example 5-1. In this example, the offset-list 21 in 2 serial 0/0 command adds an offset of 2 to the metric of routes learned from interface serial 0/0 (connected to Router R5) that are permitted by access list 21. Access list 21 permits a specific set of routes (any in the 172.16.0.0/16 network) being learned from Router R5. This command is entered in RIP configuration mode on Router R2. This configuration results in the path toward Router R4 being considered better for the set of selected routes; R4 becomes the preferred way out toward the ISP for these routes.

Example 5-1. Offset List Configuration for Router R2 in Figure 5-3

```
router rip
 offset-list 21 in 2 serial 0/0
!
access-list 21 permit 172.16.0.0 0.0.255.255
```

#### Verifying Path Control Using Offset Lists

You can use the **tracert** EXEC to verify that an offset list is affecting the path that traffic takes.

The routing table, viewed with the **show ip route** command, identifies the metrics for learned routes. You should compare these metrics to what was expected by the offset list configuration. For EIGRP, the EIGRP topology table can be examined using the **show ip eigrp topology** command. The topology table contains all routes learned from the router's EIGRP neighbors, and includes the metric information for those routes, including the best route and any other feasible routes that the router has learned about.

#### Note

Recall that only successor and feasible successor routes are displayed with the **show ip eigrp topology** command. Add the **all-links** keyword to display all routes, including those not eligible to be successor or feasible successor routes.

You can use **debug** commands, such as **debug ip rip** and **debug ip eigrp**, to view the real-time processing of incoming and outgoing RIP routing updates, to ensure that the metric is being processed appropriately.

#### Caution

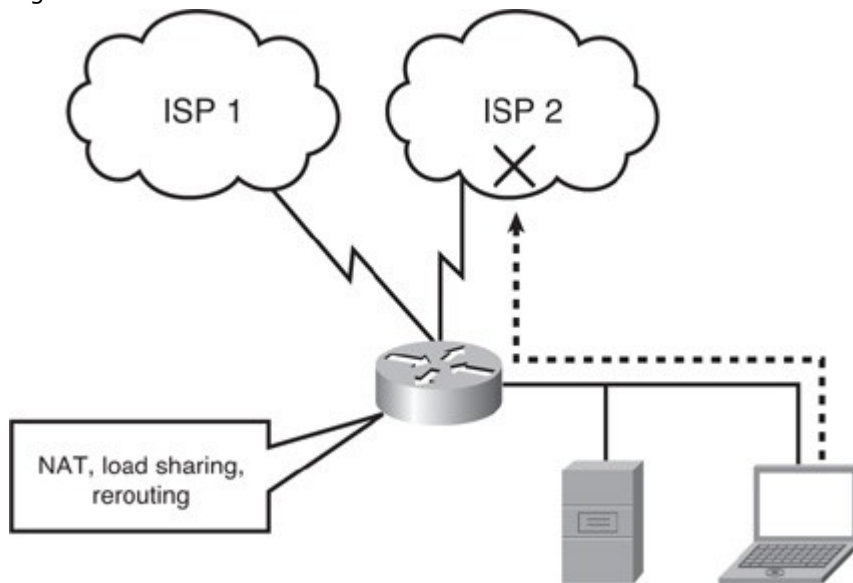
Use caution when executing **debug** commands because they may consume a lot of router resources and could cause problems in a busy production network. Debugging output takes priority over other network

traffic; too much debug output might severely reduce the performance of the router or even render it unusable in the worst case.

#### Implementing Path Control Using Cisco IOS IP SLAs

This section examines path control using Cisco IOS IP SLAs. A typical scenario for this solution is Internet branch office connectivity, with connections to two different ISPs, such as the network illustrated in Figure 5-4. In this case, the organization's edge router is configured to perform NAT, and has default routes for outbound traffic to the ISPs; branch offices, especially smaller ones, are not likely to run BGP or other routing protocols toward the ISP. The static default routes are likely to be equal cost, and the Cisco IOS will by default load balance over the links on a per-destination basis. NAT will be applied to the outbound traffic resulting from the load-balancing algorithm.

Figure 5-4. A Branch Office Scenario.



In this scenario, the edge router can detect if there is a direct failure on the link to one ISP, and in that case use the other ISP for all traffic. However, if the infrastructure within of one of the ISPs fails and the link to that ISP remains up, the edge router would continue to use that link because the static default route would still be valid.

There are multiple solutions to this issue. One approach is for the branch office router to run a dynamic routing protocol with the ISPs, so that the branch router learns the ISPs' networks in its routing table. The branch router will then be aware of any link failures within the ISPs' network. This solution is impractical for smaller branch offices, and in any case requires interaction and integration with the ISPs. It may, however, be the best solution for critical branch offices or those with large traffic volumes.

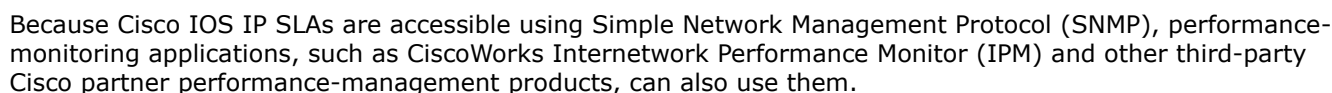
Another solution is to use either static routes or PBR, but make them subject to reachability tests toward critical destinations, such as the Domain Name System (DNS) servers within the ISP. If the DNS servers in one of the ISPs go down or are unreachable, the static route toward that ISP would be removed. These reachability tests can be performed with Cisco IOS IP SLAs that probe the DNS servers frequently and that are attached to the static routes.

The tools used for this solution include the following:

- **Object tracking**— The Cisco IOS object tracking tracks the reachability of specified objects (in this example, of DNS servers).
- **Cisco IOS IP SLAs probes**— The object tracking features can use Cisco IOS IP SLAs to send different types of probes toward the desired objects.
- **Route maps with PBR**— To associate the results of the tracking to the routing process, PBR with route maps can be used, allowing options to define specific traffic classes, such as voice, or specific applications.
- **Static routes with tracking options**— As an alternative to PBR, you can use static routes with tracking options. This solution is simpler and accommodates scenarios in which you want all outbound traffic to choose outbound exit points similarly.

This section introduces Cisco IOS IP SLAs and describes how this feature is used to control path selection. Cisco IOS IP SLAs use active traffic monitoring, generating traffic in a continuous, reliable, and predictable manner, to measure network performance.

Figure 5-5. Cisco IOS IP SLAs Measure Network Performance.



For information about SNMP operation, see the SNMP chapter of Cisco's Internetworking Technology Handbook, available at <http://www.cisco.com/en/US/docs/internetworking/technology/handbook/SNMP.html>.

As an additional feature, SNMP notifications based on the data gathered by a Cisco IOS IP SLAs operation allow the router to receive alerts when performance drops below a specified level and when problems are corrected. These thresholds can trigger additional events and actions.

## Cisco IOS IP SLAs Operation

**<http://www.ccnproute.net>**

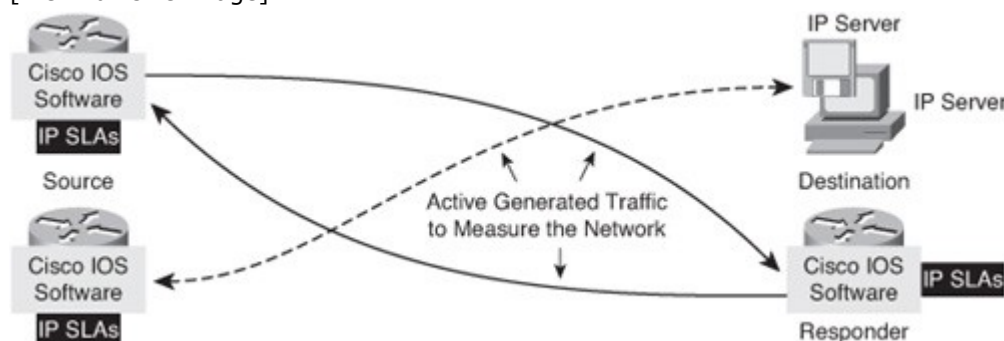
proactively identify network issues, and react to performance metrics with changes to the configuration and network.

The Cisco IOS IP SLAs feature allows performance measurements to be taken within and between Cisco devices, or between a Cisco device and a host, providing data about service levels for IP applications and services.

Cisco IOS IP SLAs measurements perform active monitoring by generating and analyzing traffic to measure performance between Cisco IOS Software devices or between a Cisco IOS device and a host, such as a network application server. With the IOS IP SLAs feature enabled, a router sends synthetic traffic to the other device, as illustrated in Figure 5-6.

Figure 5-6. IP SLAs Take Measurements Between a Cisco Device and Another Cisco Device or a Host.

[View full size image]



#### Cisco IOS IP SLAs Sources and Responders

All the IP SLAs measurement probe operations are configured on the IP SLAs *source*, either via the command-line interface (CLI) or through an SNMP tool that supports the operation of IP SLAs. The source sends probe packets to the *target*.

There are two types of IP SLAs operations: those in which the target device is running the IP SLAs *responder* component (such as a Cisco router), and those in which the target device is not running the IP SLAs responder component (such as a web server or IP host). An IP SLAs responder is a component embedded in a Cisco IOS device that allows that device to anticipate and respond to IP SLAs request packets. A Cisco IOS device can be configured as an IP SLAs responder and will provide accurate measurements without the need for dedicated probes or any complex or per-operation configuration.

The IP SLAs measurement accuracy is improved when the target is an IP SLAs responder, as described in the upcoming "Cisco IOS IP SLAs Operation with Responders" section.

#### Cisco IOS IP SLAs Operations

An *IP SLAs operation* is a measurement that includes protocol, frequency, traps, and thresholds.

The network manager configures the IP SLAs source with the target device address, protocol, and User Datagram Protocol (UDP) or Transfer Control Protocol (TCP) port number, for each operation. When the operation is finished and the response has been received, the results are stored in the IP SLAs MIB on the source, and are retrieved using SNMP.

IP SLAs operations are specific to target devices. Operations such as DNS or HTTP can be sent to any suitable computer. For operations such as testing the port used by a database, there might be risks associated with unexpected effects on actual database servers, and therefore IP SLAs responder functionality on a router can be configured to respond in place of the actual database server.

#### Cisco IOS IP SLAs Operation with Responders

Using an IP SLAs responder provides enhanced measurement accuracy—without the need for dedicated third-party external probe devices—and additional statistics that are not otherwise available via standard Internet Control Message Protocol (ICMP)-based measurements.

When a network manager configures an IP SLAs operation on the IP SLAs source, reaction conditions can also be defined, and the operation can be scheduled to be run for a period of time to gather statistics. The source uses the IP SLAs control protocol to communicate with the responder before sending test packets. To increase security of IP SLAs control messages, message digest 5 (MD5) authentication can be used to secure the control protocol exchange.

The following sequence of events occurs for each IP SLAs operation that requires a responder on the target, as illustrated in Figure 5-7:



1. At the start of the control phase, the IP SLAs source sends a control message with the configured IP SLAs operation information to IP SLAs control port UDP 1967 on the target router (the responder). The control message includes the protocol, port number, and duration of the operation. In Figure 5-7, UDP port 2020 is used for the IP SLAs test packets.

If MD5 authentication is enabled, the MD5 checksum is sent with the control message, and the responder verifies the MD5 checksum. If the authentication fails, the responder returns an "authentication failure" message.

2. If the responder processes the control message, it sends an "OK" message to the source and listens on the port specified in the control message for a specified duration. If the responder cannot process the control message, it returns an error. If the IP SLAs source does not receive a response from the responder, it tries to retransmit the control message. It will eventually time out if it does not receive a response.

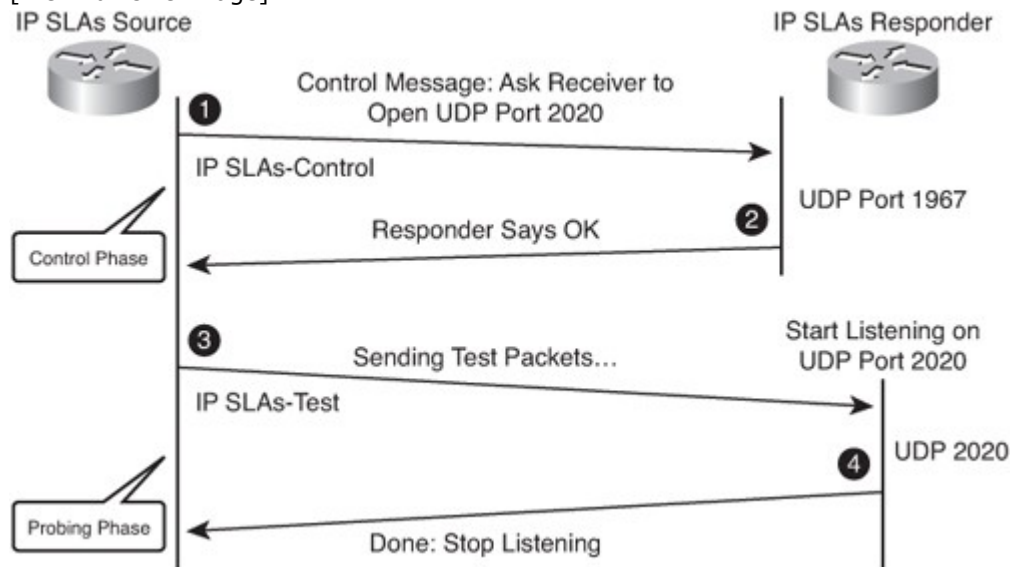
Note

The responder is capable of responding to multiple IP SLAs measurement operations that try to connect to the same port number.

3. If an "OK" message is returned, the IP SLAs operation on the source moves to the probing phase where it sends one or more test packets to the responder to compute response times. In Figure 5-7, the test messages are sent on control port 2020.
4. The responder accepts the test packets and responds. Based on the type of operation, the responder may add an "in" time stamp and an "out" time stamp in the response packet payload to account for the CPU time spent measuring unidirectional packet loss, latency, and jitter. These time stamps help the IP SLAs source make accurate assessments of one-way delay and processing time in target routers. The responder disables the user-specified port after it responds to the IP SLAs measurements packet or when the specified time expires.

Figure 5-7. IP SLAs Operation with a Responder.

[View full size image]

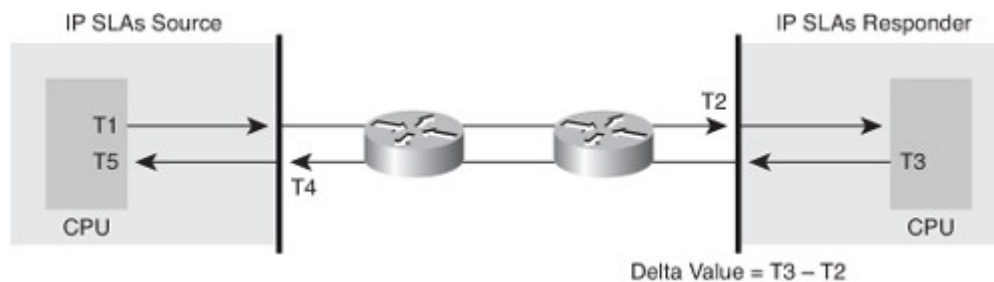


#### Cisco IOS IP SLAs with Responder Time Stamps

Figure 5-8 illustrates the use of time stamps in round-trip calculations in an operation using an IP SLAs responder. The IP SLAs source uses four time stamps for the round-trip time (RTT) calculation.

Figure 5-8. Time Stamps in an IP SLAs Operation with a Responder.

[View full size image]



The IP SLAs source sends a test packet at time T1.

Because of other high-priority processes, routers might take tens of milliseconds to process incoming packets. For example, the reply to a test packet might be sitting in a queue waiting to be processed. To account for this delay, the IP SLAs responder includes both the receipt time (T2) and the transmitted time (T3) in the response packet. The time stamps are accurate to submilliseconds.

The IP SLAs source subtracts T2 from T3 to determine the delta value—the time spent processing the test packet in the IP SLAs responder. The delta value is subtracted from the overall RTT.

The same principle is applied by IP SLAs source. The incoming time stamp (T4) is taken at the interrupt level to allow for greater accuracy in the RTT calculation. The T4 time stamp, rather than the T5 time stamp (when the packet is processed), is used in the RTT calculation.

The two time stamps taken in the IP SLAs responder also allow one-way delay, jitter, and directional packet loss to be tracked. These statistics are critical for understanding asynchronous network behavior. To calculate these one-way delay measurements, the source and target need to be synchronized to the same clock source, and therefore, the Network Time Protocol (NTP) must be configured on both.

#### Configuring Path Control Using IOS IP SLAs

This section describes some of the commands used to configure path control using IOS IP SLAs.

The following steps are required to configure Cisco IOS IP SLAs functionality:

- Step 1. Define one or more IP SLAs operations (or probes).
- Step 2. Define one or more tracking objects, to track the state of IOS IP SLAs operations.
- Step 3. Define the action associated with the tracking object.

These steps are detailed in the following sections.

#### Configuring Cisco IOS IP SLAs Operations

This section describes some of the configuration commands used to define IP SLAs operations.

Use the **ip sla operation-number** global configuration command (or the **ip sla monitor operation-number** global configuration command) to begin configuring a Cisco IOS IP SLAs operation and to enter IP SLA configuration mode (or rtr configuration mode). The *operation-number* is the identification number of the IP SLAs operation you want to configure.

#### Note

Effective with Cisco IOS Release 12.4(4)T, 12.2(33)SB, and 12.2(33)SXI, the **ip sla monitor** command is replaced by the **ip sla** command.

#### Note

From IP SLA configuration mode, a variety of commands can be entered, as shown here:

```
R1(config-ip-sla)#?
```

IP SLAs entry configuration commands:

```
dhcp DHCP Operation
dns DNS Query Operation
exit Exit Operation Configuration
frame-relay Frame-relay Operation
ftp FTP Operation
http HTTP Operation
icmp-echo ICMP Echo Operation
```

icmp-jitter ICMP Jitter Operation  
 path-echo Path Discovered ICMP Echo Operation  
 path-jitter Path Discovered ICMP Jitter Operation  
 slm SLM Operation  
 tcp-connect TCP Connect Operation  
 udp-echo UDP Echo Operation  
 udp-jitter UDP Jitter Operation  
 voip Voice Over IP Operation

R1(config-ip-sla)#

The ICMP echo operation is used to cause ICMP echo requests to be sent to a destination to check connectivity. Use the `icmp-echo {destination-ip-address | destination-hostname} [source-ip {ip-address | hostname} | source-interface interface-name]` IP SLA configuration mode command (or the `type echo protocol ipIcmpEcho {destination-ip-address | destination-hostname} [source-ipaddr {ip-address | hostname} | source-interface interface-name]` rtr configuration mode command) to configure an IP SLAs ICMP echo operation. The parameters of these commands are defined in Table 5-3.

Table 5-3. *icmp-echo and type echo protocol ipIcmpEcho Commands*

| Parameter                                                                                                       | Description                                                                                                                                                                             |
|-----------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>destination-ip-address   destination-hostname</i>                                                            | Destination IPv4 or IPv6 address or hostname.                                                                                                                                           |
| <b>source-ip</b> { <i>ip-address   hostname</i> }<br>(or <b>source-ipaddr</b> { <i>ip-address   hostname</i> }) | (Optional) Specifies the source IPv4 or IPv6 address or hostname. When a source IP address or hostname is not specified, the IP SLAs chooses the IP address nearest to the destination. |
| <b>source-interface</b> <i>interface-name</i>                                                                   | (Optional) Specifies the source interface for the operation.                                                                                                                            |

#### Note

Effective with Cisco IOS Release 12.4(4)T, 12.2(33)SB, and 12.2(33)SXI, the **type echo protocol ipIcmpEcho** command is replaced by the **icmp-echo** command.

Use the **frequency seconds** IP SLA configuration submode command (or rtr configuration submode command) to set the rate at which a specified IP SLAs operation repeats. (For example, this command can be entered within the `icmp-echo` command mode.) The *seconds* parameter is the number of seconds between the IP SLAs operations; the default is 60.

Use the **timeout milliseconds** IP SLA configuration submode command (or rtr configuration submode command) to set the amount of time a Cisco IOS IP SLAs operation waits for a response from its request packet. (For example, this command can be entered within the `icmp-echo` command mode.)

The *milliseconds* parameter is the number of milliseconds (ms) the operation waits to receive a response from its request packet. It is recommended that the value of the *milliseconds* parameter be based on the sum of both the maximum RTT value for the packets and the processing time of the IP SLAs operation.

After the Cisco IP SLAs operation is configured, it needs to be scheduled. Use the `ip sla schedule operation-number [life {forever | seconds}] [start-time {hh:mm:ss} [month day | day month] | pending | now | after hh:mm:ss] [ageout seconds] [recurring]` global configuration mode command (or the `ip sla monitor schedule operation-number [life {forever | seconds}] [start-time {hh:mm:ss} [month day | day month] | pending | now | after hh:mm:ss] [ageout seconds] [recurring]` global configuration mode command) to configure the scheduling parameters for a single Cisco IOS IP SLAs operation. The parameters of these commands are defined in Table 5-4.

Table 5-4. *ip sla schedule and ip sla monitor schedule* Commands

| Parameter             | Description                                                                                                                                                                                                                                                                                                             |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| operation-number      | Number of the IP SLAs operation to schedule.                                                                                                                                                                                                                                                                            |
| life forever          | (Optional) Schedules the operation to run indefinitely.                                                                                                                                                                                                                                                                 |
| <b>life</b> seconds   | (Optional) Number of seconds the operation actively collects information. The default is 3600 seconds (1 hour).                                                                                                                                                                                                         |
| start-time            | (Optional) Time when the operation starts.                                                                                                                                                                                                                                                                              |
| hh:mm[:ss]            | Specifies an absolute start time using hour, minute, and (optionally) second. Use the 24-hour clock notation. For example, start time 01:02 means "start at 1:02 a.m.," and start time 13:01:30 means "start at 1:01 p.m. and 30 seconds." The current day is implied unless you specify <i>amonth</i> and <i>day</i> . |
| month                 | (Optional) Name of the month to start the operation in. If month is not specified, the current month is used. Use of this argument requires that a day be specified. You can specify the month by using either the full English name or the first three letters of the month.                                           |
| day                   | (Optional) Number of the day (in the range 1 to 31) to start the operation on. If a day is not specified, the current day is used. Use of this argument requires that a month be specified.                                                                                                                             |
| pending               | (Optional) No information is collected. This is the default value.                                                                                                                                                                                                                                                      |
| now                   | (Optional) Indicates that the operation should start immediately.                                                                                                                                                                                                                                                       |
| <b>after</b> hh:mm:ss | (Optional) Indicates that the operation should start <i>hh</i> hours, <i>mm</i> minutes, and <i>ss</i> seconds after this command was entered.                                                                                                                                                                          |
| <b>ageout</b> seconds | (Optional) Number of seconds to keep the operation in memory when it is not actively collecting information. The default is 0 seconds (never ages out).                                                                                                                                                                 |
| recurring             | (Optional) Indicates that the operation will start automatically at the specified time and for the specified duration every day.                                                                                                                                                                                        |

#### Note

Effective with Cisco IOS Release 12.4(4)T, 12.2(33)SB, and 12.2(33)SXI, the **ip sla monitor schedule** command is replaced by the **ip sla schedule** command.

#### Configuring Cisco IOS IP SLAs Tracking Objects

This section examines some of the configuration commands used to define tracking objects, to track the state of IOS IP SLAs operations.

Use the `track object-number ip sla operation-number {state | reachability} global` configuration command (or the `track object-number rtr operation-number {state | reachability} global` configuration command) to track the state of an IOS IP SLAs operation, and enter track configuration mode. The parameters of these commands are defined in Table 5-5.

Table 5-5. *track ip sla and track rtr* Commands

| Parameter        | Description                                                                      |
|------------------|----------------------------------------------------------------------------------|
| object-number    | Object number representing the object to be tracked. The range is from 1 to 500. |
| operation-number | Number used for the identification of the IP SLAs operation you are              |

|              |                                        |
|--------------|----------------------------------------|
| Parameter    | Description                            |
|              | tracking.                              |
| <b>state</b> | Tracks the operation return code.      |
| reachability | Tracks whether the route is reachable. |

#### Note

Effective with Cisco IOS Release 12.4(20)T, 12.2(33)SXI1, 12.2(33)SRE and Cisco IOS XE Release 2.4, the **track rtr** command is replaced by the **track ip sla** command.

Use the delay {up seconds [down seconds] | [up seconds] down seconds} track configuration command to specify a period of time to delay communicating state changes of a tracked object. The parameters of this command are defined in Table 5-6.

Table 5-6. *delay* Commands

|           |                                                                        |
|-----------|------------------------------------------------------------------------|
| Parameter | Description                                                            |
| up        | Time to delay the notification of an up event.                         |
| down      | Time to delay the notification of a down event.                        |
| seconds   | Delay value, in seconds. The range is from 0 to 180. The default is 0. |

#### Configuring the Action Associated with the Tracking Object

This section describes one of the configuration commands used to define the action associated with the tracking object.

Use the ip route prefix mask {ip-address | interface-type interface-number [ip-address]} [dhcp] [distance] [name next-hop-name] [permanent | tracknumber] [tag tag] global configuration command to establish a static route that tracks an object. The parameters of this command are defined in Table 5-7.

Table 5-7. *ip route* Command

|                                 |                                                                                                                                                                                                           |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameter                       | Description                                                                                                                                                                                               |
| prefix                          | IP route prefix for the destination.                                                                                                                                                                      |
| mask                            | Prefix mask for the destination.                                                                                                                                                                          |
| ip-address                      | IP address of the next hop that can be used to reach that network.                                                                                                                                        |
| interface-type interface-number | Network interface type and interface number.                                                                                                                                                              |
| dhcp                            | (Optional) Enables a Dynamic Host Configuration Protocol (DHCP) server to assign a static route to a default gateway (option 3). Note that you specify the <b>dhcp</b> keyword for each routing protocol. |
| distance                        | (Optional) Administrative distance. The default administrative distance for a static route is 1.                                                                                                          |
| <b>name</b> next-hop-name       | (Optional) Applies a name to the specified route.                                                                                                                                                         |
| permanent                       | (Optional) Specifies that the route will not be removed, even if the interface shuts down.                                                                                                                |
| <b>track</b> number             | (Optional) Associates a track object with this route. Valid values for the <i>number</i> argument range from 1 to 500.                                                                                    |

| Parameter      | Description                                                                                             |
|----------------|---------------------------------------------------------------------------------------------------------|
| <b>tag tag</b> | (Optional) Tag value that can be used as a “match” value for controlling redistribution via route maps. |

The next section introduces some of the commands used to verify path control using IOS IP SLAs. The section after that illustrates two examples of IOS IP SLAs configuration and verification.

#### Verifying Path Control Using IOS IP SLAs

This section describes some of the commands used to verify path control using IOS IP SLAs.

Use the **show ip sla configuration** [operation] command (or the **show ip sla monitor configuration** [operation] command) to display configuration values including all defaults for all Cisco IOS IP SLAs operations, or for a specified operation. The *operation* parameter is the number of the IP SLAs operation for which the details will be displayed.

#### Note

Effective with Cisco IOS Release 12.4(20)T, 12.2(33)SX11, 12.2(33)SRE and Cisco IOS XE Release 2.4, the **show ip sla monitor configuration** command is replaced by the **show ip sla configuration** command.

Use the **show ip sla statistics** [operation-number] [details] command (or the **show ip sla monitor statistics** [operation-number] [details] command) to display the current operational status and statistics of all Cisco IOS IP SLAs operations, or of a specified operation. The parameters of these commands are defined in Table 5-8.

Table 5-8. *show ip sla statistics* and *show ip sla monitor statistics* Commands

| Parameter        | Description                                                                                   |
|------------------|-----------------------------------------------------------------------------------------------|
| operation-number | (Optional) Number of the operation for which operational status and statistics are displayed. |
| details          | (Optional) Operational status and statistics are displayed in greater detail.                 |

#### Note

Effective with Cisco IOS Release 12.4(20)T, 12.2(33)SX11, 12.2(33)SRE and Cisco IOS XE Release 2.4, the **show ip sla monitor statistics** command is replaced by the **show ip sla statistics** command.

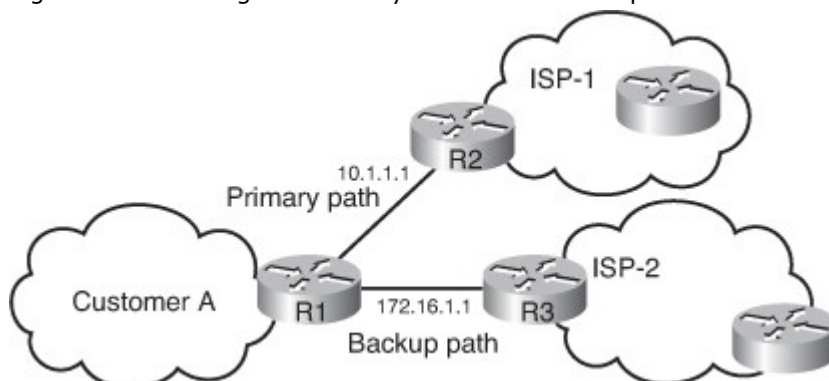
#### Examples of Path Control Using Cisco IOS IP SLAs

This section uses two examples to illustrate IOS IP SLAs configuration and verification.

##### Tracking Reachability to Two ISPs

Figure 5-9 illustrates a scenario in which Customer A is multihoming to two ISPs. Customer A is not using BGP with the ISPs; instead, it is using static default routes. Two default static routes with different administrative distances are configured, so that the link to ISP-1 is the primary link and the link to ISP-2 is the backup link. The static default route with the lower administrative distance will be preferred and injected into the routing table.

Figure 5-9. Tracking Reachability to Two ISPs Example Network.



However, if there is a problem with the ISP-1 router or with its connectivity toward the Internet but its interface to Customer A is still up, all traffic from Customer A will still go to that ISP. This traffic may then get lost within the ISP. The solution to this issue is the Cisco IOS IP SLAs functionality, which can be used to continuously check the reachability of a specific destination (such as a provider edge [PE] router interface, the ISP's DNS server, or any other specific destination) and conditionally announce the default route only if the connectivity is verified.

The Cisco IOS IP SLAs configuration of R1 is provided in Example 5-2.

Example 5-2. Cisco IOS IP SLAs Configuration of Router R1 in Figure 5-9

```
R1(config)#ip sla monitor 11
R1(config-rtr)#type echo protocol ipIcmpEcho 10.1.1.1 source-interface
FastEthernet0/0
R1(config-rtr-echo)#frequency 10
R1(config-rtr-echo)#exit
R1(config)#ip sla monitor schedule 11 life forever start-time now
R1(config)#track 1 rtr 11 reachability
R1(config-track)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 10.1.1.1 2 track 1

R1(config)#ip sla monitor 22
R1(config-rtr)#type echo protocol ipIcmpEcho 172.16.1.1 source-
interface
FastEthernet0/1
R1(config-rtr-echo)#frequency 10
R1(config-rtr-echo)#exit
R1(config)#ip sla monitor schedule 22 life forever start-time now
R1(config)#track 2 rtr 22 reachability
R1(config-track)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 172.16.1.1 3 track 2
```

The first step in this configuration defines the probe; probe 11 is defined by the **ip sla monitor 11** command. The test defined with the **type echo protocol ipIcmpEcho 10.1.1.1 source-interface FastEthernet0/0** command specifies that the ICMP echoes are sent to destination 10.1.1.1 (R2) to check connectivity, with the Fast Ethernet 0/0 interface used as the source interface. The **frequency 10** command schedules the connectivity test to repeat every 10 seconds. The **ip sla monitor schedule 11 life forever start-time now** command defines the start and end time of the connectivity test for probe 11; the start time is now and it will continue forever.

The second step defines the tracking object, which is linked to the probe from the first step. The **track 1 rtr 11 reachability** command specifies that object 1 is tracked; it is linked to probe 11 (defined in the first step) so that the reachability of the 10.1.1.1 is tracked.

The last step defines an action based on the status of the tracking object. The **ip route 0.0.0.0 0.0.0.0 10.1.1.1 2 track 1** command conditionally configures the default route, via 10.1.1.1, with an administrative distance of 2, if the result of tracking object 1 is true. Thus, if 10.1.1.1 is reachable, a static default route via 10.1.1.1 with an administrative distance of 2, is installed in the routing table.

This scenario requires the configuration of two probes, two tracking objects, and two conditionally announced default routes. The second set of configuration commands in Example 5-2 is almost the same as the first set. Probe 22, defined by the **ip sla monitor 22** command, defines the test condition for the reachability of the backup ISP destination address 172.16.1.1, using Fast Ethernet 0/1 as the source address. The test is every 10 seconds, from now to forever. Tracking object 2 is related to the second probe, as defined by the **track 2 rtr 22 reachability** command. The default route configured, via 172.16.1.1, is using a higher administrative distance of 3, because the backup ISP is to be used only if the primary ISP is not available. This default route is offered to the routing table if the result of tracking object 2 is true.

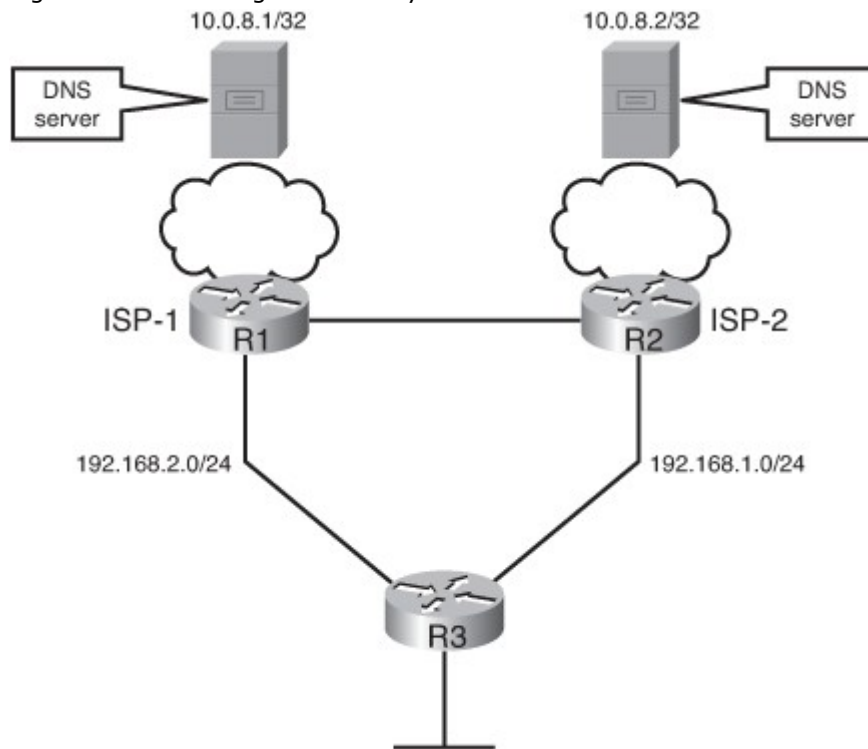
Tracking DNS Server Reachability in the Two ISPs

Figure 5-10 illustrates the network for this example scenario. R3 represents a branch office connected to two ISPs. In this scenario Cisco IOS IP SLAs are used to track the reachability to the DNS servers (with IP



addresses 10.0.8.1 and 10.0.8.2) and tie the results to the static default routes on R3. If there is a DNS server failure, the Cisco IOS IP SLAs probes will fail, the static default route to that DNS will be removed, and all traffic will be rerouted toward the other ISP.

Figure 5-10. Tracking Reachability to DNS Servers in the Two ISPs Example Network.



#### Note

This network was created in a lab to simulate a branch office scenario. The DNS server addresses are simulated by loopback 0 interfaces on R1 and R2. EIGRP is running between R1, R2, and R3.

The following steps detail the implementation and verification of Cisco IOS IP SLAs in this example:

- Step 1. Verify reachability to the DNS servers.
- Step 2. Configure Cisco IOS IP SLAs.
- Step 3. Verify Cisco IOS IP SLAs operations.
- Step 4. Configure tracking options.
- Step 5. Configure static default routes or PBR that are tied to object tracking (the DNS servers).
- Step 6. Verify dynamic operations and routing changes when the tracked objects fail.

Example 5-3 illustrates the results of the reachability verification tests from R3 to the DNS servers.

Example 5-3. Results of Reachability Tests to DNS Servers from R3

```
R3#ping 10.0.8.1
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 10.0.8.1, timeout is 2 seconds:
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/36 ms
```

```
R3#ping 10.0.8.2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 10.0.8.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
R3#
```

After confirming that the reachability tests are successful, the Cisco IOS IP SLAs are configured. The configuration is shown in Example 5-4. The `ip sla monitor 99` command is used to create an ICMP echo probe on R3 to the first DNS server; the operation number 99 is locally significant only to the router. (Note that there are many other types of probes other than the ICMP echo probes that could be created.) The `frequency 10` command schedules the connectivity test to repeat every 10 seconds. The probe is scheduled to start now, and to run forever. A second probe, 100, is similarly created to test connectivity to the second DNS server.

Example 5-4. Configuration of Router R3 in Figure 5-10

```
ip sla monitor 99
type echo protocol ipIcmpEcho 10.0.8.1
frequency 10
ip sla monitor schedule 99 life forever start-time now

ip sla monitor 100
type echo protocol ipIcmpEcho 10.0.8.2
frequency 10
ip sla monitor schedule 100 life forever start-time now
```

The IP SLAs configuration is verified next, using the `show ip sla monitor configuration` command. The partial output is shown in Example 5-5, illustrating the details of the configuration of operation 99. This output confirms that the operation is an echo operation to 10.0.8.1 with a frequency of 10 seconds, and that it has already started (the start time has already passed).

Example 5-5. `show ip sla monitor configuration` Output on R3

```
Code View: Scroll / Show All
R3(config)#do show ip sla monitor configuration
SA Agent, Infrastructure Engine-II
Entry number: 99
Owner:
Tag:
Type of operation to perform: echo
Target address: 10.0.8.1
Request size (ARR data portion): 28
Operation timeout (milliseconds): 5000
Type of Service parameters: 0x0
Verify data: No
Operation frequency (seconds): 10
Next Scheduled Start Time: Start Time already passed
Group Scheduled: FALSE
Life (seconds): Forever
Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE
Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 5000
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1
```

```
Statistic distribution interval (milliseconds): 20
Number of history Lives kept: 0
Number of history Buckets kept: 15
--More--
```

The `show ip sla monitor statistics` command is used next, to display the number of successes, failures, and the results of the latest operations. The output is shown in Example 5-6, and it confirms that operation 99 has succeeded 16 times already, had no failures, and the last operation returned an "OK" result. Operation 100 has succeeded 15 times, had no failures, and its last operation also returned an "OK" result.

Example 5-6. *show ip sla monitor statistics* Output on R3

```
R3(config)#do show ip sla monitor statistics
Round trip time (RTT) Index 99
 Latest RTT: 20 ms
Latest operation start time: *18:07:10.306 UTC Fri May 24 2002
Latest operation return code: OK
Number of successes: 16
Number of failures: 0
Operation time to live: Forever

Round trip time (RTT) Index 100
 Latest RTT: 19 ms
Latest operation start time: *18:07:12.006 UTC Fri May 24 2002
Latest operation return code: OK
Number of successes: 15
Number of failures: 0
Operation time to live: Forever

R3(config)#
```

The next step is to configure tracking objects, as illustrated in Example 5-7. The first tracking object is tied to IP SLAs object 99 and has 10 seconds of down delay and 1 second of up delay, representing the level of sensitivity to changes of tracked objects. The delay helps to alleviate the affect of flapping objects, those that are going down and up rapidly. In this case, if the DNS server fails momentarily and comes back up within 10 seconds, there is no impact. The `ip route` command creates a static default route via 192.168.2.2 (R1) that appears or disappears, depending on the success or failure of the IP SLAs operation. Notice that this command reference the tracking object number 1, which in turn reference IP SLAs operation number 99. The second tracking object is tied to IP SLAs object 100 and has a similar configuration.

Example 5-7. Tracking Object Configuration of Router R3 in Figure 5-10

```
track 1 rtr 99 reachability
 delay down 10 up 1
ip route 0.0.0.0 0.0.0.0 192.168.2.2 track 1

track 2 rtr 100 reachability
 delay down 10 up 1
ip route 0.0.0.0 0.0.0.0 192.168.1.2 track 2
```

Example 5-8 shows the static routes in the IP routing table. This output confirms that both static default routes currently appear in the routing table.

#### Example 5-8. Routing Table on Router R3

```
R3#show ip route static
S* 0.0.0.0 0.0.0.0 [1/0] via 192.168.2.2
 via 192.168.1.2
```

To examine the routing behavior, IP routing debugging is enabled on R3, with the **debug ip routing** command. The DNS address on R2 is shut down. (Recall that in this example, the DNS address is simulated by interface loopback 0 on R2; thus a **shutdown** command on this interface is all that is required.)

Note

The **debug ip routing** command may generate a significant amount of output.

The debug output on R3 is shown in Example 5-9. The EIGRP route to 10.0.8.2 is immediately deleted, and there are now no routes to 10.0.8.2. This is the object being tracked with the track 2 command; it tracks reachability to IP SLAs object 100, which is an ICMP echo to 10.0.8.2. After about 10 seconds, the value specified in the delay command, the static default route via 192.168.1.2 (R2) is deleted.

#### Example 5-9. *debug ip routing* Output on R3

```
Code View: Scroll / Show All
R3#
3w6d: RT: delete route to 10.0.8.2 via 192.168.1.2, eigrp metric [90/156160]
3w6d: RT: SET_LAST_RDB for 10.0.8.2 255.255.255.255
 OLD rdb: via 192.168.1.2, FastEthernet0/1

3w6d: RT: no routes to 10.0.8.2
3w6d: RT: NET-RED 10.0.8.2 255.255.255.255
3w6d: RT: delete subnet route to 10.0.8.2 255.255.255.255
3w6d: RT: NET-RED 10.0.8.2 255.255.255.255
R3#
3w6d: RT: del 0.0.0.0 via 192.168.1.2, static metric [1/0]
3w6d: RT: NET-RED 0.0.0.0 0.0.0.0
R3#
3w6d: RT: NET-RED 0.0.0.0 0.0.0.0
R3#
```

Debugging is disabled, and the statistics are viewed again, using the show ip sla monitor statistics command, as displayed in Example 5-10. This output confirms that there have been 11 failures on the IP SLAs object 100; these are failures in the ICMP echo to 10.0.8.2. The latest return code is "Timeout."

#### Example 5-10. *show ip sla statistics* Output on R3

```
R3#show ip sla monitor statistics
<Output omitted>
Round Trip Time (RTT) for Index 100
 Latest RTT: NoConnection/Busy/Timeout
Latest operation start time: *17:29:26.572 UTC Sun Aug 2 2009
Latest operation return code: Timeout
Number of successes: 80
Number of failures: 11
Operation time to live: Forever
```

```
R3#
```

The static routes in the IP routing table now are shown in Example 5-11. This output confirms that only one static default remains, via 192.168.2.2 (R1).

Example 5-11. *show ip route static* Output on R3

```
R3#show ip route static
S* 0.0.0.0 0.0.0.0 [1/0] via 192.168.2.2
R3#
```

To examine the routing behavior when connectivity to the R2 DNS is restored, IP routing debugging is enabled on R3 again, with the **debug ip routing** command, and the DNS address on R2 is enabled by performing a **no shutdown** command on the loopback 0 interface on R2.

The debug output on R3 is shown in Example 5-12. The EIGRP route to 10.0.8.2 comes up, and almost immediately the default static route via 192.168.1.2 (R2) comes up.

Example 5-12. *debug ip routing* Output on R3

```
Code View: Scroll / Show All
3w6d: RT: SET_LAST_RDB for 10.0.8.2 255.255.255.255
NEW rdb: via 192.168.1.2

3w6d: RT: add 10.0.8.2 255.255.255.255 via 192.168.1.2, eigrp metric [90/156160]
3w6d: RT: NET-RED 10.0.8.2 255.255.255.255
R3#
3w6d: RT: add 0.0.0.0 0.0.0.0 via 192.168.1.2, static metric [1/0]
3w6d: RT: NET-RED 0.0.0.0 0.0.0.0
3w6d: RT: NET-RED 0.0.0.0 0.0.0.0
R3#
3w6d: RT: NET-RED 0.0.0.0 0.0.0.0
R3#
```

The routing table now is shown in Example 5-13; both static default routes are there. Full connectivity has been restored.

Example 5-13. *show ip route static* Output on R3

```
R3#show ip route static
S* 0.0.0.0 0.0.0.0 [1/0] via 192.168.2.2
 via 192.168.1.2
```

An alternative solution for this example network, using PBR, is presented at the end of the next section, after PBR is detailed.

In summary, there are many possibilities available with object tracking and Cisco IOS IP SLAs. As shown in these examples, you can base a probe on reachability, changing routing operations and path control based on the ability to reach an object. You can also use Cisco IOS IP SLAs with Cisco IOS Optimized Edge Routing (OER) to allow paths to be changed based on network conditions such as delay, load, and so forth. (Cisco IOS OER allows the best exit path to be selected, based on a defined policy, and is described briefly in the "Cisco IOS Optimized Edge Routing" section, later in this chapter.)

In deploying the Cisco IOS IP SLAs solution, the impact of the additional probe traffic being generated should also be considered, including how that traffic affects bandwidth utilization and congestion levels. Tuning the

configuration (for example with the **delay** and **frequency** commands) becomes critical to mitigate possible issues related to excessive transitions and route changes in the presence of flapping tracked objects.

### Implementing Path Control Using Policy-Based Routing

Chapter 4 describes route maps and how you can use them for route filtering. This section describes another use for route maps, with PBR. PBR enables the administrator to define a routing policy other than basic destination-based routing using the routing table. With PBR, route maps can be used to match source and destination addresses, protocol types, and end-user applications. When a match occurs, a set command can be used to define items, such as the interface or next-hop address to which the packet should be sent.

#### Using PBR to Control Path Selection

In modern high-performance internetworks, organizations need the freedom to implement packet forwarding and routing according to their own defined policies in a way that goes beyond traditional routing protocol concerns.

Routers normally forward packets to destination addresses based on information in their routing tables. By using PBR, introduced in Cisco IOS Release 11.0, you can implement policies that selectively cause packets to take different paths based on source address, protocol types, or application types. Therefore, PBR overrides the router's normal routing procedures.

PBR also provides a mechanism to mark packets with different types of service (ToS). This feature can be used in conjunction with Cisco IOS queuing techniques so that certain kinds of traffic can receive preferential service.

PBR provides an extremely powerful, simple, and flexible tool to implement solutions in cases where legal, contractual, or political constraints dictate that traffic be routed through specific paths. Benefits you can achieve by implementing PBR include the following:

- **Source-based transit provider selection**— ISPs and other organizations can use PBR to route traffic originating from different sets of users through different Internet connections across policy routers.
- **QoS**— Organizations can provide QoS to differentiated traffic by setting the ToS values in the IP packet headers in routers at the periphery of the network and then leveraging queuing mechanisms to prioritize traffic in the network's core or backbone. This setup improves network performance by eliminating the need to classify the traffic explicitly at each WAN interface in the network's core or backbone.
- **Cost savings**— Using PBR, an organization can direct the bulk traffic associated with a specific activity to use a higher-bandwidth, high-cost link for a short time and to continue basic connectivity over a lower-bandwidth, low-cost link for interactive traffic.
- **Load sharing**— In addition to the dynamic load-sharing capabilities offered by destination-based routing that the Cisco IOS Software has always supported, network managers can implement policies to distribute traffic among multiple paths based on the traffic characteristics.

#### Configuring PBR

Configuring PBR involves configuring a route map with **match** and **set** commands and then applying the route map to the interface.

When configuring PBR, it is important to note that PBR is applied to *incoming* packets. Enabling PBR causes the router to evaluate all packets incoming on the interface using a route map configured for that purpose.

The steps required to implement path control include the following:

1. Choose the path control tool to use. Path control tools manipulate or bypass the IP routing table. For PBR, **route-map** commands are used.
2. Implement the traffic-matching configuration, specifying which traffic will be manipulated; **match** commands are used within route maps.
3. Define the action for the matched traffic, using **set** commands within route maps.
4. Optionally, fast-switched PBR or Cisco Express Forwarding (CEF)-switched PBR can be enabled. Fast-switched PBR must be enabled manually. CEF-switched PBR is automatically enabled when CEF switching is enabled (which it is by default in recent IOS versions) and PBR is enabled.
5. Apply the route map to incoming traffic or to traffic locally generated on the router.
6. Verify path control results, using **show** commands.

You can configure the route map statements used for PBR as **permit** or **deny**. The following defines how these options work:

- If the statement is marked as **deny**, a packet meeting the match criteria is not policybased routed. Instead, it is sent through the normal forwarding channels; in other words, destination-based routing is performed.
- Only if the statement is marked as **permit** and the packet meets all the match criteria are the **set** commands applied.
- If no match is found in the route map, the packet is *not* dropped; it is forwarded through the normal routing channel, which means that destination-based routing is performed.
- If you do not want to revert to normal forwarding but instead want to drop a packet that does not match the specified criteria, configure a **set** statement to route the packets to interface null 0 as the last entry in the route map.

#### PBR match Commands

IP standard or extended access lists can be used to establish PBR match criteria using the `match ip address {access-list-number | name} [...access-list-number | name] | prefix-list prefix-list-name [...prefix-list-name]` route map configuration command, as explained in Table 5-9. You can use a standard IP access list to specify match criteria for a packet's source address. You can use extended access lists to specify match criteria based on source and destination addresses, application, protocol type, and ToS.

Table 5-9. *match ip address* Command

| Parameter                                  | Description                                                                                                                                                                |
|--------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>access-list-number   name</code>     | The number or name of a standard or extended access list to be used to test incoming packets. If multiple access lists are specified, matching any one results in a match. |
| <b>prefix-list</b> <i>prefix-list-name</i> | Specifies the name of a prefix list to be used to test packets. If multiple prefix lists are specified, matching any one results in a match.                               |

Use the `match length min max` route map configuration command, explained in Table 5-10, to establish criteria based on the packet length between specified minimum and maximum values. For example, a network administrator could use the match length as the criterion that distinguishes between interactive and file transfer traffic, because file transfer traffic usually has larger packet sizes.

Table 5-10. *match length* Command

| Parameter        | Description                                                         |
|------------------|---------------------------------------------------------------------|
| <code>min</code> | The packet's minimum Layer 3 length, inclusive, allowed for a match |
| <code>max</code> | The packet's maximum Layer 3 length, inclusive, allowed for a match |

#### PBR set Commands

If the **match** statements are satisfied, you can use one or more of the **set** statements described in this section to specify the criteria for forwarding packets through the router.

The router evaluates the first four **set** commands for PBR shown in this section in the order they are presented. As soon as a destination address or interface has been chosen, other **set** commands for changing the destination address or interface are ignored. Note, however, that some of these commands affect only packets for which there is an *explicit* route in the routing table, and others affect only packets for which there is *no explicit* route in the routing table.

By default, a packet that is not affected by any of the **set** commands in a route map statement it has matched is not policy routed and is forwarded normally; in other words, destination-based routing is performed.

##### set ip next-hop Command

The `set ip next-hop ip-address [...ip-address]` route map configuration command provides a list of IP addresses used to specify the adjacent next-hop router in the path toward the destination to which the packets should be forwarded. If more than one IP address is specified, the first IP address associated with a



currently up and connected interface is used to route the packets. Table 5-11 explains the `set ip next-hop` command.

Table 5-11. *set ip next-hop* Command

| Parameter  | Description                                                                                               |
|------------|-----------------------------------------------------------------------------------------------------------|
| ip-address | The IP address of the next hop to which packets are output. It must be the address of an adjacent router. |

The **set ip next-hop** command affects all packet types and is always used if configured.

Note

With the **set ip next-hop** command, the routing table is checked only to determine whether the next hop can be reached. It is not checked to determine whether there is an explicit route for the packet's destination address.

set interface Command

The `set interface type number [... type number]` route map configuration command provides a list of interfaces through which the packets can be routed. If more than one interface is specified, the first interface that is found to be up is used to forward the packets. Table 5-12 explains this command.

Table 5-12. *set interface* Command

| Parameter   | Description                                               |
|-------------|-----------------------------------------------------------|
| type number | The interface type and number to which packets are output |

If there is *no* explicit route for the destination address of the packet in the routing table (for example, if the packet is a broadcast or is destined for an unknown address), the **set interface** command has no effect and is ignored. A default route in the routing table is *not* considered an explicit route for an unknown destination address.

set ip default next-hop Command

The `set ip default next-hop ip-address [...ip-address]` route map configuration command provides a list of default next-hop IP addresses. If more than one IP address is specified, the first next hop specified that appears to be adjacent to the router is used. The optional specified IP addresses are tried in turn. Table 5-13 explains this command.

Table 5-13. *set ip default next-hop* Command

| Parameter  | Description                                                                                               |
|------------|-----------------------------------------------------------------------------------------------------------|
| ip-address | The IP address of the next hop to which packets are output. It must be the address of an adjacent router. |

A packet is routed to the next hop specified by the **set ip default next-hop** command only if there is *no* explicit route for the packet's destination address in the routing table. A default route in the routing table is *not* considered an explicit route for an unknown destination address.

set default interface Command

The `set default interface type number [...type number]` route map configuration command provides a list of default interfaces. If no explicit route is available to the destination address of the packet being considered for policy routing, it is routed to the first up interface in the list of specified default interfaces. Table 5-14 provides information about this command.

Table 5-14. *set default interface* Command

| Parameter   | Description                                                |
|-------------|------------------------------------------------------------|
| type number | The interface type and number to which packets are output. |

A packet is routed to the next hop specified by the **set default interface** command only if there is *no* explicit route for the packet's destination address in the routing table. A default route in the routing table is *not* considered an explicit route for an unknown destination address.

PBR also provides a mechanism to mark packets using the **set ip tos** and **set ip precedence** commands, as shown in the next two sections.

#### set ip tos Command

The **set ip tos** [*number* | *name*] route map configuration command is used to set some of the bits in the IP ToS field in the IP packet. The ToS field in the IP header is 8 bits long, with 5 bits for setting the class of service (CoS) and 3 bits for the IP precedence. The CoS bits are used to set the delay, throughput, reliability, and cost.

The set ip tos command is used to set the 5 CoS bits. Values 0 through 15 are used (one of the bits is reserved). Table 5-15 provides the names and numbers of the defined ToS values used in this command.

Table 5-15. *set ip tos* Command

| Parameter <i>number</i>   <i>name</i> | Description                    |
|---------------------------------------|--------------------------------|
| <b>0   normal</b>                     | Sets the normal ToS            |
| <b>1   min-monetary-cost</b>          | Sets the min-monetary-cost ToS |
| <b>2   max-reliability</b>            | Sets the max reliable ToS      |
| <b>4   max-throughput</b>             | Sets the max throughput ToS    |
| <b>8   min-delay</b>                  | Sets the min delay ToS         |

#### set ip precedence Command

The set ip precedence [*number* | *name*] route map configuration command enables you to set the 3 IP precedence bits in the IP packet header. With 3 bits, you have eight possible values for the IP precedence; values 0 through 7 are defined. This command is used when implementing QoS and can be used by other QoS services, such as weighted fair queuing (WFQ) and weighted random early detection (WRED). Table 5-16 provides the names and numbers of the defined IP precedence values used in this command.

Table 5-16. *set ip precedence* Command

| Parameter <i>number</i>   <i>name</i> | Description                              |
|---------------------------------------|------------------------------------------|
| <b>0   routine</b>                    | Sets the routine precedence              |
| <b>1   priority</b>                   | Sets the priority precedence             |
| <b>2   immediate</b>                  | Sets the immediate precedence            |
| <b>3   flash</b>                      | Sets the Flash precedence                |
| <b>4   flash-override</b>             | Sets the Flash override precedence       |
| <b>5   critical</b>                   | Sets the critical precedence             |
| <b>6   internet</b>                   | Sets the internetwork control precedence |
| <b>7   network</b>                    | Sets the network control precedence      |

You can use the **set** commands in conjunction with each other.

#### Configuring PBR on an Interface

To identify a route map to use for policy routing on an interface, use the ip policy route-map map-tag interface configuration command. Table 5-17 explains the parameter.

Table 5-17. *ip policy route-map* Command

| Parameter | Description                                                                                                           |
|-----------|-----------------------------------------------------------------------------------------------------------------------|
| map-tag   | The name of the route map to use for policy routing. It must match a map tag specified by a <b>route-map</b> command. |

Remember that policy-based routing is configured on the interface that *receives* the packets, not on the interface from which the packets are sent.

Packets originating on the router are not normally policy routed. Local policy routing enables packets originating on the router to take a route other than the obvious shortest path. To identify a route map to use for local policy routing, use the `ip local policy route-map map-tag global` configuration command. Table 5-18 explains the parameter. This command applies the specified route map to packets originating on the router.

Table 5-18. *ip local policy route-map* Command

| Parameter | Description                                                                                                                 |
|-----------|-----------------------------------------------------------------------------------------------------------------------------|
| map-tag   | The name of the route map to use for local policy routing. It must match a map tag specified by a <b>route-map</b> command. |

Since Cisco IOS Release 12.0, IP PBR can now be fast switched. Before this feature, policy routing could only be process switched, which meant that on most platforms, the switching rate was approximately 1000 to 10,000 packets per second. This was not fast enough for many applications. Users who need policy routing to occur at faster speeds can now implement policy routing without slowing down the router.

PBR must be configured before you configure fast-switched policy routing. Fast switching of policy routing is disabled by default. To enable it, use the **ip route-cache policy** interface configuration command.

Fast-switched PBR supports all the **match** commands and most of the **set** commands, except for the following restrictions:

- The **set ip default next-hop** and **set default interface** commands are not supported.
- The **set interface** command is supported only over point-to-point links unless a route-cache entry exists using the same interface specified in the **set interface** command in the route map. Also, when process switching, the routing table is checked to determine whether the interface is on an appropriate path to the destination. The software does not make this check during fast switching. Instead, if the packet matches, the software blindly forwards the packet to the specified interface.

#### Note

The **ip route-cache policy** command is strictly for fast-switched PBR, and therefore, not required for a CEF-switched PBR.

#### Verifying PBR

To display the route maps used for policy routing on the router's interfaces, use the **show ip policy** EXEC command.

To display configured route maps, use the **show route-map [map-name]** EXEC command, where *map-name* is an optional name of a specific route map.

Use the **debug ip policy** EXEC command to display IP policy routing packet activity. This command shows in detail what policy routing is doing. It displays information about whether a packet matches the criteria and, if so, the resulting routing information for the packet.

#### Note

Because the **debug ip policy** command generates a significant amount of output, use it only when traffic on the IP network is low, so that other activity on the system is not adversely affected.

To discover the routes that the packets follow when traveling to their destination from the router, use the **traceroute** EXEC command. To change the default parameters and invoke an extended **traceroute**, enter the command without a destination argument. You are then stepped through a dialog to select the desired parameters.

To check host reachability and network connectivity, use the **ping EXEC** command. You can use the **ping** command's extended command mode to specify the supported header options by entering the command without any arguments.

#### PBR Examples

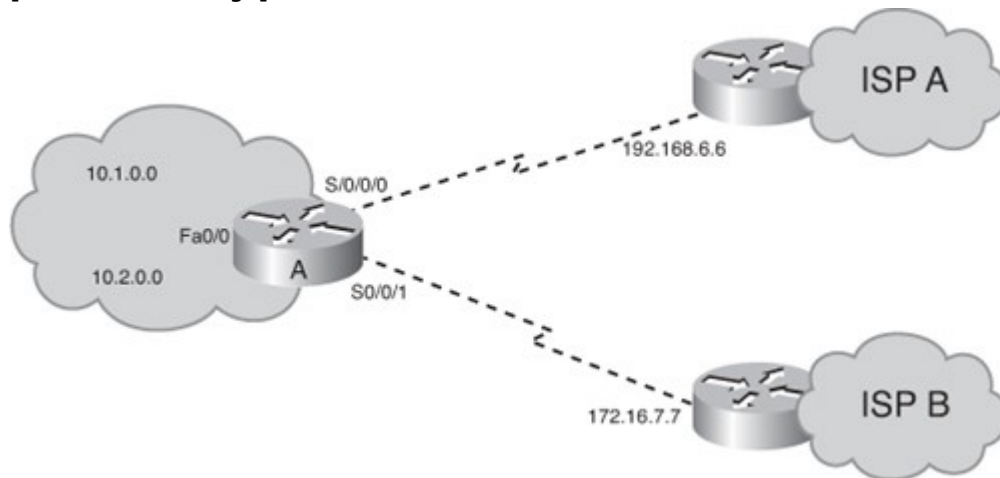
This section provides three examples of PBR.

##### Using PBR When Connecting Two ISPs

In Figure 5-11, Router A provides Internet access for a private enterprise and is connected to two different ISPs. This router is advertising a 0.0.0.0 default route into the enterprise network to avoid large routing tables.

Figure 5-11. Router A Is Connected to Two ISPs.

[View full size image]



Therefore, when traffic from the enterprise networks 10.1.0.0 and 10.2.0.0 reaches Router A, it can go to either ISP A or ISP B. The company prefers to have ISP A and ISP B receive approximately equal amounts of traffic. PBR is implemented on Router A to shape, or load balance, traffic from Router A to each of the ISPs. All traffic sourced from the 10.1.0.0 subnet is forwarded to ISP A if there is no specific route to the destination in the routing table (the default route is not used). All traffic sourced from the 10.2.0.0 subnet is forwarded to ISP B if there is no specific route to the destination in the routing table.

#### Caution

Remember, this policy provides for an outbound traffic policy from the enterprise to its ISPs only. It does not determine the inbound traffic policy for Router A. It is possible that traffic from 10.1.0.0 going out to ISP A will receive responses through ISP B.

Example 5-14 shows the configuration for Router A. Route map equal-access is configured.

Example 5-14. Configuration of Router A in Figure 5-11

```
RouterA(config)#access-list 1 permit 10.1.0.0 0.0.255.255
RouterA(config)#access-list 2 permit 10.2.0.0 0.0.255.255

RouterA(config)#route-map equal-access permit 10
RouterA(config-route-map)#match ip address 1
RouterA(config-route-map)#set ip default next-hop 192.168.6.6

RouterA(config-route-map)#route-map equal-access permit 20
RouterA(config-route-map)#match ip address 2
RouterA(config-route-map)#set ip default next-hop 172.16.7.7

RouterA(config-route-map)#route-map equal-access permit 30
RouterA(config-route-map)#set default interface null0
```

```
RouterA(config-route-map)#exit
RouterA(config)#interface FastEthernet 0/0
RouterA(config-if)#ip address 10.1.1.1 255.255.255.0
RouterA(config-if)#ip policy route-map equal-access
RouterA(config-if)#exit
RouterA(config)#interface Serial 0/0/0
RouterA(config-if)#ip address 192.168.6.5 255.255.255.0
RouterA(config-if)#exit
RouterA(config)#interface Serial 0/0/1
RouterA(config-if)#ip address 172.16.7.6 255.255.255.0
```

The **ip policy route-map equal-access** command is applied to the Fast Ethernet 0/0 interface, the *incoming* interface receiving the packets to be policy-routed.

Sequence number 10 in route map equal-access is used to match all packets sourced from any host in subnet 10.1.0.0. If there is a match, and if the router has no explicit route for the packet's destination, it is sent to next-hop address 192.168.6.6 (ISP A's router).

Sequence number 20 in route map equal-access is used to match all packets sourced from any host in subnet 10.2.0.0. If there is a match, and if the router has no explicit route for the packet's destination, it is sent to next-hop address 172.16.7.7 (ISP B's router).

Sequence number 30 in route map equal-access is used to drop all traffic not sourced from subnet 10.1.0.0 or 10.2.0.0. The null 0 interface is a route to nowhere; traffic is dropped.

The outputs shown in Examples 5-15, 5-16, and 5-17 are from Router A in Figure 5-11. Example 5-15 provides an example of show ip policy command output, indicating that the route map called equal-access is used for PBR on the router's Fast Ethernet 0/0 interface.

Example 5-15. show ip policy on Router A in Figure 5-11

```
RouterA#show ip policy
Interface Route map
FastEthernet0/0 equal-access
```

Example 5-16 provides an example of show route-map command output, indicating that three packets have matched sequence 10 of the equal-access route map.

Example 5-16. show route-map on Router A in Figure 5-11

```
RouterA#show route-map
route-map equal-access, permit, sequence 10
Match clauses:
 ip address (access-lists): 1
Set clauses:
 ip default next-hop 192.168.6.6
Policy routing matches: 3 packets, 168 bytes
route-map equal-access, permit, sequence 20
Match clauses:
 ip address (access-lists): 2
Set clauses:
 ip default next-hop 172.16.7.7
route-map equal-access, permit, sequence 30
Set clauses:
 default interface null0
```

Example 5-17 provides an example of the debug ip policy command output. The output indicates that a packet from 10.1.1.1 destined for 172.19.1.1 has been received on interface Fast Ethernet 0/0 and that it is

policy-routed on Serial 0/0/0 to next hop 192.168.6.6 (because the source address of 10.1.1.1 matches line 10 of route map equal-access).

Example 5-17. debug ip policy on Router A in Figure 5-11

Code View: Scroll / Show All

RouterA#**debug ip policy**

Policy routing debugging is on

11:51:25: IP: s=10.1.1.1 (FastEthernet0/0), d=172.19.1.1, len 100, **policy match**

11:51:25: IP: route map equal-access, item 10, permit

11:51:25: IP: s=10.1.1.1 (FastEthernet0/0), d=172.19.1.1 (Serial0/0/0), len 100, **policy routed**

11:51:25: IP: FastEthernet0/0/0 to Serial0/0/0 192.168.6.6

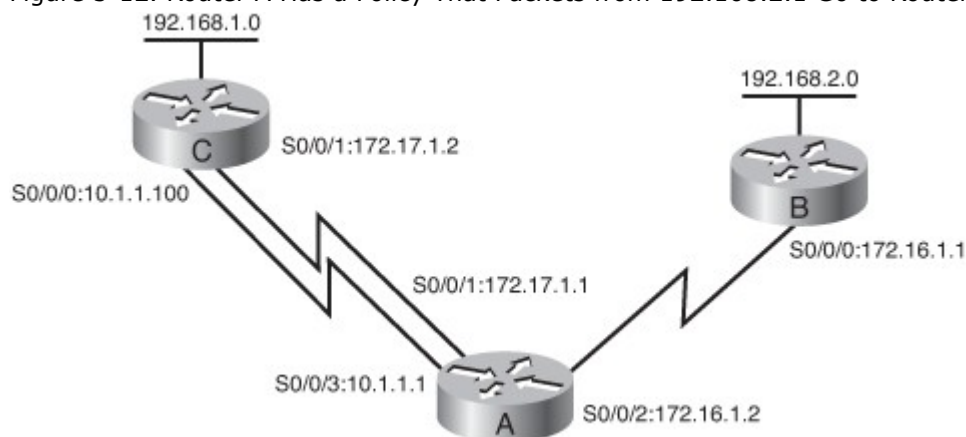
Note

The **show logging** command shows the logging buffer, including the output of the **debug** command.

Using PBR Based on Source Address

In Figure 5-12, Router A has a policy that packets with a source address of 192.168.2.1 (on the other side of Router B) should go out to Router C's interface Serial 0/0/1, 172.17.1.2 (via Router A's S0/0/1 interface). All other packets should be routed according to their destination address. Example 5-18 shows the relevant part of the configuration for Router A.

Figure 5-12. Router A Has a Policy That Packets from 192.168.2.1 Go to Router C's Interface S0/0/1.



Example 5-18. Configuration of Router A in Figure 5-12

```
RouterA(config)#interface Serial0/0/2
RouterA(config-if)#ip address 172.16.1.2 255.255.255.0
RouterA(config-if)#ip policy route-map test
RouterA(config-if)#route-map test permit 10
RouterA(config-route-map)#match ip address 1
RouterA(config-route-map)#set ip next-hop 172.17.1.2
RouterA(config-route-map)#exit
RouterA(config)#access-list 1 permit 192.168.2.1 0.0.0.0
```

Router A's Serial 0/0/2 interface, where packets from 192.168.2.1 go *into* Router A, is configured to do policy routing with the **ip policy route-map** command. The route map test is used for this policy routing. It tests the IP addresses in packets against access list 1 to determine which packets will be policy-routed.

Access list 1 specifies that packets with a source address of 192.168.2.1 are policy routed. Packets that match access list 1 are sent to the next-hop address 172.17.1.2, which is Router C's Serial 0/0/1 interface. All other packets are forwarded normally, according to their destination address. (Recall that access lists have an implicit **deny any** at the end, so no other packets are permitted by access list 1.)

The outputs shown in Examples 5-19, 5-20, and 5-21 are from Router A in Figure 5-12. Example 5-19 provides an example of the show ip policy command output. It indicates that the route map called test is used for policy routing on the router's interface Serial 0/0/2.

Example 5-19. show ip policy Output on Router A in Figure 5-12

```
RouterA#show ip policy
Interface Route map
Serial0/0/2 test
```

The show route-map command, shown in Example 5-20, indicates that three packets have matched sequence 10 of the test route map.

Example 5-20. show route-map Output on Router A in Figure 5-12

```
RouterA#show route-map
route-map test, permit, sequence 10
 Match clauses:
 ip address (access-lists): 1
 Set clauses:
 ip next-hop 172.17.1.2
Policy routing matches: 3 packets, 168 bytes
```

Example 5-21 provides an example of the output of the debug ip policy command. The output indicates that a packet from 172.16.1.1 destined for 192.168.1.1 was received on interface Serial 0/0/2 and that it was rejected by the policy on that interface. The packet is routed normally (by destination). Another packet, from 192.168.2.1 destined for 192.168.1.1, was later received on the same interface, Serial 0/0/2. This packet matched the policy on that interface and therefore was policy routed and sent out interface Serial 0/0/1 to 172.17.1.2.

Example 5-21. Example of debug ip policy on Router A in Figure 5-12

```
Code View: Scroll / Show All
RouterA#debug ip policy
Policy routing debugging is on

...
11:50:51: IP: s=172.16.1.1 (Serial0/0/2), d=192.168.1.1 (Serial0/0/3), len 100,
policy rejected — normal forwarding
...
11:51:25: IP: s=192.168.2.1 (Serial0/0/2), d=192.168.1.1, len 100, policy match
11:51:25: IP: route map test, item 10, permit
11:51:25: IP: s=192.168.2.1 (Serial0/0/2), d=192.168.1.1 (Serial0/0/1), len 100,
policy routed
11:51:25: IP: Serial0/0/2 to Serial0/0/1 172.17.1.2
```

Alternative Solution IP SLAs Configuration Example Using PBR

This section presents an alternative solution to the configuration of the R3 router in Figure 5-10 given earlier in this chapter in the "Examples of Path Control Using Cisco IOS IP SLAs" section. A partial configuration is shown in Example 5-22, providing just the configuration for reachability to the R1 router. Explanatory comments are provided within the configuration. (Configuration for reachability to the R2 router would be



similar.) Using PBR allows the configuration to be very granular, to support other options. In this example, PBR points to a next-hop address that is tracked via Cisco IOS IP SLAs.

Example 5-22. Partial Alternative Configuration for Router R3 in Figure 5-10

```
!Configure the object to be tracked; object 1 will be up if the router
!can ping 10.0.8.1
ip sla 99
 icmp-echo 10.0.8.1
 frequency 10
 timeout 5000
ip sla schedule 99 start-time now life forever
!
track 1 rtr 99 reachability
!
!Enable policy routing using route map IP-SLA
interface FastEthernet 0/0
 ip address 10.2.8.1 255.255.255.0
 ip policy route-map IP-SLA
!
!Configure a route-map to set the next-hop to 192.168.2.1 (R1) if
! object 1 is up. If object 1 is down, then policy routing fails
! and unicast routing will route the packet.
route-map IP-SLA
 set ip next-hop verify-availability 192.168.2.1 10 track 1
```

This configuration uses the `set ip next-hop verify-availability [next-hop-address sequence track object]` route-map configuration command to configure policy routing to verify the reachability of the next hop of a route map before the router performs policy routing to that next hop. Table 5-19 explains the parameters of this command.

Table 5-19. *set ip next-hop verify-availability* Command

| Parameter        | Description                                                                                              |
|------------------|----------------------------------------------------------------------------------------------------------|
| next-hop-address | (Optional) IP address of the next hop to which packets will be forwarded.                                |
| sequence         | (Optional) Sequence of next hops. The acceptable range is from 1 to 65535.                               |
| track            | (Optional) The tracking method is track.                                                                 |
| object           | (Optional) Object number that the tracking subsystem is tracking. The acceptable range is from 1 to 500. |

Because of the use of route maps, this type of configuration allows you more granularity to define, via access lists or prefix lists, which traffic classes will be subject to changes based on the results of the object tracking. For example routes for voice, mission-critical data, and other traffic types could be changed.

#### Advanced Path Control Tools

This section provides a brief overview of additional path control mechanisms that you might encounter in your enterprise networks.

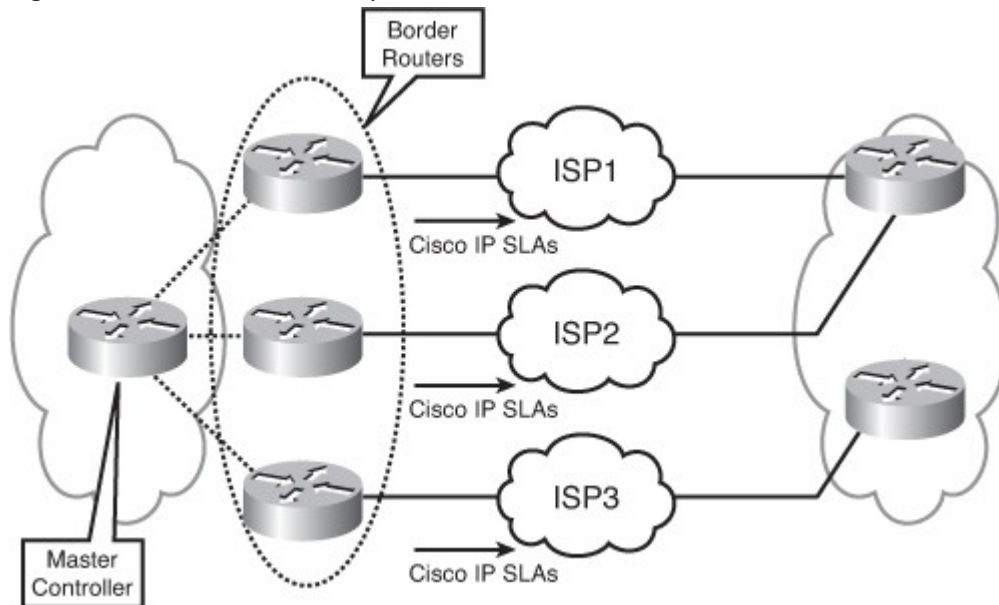
##### Cisco IOS Optimized Edge Routing

Cisco IOS OER is intended for sites using multiple Internet or WAN service providers. Cisco IOS OER uses tools such as Cisco IOS IP SLAs to automatically detect network service degradation and to make dynamic routing decisions and adjustments based on criteria such as response time, packet loss, jitter, path availability, traffic load distribution, and so forth.

In contrast, normal routing, using routing protocols, focuses on detecting a routing path using static routing metrics, rather than the condition of the service over that path.

An example is illustrated in Figure 5-13. The Cisco IOS OER edge routers, called border routers, monitor information about route prefixes (using traditional routing protocols) and gather performance statistics over each external interface (in this example, using Cisco IOS IP SLAs).

Figure 5-13. Cisco IOS OER Operations.



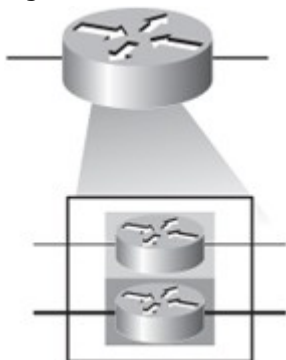
This information is periodically reported to another router called the master controller. If the prefixes and exit links comply with a configured policy based on performance and service metrics, routing remains as is. If not, the master controller makes a policy-based decision and notifies the border routers, which change the path, by such mechanisms as adding static routes or changing routing protocol parameters.

#### Virtualization

Virtualization is another advanced technology being used in enterprise networks that includes benefits such as traffic segregation across a common physical network infrastructure.

An example of virtualization is the use of virtual routing and forwarding (VRF) tables, which are virtual routing tables used to separate the routing function by group, on one physical router, as illustrated in Figure 5-14.

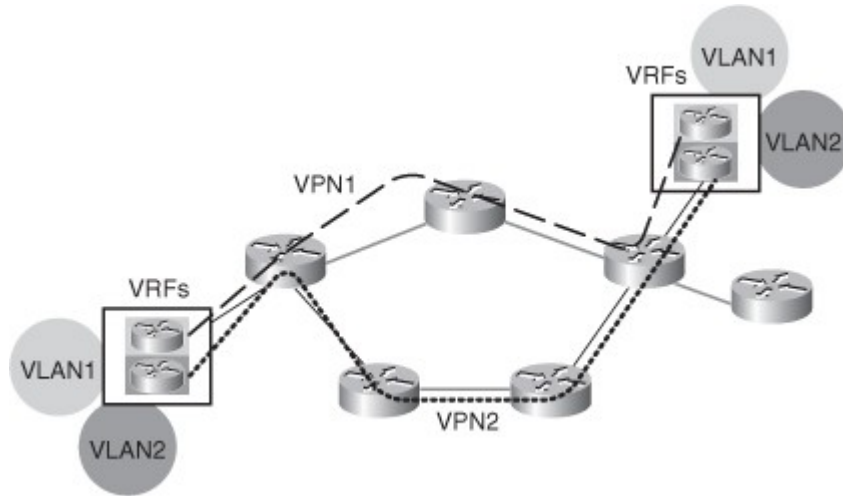
Figure 5-14. VRF Creates Separate Virtual Routing Tables in One Physical Router.



For example, employee routes could be kept separate from guest routes by using two different VRFs. These VRFs could also be associated with other virtualization and traffic segregation elements on the network, such as virtual LANs (VLANs), virtual private networks (VPNs), and generic routing encapsulation (GRE) tunnels, to provide an end-to-end, segregated path across the network. An example is illustrated in Figure 5-15, in which path control is based on a design decision to engineer different paths, end to end, with a variety of network virtualization technologies. In this figure, two business units are associated with two different VRFs

on the end routers. These VRFs are associated with different VLANs and VPNs throughout the network, to provide an end-to-end segregated path across the network.

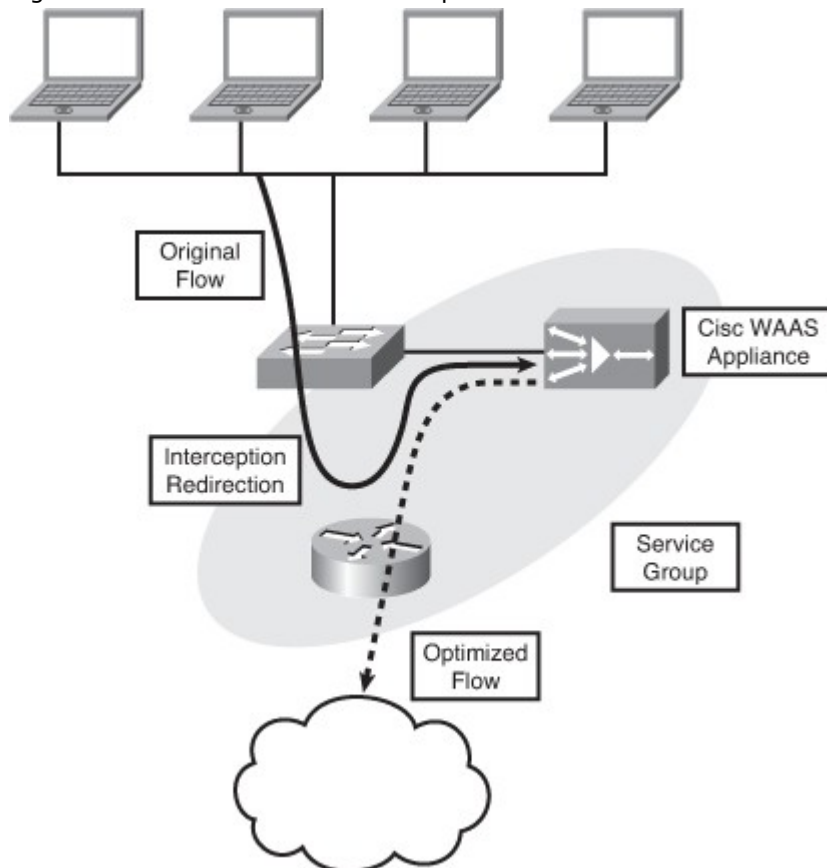
Figure 5-15. Virtualization Technologies Used for Path Control.



#### Cisco Wide Area Application Services

Cisco WAAS is a good example of the use of PBR to adjust the path of traffic based on advanced services for that traffic, to provide both scalability and high availability. Technologies such as Web Cache Communications Protocol (WCCP) perform a similar function, which is to have routers redirect normal traffic flows into Cisco WAAS devices, where a series of data reduction, flow optimization, and application acceleration services are implemented, and then have them route the flows back into their normal path across the WAN. This scenario is illustrated in the example in Figure 5-16. This use of path control is becoming common in networks with branch offices.

Figure 5-16. WCCP Used for WAN Optimization.



## Summary

In this chapter, you learned about implementing path control. The chapter focused on the following topics:

- Redundant network considerations including resiliency, availability, adaptability, performance, support for network and application services, predictability, and asymmetric traffic.
- Path control tools including a good addressing design, redistribution and other routing protocol characteristics, passive interfaces, distribute lists, prefix lists, administrative distance, route maps, route tagging, offset lists, Cisco IOS IP SLAs, and PBR. (Advanced tools covered briefly include Cisco IOS OER, virtualization, and Cisco WAAS.)
- Offset lists, a mechanism for increasing incoming and outgoing metrics to routes learned via EIGRP or RIP. Configuration of offset lists is performed with the **offset-list** {*access-list-number* | *access-list-name*} {**in** | **out**} *offset* [*interface-type interface-number*] router configuration command. Verification of offset lists can be performed with the **traceroute** command, the **show ip route** command, and the **show ip eigrp topology** command.
- Cisco IOS IP SLAs, which use active traffic monitoring, generating traffic in a continuous, reliable, and predictable manner, to measure network performance. IOS IP SLAs can be used in conjunction with other tools, including the following:
  - Object tracking, to track the reachability of specified objects
  - Cisco IOS IP SLAs probes, to send different types of probes toward the desired objects
  - Route maps with PBR, to associate the results of the tracking to the routing process
  - Static routes with tracking options, as a simpler alternative to PBR
- Cisco IOS IP SLAs terminology, including the following:
  - All the Cisco IOS IP SLAs measurement probe operations are configured on the IP SLAs *source*, either by the CLI or through an SNMP tool that supports IP SLAs operation. The source sends probe packets to the *target*.
  - There are two types of IP SLAs operations: those in which the target device is running the IP SLAs *responder* component, and those in which the target device is not running the IP SLAs responder component (such as a web server or IP host).
  - An *IP SLAs operation* is a measurement that includes protocol, frequency, traps, and thresholds.
- Configuring IOS IP SLAs, including the use of the following commands:
  - The **ip sla operation-number** global configuration command (or the **ip sla monitor operation-number** global configuration command) to begin configuring a Cisco IOS IP SLAs operation and enter IP SLA configuration mode (or rtr configuration mode).
  - The **icmp-echo** {*destination-ip-address* | *destination-hostname*} [**source-ip** {*ip-address* | *hostname*} | **source-interface** *interface-name*] IP SLA configuration mode command (or the **type echo protocol ipIcmpEcho** {*destination-ip-address* | *destination-hostname*} [**source-ipaddr**{*ip-address* | *hostname*} | **source-interface** *interface-name*] rtr configuration mode command) to configure an IP SLAs ICMP echo operation.
  - The **frequency seconds** IP SLA configuration submode command (or rtr configuration submode command) to set the rate at which a specified IP SLAs operation repeats.
  - The **timeout milliseconds** IP SLA configuration submode command (or rtr configuration submode command) to set the amount of time a Cisco IOS IP SLAs operation waits for a response from its request packet.
  - The **ip sla schedule operation-number** [**life** {**forever** | *seconds*}] [**start-time** {*hh:mm[:ss]* [*month day* | *day month*] | **pending** | **now** | **after hh:mm:ss**}] [**ageout seconds**] [**recurring**] global configuration mode command (or the **ip sla monitor schedule operation-number**[**life** {**forever** | *seconds*}] [**start-time** {*hh:mm[:ss]* [*month day* | *day month*] | **pending** | **now** | **after hh:mm:ss**}] [**ageout seconds**] [**recurring**] global configuration mode command) to configure the scheduling parameters for a single Cisco IOS IP SLAs operation.
  - The **track object-number ip sla operation-number** {**state** | **reachability**} global configuration command (or the **track object-number rtr operation-number** {**state** | **reachability**} global configuration command) to track the state of an IOS IP SLAs operation, and enter track configuration mode.

- The **delay** {**up seconds** [**down seconds**] | [**up seconds**] **down seconds**} track configuration command to specify a period of time to delay communicating state changes of a tracked object.
- The **ip route prefix mask** {*ip-address* | *interface-type interface-number* [*ip-address*]} [**dhcp**] [*distance*] [**name next-hop-name**] [**permanent** | **track number**] [**tag tag**] global configuration command to establish a static route that tracks an object.
- Verifying Cisco IOS IP SLAs, including the use of the **show ip sla configuration** [*operation*] command (or the **show ip sla monitor configuration** [*operation*] command), and the **show ip sla statistics** [*operation-number*] [**details**] command (or the **show ip sla monitor statistics** [*operation-number*] [**details**] command).
- Using PBR to control path selection, providing benefits including source-based transit provider selection, QoS, cost savings, and load sharing. PBR is applied to *incoming* packets; enabling PBR causes the router to evaluate all packets incoming on the interface using a route map configured for that purpose.
- Configuring and verifying PBR, including the following steps:
  - Choose the path control tool to use; for PBR, **route-map** commands are used.
  - Implement the traffic-matching configuration, specifying which traffic will be manipulated; **match** commands are used within route maps.
  - Define the action for the matched traffic, using **set** commands within route maps.
  - Optionally, fast-switched PBR or CEF-switched PBR can be enabled. Fast-switched PBR must be enabled manually. CEF-switched PBR is automatically enabled when CEF switching is enabled and PBR is enabled.
  - Apply the route map to incoming traffic or to traffic locally generated on the router.
  - Verify path control results, using **show** commands.
- PBR **match** commands, including the following:
  - The **match ip address** {*access-list-number* | *name*} [...*access-list-number* | *name*] route map configuration command
  - The **match length** *min max* route map configuration command
- PBR **set** commands, including the following four which are evaluated in this order (as soon as a destination address or interface has been chosen, other **set** commands for changing the destination address or interface are ignored):
  - The **set ip next-hop** *ip-address* [...*ip-address*] route map configuration command, which affects all packet types and is always used if configured.
  - The **set interface** *type number* [...*type number*] route map configuration command. If there is *no* explicit route for the destination address of the packet in the routing table (for example, if the packet is a broadcast or is destined for an unknown address), the **set interface** command has no effect and is ignored. A default route in the routing table is *not* considered an explicit route for an unknown destination address.
  - The **set ip default next-hop** *ip-address* [...*ip-address*] route map configuration command. A packet is routed to the next hop specified by the **set ip default next-hop** command only if there is *no* explicit route for the packet's destination address in the routing table. A default route in the routing table is *not* considered an explicit route for an unknown destination address.
  - The **set default interface** *type number* [...*type number*] route map configuration command. A packet is routed to the next hop specified by the **set default interface** command only if there is *no* explicit route for the packet's destination address in the routing table. A default route in the routing table is *not* considered an explicit route for an unknown destination address.
- Other PBR **set** commands, including the following:
  - The **set ip tos** [*number* | *name*] route map configuration command, used to set the 5 CoS bits. Values 0 through 15 are used; one of the bits is reserved.
  - The **set ip precedence** [*number* | *name*] route map configuration command, used to set the 3 IP precedence bits in the IP packet header.
  - The **set ip next-hop verify-availability** [*next-hop-address sequence track object*] route-map configuration command to configure policy routing to verify the reachability of the next hop of a route map before the router performs policy routing to that next hop.

- Commands to configure PBR on an interface, including the following:
  - The **ip policy route-map** *map-tag* interface configuration command, configured on the interface that *receives* the packets, not on the interface from which the packets are sent
  - The **ip local policy route-map** *map-tag* global configuration command, to apply a route map to packets originating on the router
- Commands to verify PBR, including the **show ip policy** command, the **show route-map** [*map-name*] command, the **debug ip policy** command, the **traceroute** command, and **ping** command.
- Advanced path control tools, including the following:
  - Cisco IOS OER, which uses tools such as Cisco IOS IP SLAs to automatically detect network service degradation and to make dynamic routing decisions and adjustments based on criteria such as response time, packet loss, jitter, path availability, traffic load distribution, and so forth
  - Virtualization, such as the use of VRF tables, VLANs, VPNs, and GRE tunnels
  - Cisco WAAS, including the use of WCCP to redirect normal traffic flows into Cisco WAAS devices

## References

For additional information, see these resources:

- “Cisco IOS Software Releases 12.4 Mainline” support  
page: [http://www.cisco.com/en/US/products/ps6350/tsd\\_products\\_support\\_series\\_home.html](http://www.cisco.com/en/US/products/ps6350/tsd_products_support_series_home.html)
- The Cisco IOS Command  
Reference: [http://www.cisco.com/en/US/products/ps6350/prod\\_command\\_reference\\_list.html](http://www.cisco.com/en/US/products/ps6350/prod_command_reference_list.html)
- The Cisco IOS IP SLAs Command  
Reference: [http://www.cisco.com/en/US/docs/ios/ipsla/command/reference/sla\\_book.html](http://www.cisco.com/en/US/docs/ios/ipsla/command/reference/sla_book.html)
- Cisco Optimized Edge Routing (OER) home  
page: [http://www.cisco.com/en/US/tech/tk1335/tsd\\_technology\\_support\\_sub-protocol\\_home.html](http://www.cisco.com/en/US/tech/tk1335/tsd_technology_support_sub-protocol_home.html)

## Review Questions

Answer the following questions, and then see Appendix A, “Answers to Review Questions,” for the answers.

1. List some considerations for redundant networks.
2. How does address summarization help keep a network stable?
3. List some path control tools.
4. Indicate whether each statement is referring to OSPF or EIGRP.

| Statement                                                                | Routing Protocol |
|--------------------------------------------------------------------------|------------------|
| Metric can be changed only for external routes at redistribution points. |                  |
| Next hop can be set for all routes under various conditions.             |                  |
| Can be configured only on ABRs and ASBRs.                                |                  |
| Unequal-cost load balancing is available.                                |                  |
| All routes can be tagged.                                                |                  |

5. Select the true statements.
  - a. An empty prefix list denies all prefixes.
  - b. Offset lists increase the incoming metric of routes.
  - c. A distribute list allows an access list to be applied to routing updates.
  - d. If a prefix is permitted, the route is used. If a prefix is denied, the route is not used.
  - e. Offset lists decrease the incoming metric of routes.
6. In the **offset-list** command, what is the *access-list-number* or *access-list-name* parameter used for?
7. Fill in the blank: \_\_\_\_\_ use active traffic monitoring, generating traffic in a continuous, reliable, and predictable manner, to measure network performance



8. What is a Cisco IOS IP SLAs responder?
9. Which ports are used when an IP SLAs source sends to an IP SLAs responder?
10. Select the true statements about IP SLAs.
  - a. Operations are configured on the IP SLAs source.
  - b. Operations are configured on the IP SLAs responder.
  - c. A Cisco IOS device can be an IP SLAs responder.
  - d. A Cisco IOS device can be an IP SLAs source.
  - e. A web server can be an IP SLAs source.
11. Write the command to track the reachability of IOS IP SLAs operation number 100 with object number 2.
12. Write the command to start IP SLAs operation number 100 immediately and have it never end.
13. What are some benefits of policy-based routing (PBR)?
14. To which packets on an interface is PBR applied?
15. When a route map is used for PBR, which of the following are true statements?
  - a. If the statement is marked as **deny**, a packet meeting the match criteria is sent through the normal forwarding channels.
  - b. If the statement is marked as **deny**, a packet meeting the match criteria is dropped.
  - c. If the statement is marked as **permit** and the packet meets all the match criteria, the **set** commands are applied.
  - d. If the statement is marked as **permit** and the packet meets all the match criteria, the packet is sent through the normal forwarding channels.
  - e. If no match is found in the route map, the packet is not dropped.
  - f. If no match is found in the route map, the packet is dropped.
16. In which order are the following **set** commands evaluated?  
set default interface  
set interface  
set ip default next-hop  
set ip next-hop
17. When is the **set default interface** command used?
18. When are the **ip policy route-map** and **ip local policy route-map** commands used?
19. What does the **set ip next-hop verify-availability** command do?
20. How does OER differ from normal routing?
21. What are virtual routing forwarding (VRF) tables?

#### Chapter 6. Implementing a Border Gateway Protocol Solution for ISP Connectivity

This chapter introduces the Border Gateway Protocol (BGP), including the fundamentals of BGP operation. This chapter covers the following topics:

- BGP Terminology, Concepts, and Operation
- Configuring BGP
- Verifying and Troubleshooting BGP
- Basic BGP Path Manipulation Using Route Maps
- Filtering BGP Routing Updates

The Internet is becoming a vital resource for most organizations, resulting in redundant connections to multiple Internet service providers (ISPs). With multiple connections, Border Gateway Protocol (BGP) is an alternative to using default routes to control path selections.

Configuring and troubleshooting BGP can be complex. A BGP administrator must understand the various options involved in properly configuring BGP for scalable internetworking. This chapter introduces BGP terminology and concepts, and provides BGP configuration, verification, and troubleshooting techniques. The chapter also introduces route maps for manipulating BGP path attributes and filters for BGP routing updates.



## BGP Terminology, Concepts, and Operation

This section provides an introduction to BGP and an explanation of various BGP terminology and concepts. Autonomous Systems

To understand BGP, you first need to understand how it is different from the other protocols discussed so far in this book, including an understanding of autonomous systems.

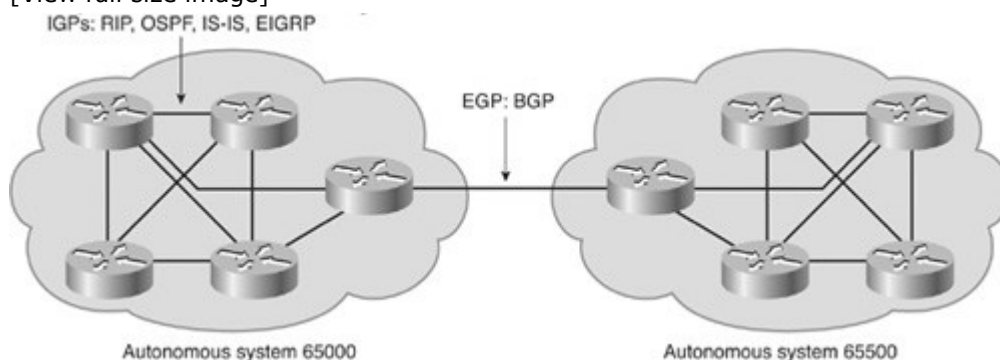
One way to categorize routing protocols is by whether they are interior or exterior:

- **Interior gateway protocol (IGP)**— A routing protocol that exchanges routing information *within* an autonomous system (AS). Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Intermediate System-to-Intermediate System (IS-IS), and Enhanced Interior Gateway Routing Protocol (EIGRP) are examples of IGPs for IP.
- **Exterior gateway protocol (EGP)**— A routing protocol that exchanges routing information *between* different autonomous systems. BGP is an example of an EGP.

BGP is an interdomain routing protocol (IDRP), which is also known as an EGP. All the routing protocols you have seen so far in this book are IGPs.

Figure 6-1 illustrates the concept of IGPs and EGPs.

Figure 6-1. IGPs Operate Within an Autonomous System, and EGPs Operate Between Autonomous Systems. [View full size image]



### Note

The term *IDRP* as used in this sense is a generic term, not the IDRP defined in ISO/IEC International Standard 10747, *Protocol for the Exchange of Inter-Domain Routing Information Among Intermediate Systems to Support Forwarding of ISO 8473 PDUs*.

BGP Version 4 (BGP-4) is the latest version of BGP. It is defined in RFC 4271, *A Border Gateway Protocol (BGP-4)*. As noted in this RFC, the classic definition of an autonomous system is “a set of routers under a single technical administration, using an Interior Gateway Protocol (IGP) and common metrics to determine how to route packets within the autonomous system, and using an inter-autonomous system routing protocol to determine how to route packets to other autonomous systems.”

### Note

Extensions to BGP-4, known as BGP4+, have been defined to support multiple protocols, including IP Version 6 (IPv6). These multiprotocol extensions to BGP are defined in RFC 4760, *Multiprotocol Extensions for BGP-4*.

Autonomous systems might use more than one IGP, with potentially several sets of metrics. The important characteristic of an autonomous system from the BGP point of view is that the autonomous system appears to other autonomous systems to have a single coherent interior routing plan, and it presents a consistent picture of which destinations can be reached through it. All parts of the autonomous system must be connected to each other.

The Internet Assigned Numbers Authority (IANA) is the umbrella organization responsible for allocating autonomous system numbers. Regional Internet registries (RIRs) are nonprofit corporations established for the purpose of administration and registration of IP address space and autonomous system numbers. There are five RIRs, as follows:

- African Network Information Centre (AfriNIC) is responsible for the continent of Africa.

- Asia Pacific Network Information Centre (APNIC) administers the numbers for the Asia Pacific region.
- American Registry for Internet Numbers (ARIN) has jurisdiction over assigning numbers for Canada, the United States, and several islands in the Caribbean Sea and North Atlantic Ocean.
- Latin American and Caribbean IP Address Regional Registry (LACNIC) is responsible for allocation in Latin America and portions of the Caribbean.
- Réseaux IP Européens Network Coordination Centre (RIPE NCC) administers the numbers for Europe, the Middle East, and Central Asia.

The autonomous system designator is a 16-bit number, with a range of 1 to 65,535. RFC 1930, *Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)*, provides guidelines for the use of autonomous system numbers. A range of autonomous system numbers, 64512 to 65,535, is reserved for private use, much like the private IP addresses.

Note

All of the examples and exercises in this book use private autonomous system numbers, to avoid publishing autonomous system numbers belonging to an organization.

Note

RFC 4893, BGP Support for Four-Octet AS Number Space, preparing for the anticipated depletion of BGP 16-bit autonomous system numbers, describes extensions to BGP to use a 32-bit autonomous system number. The Cisco document "Explaining 4-Octet Autonomous System (AS) Numbers for Cisco IOS," available at [http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6554/ps6599/white\\_paper\\_C11\\_516823.html](http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6554/ps6599/white_paper_C11_516823.html), explains how the new numbering scheme can be implemented in Cisco routers.

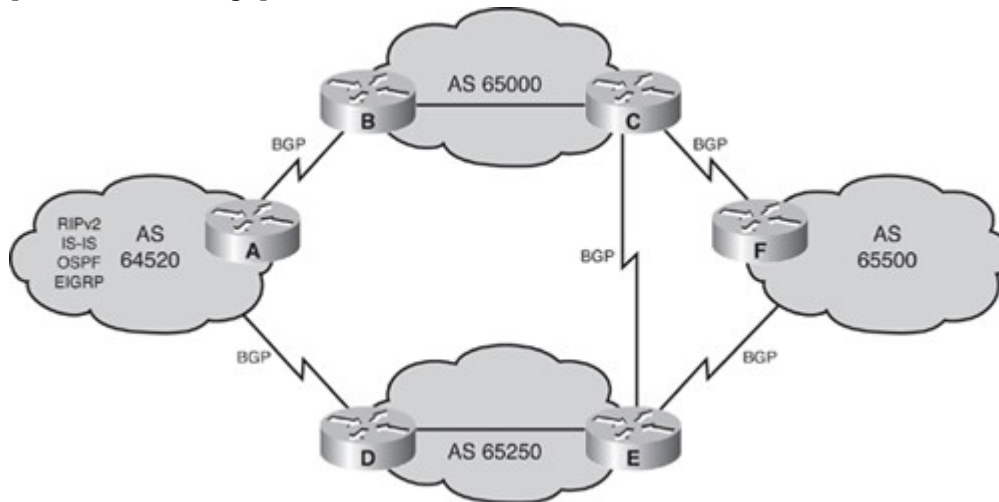
You need to use the IANA-assigned autonomous system number, rather than a private autonomous system number, only if your organization plans to use an EGP, such as BGP, to connect to a public network such as the Internet.

#### BGP Use Between Autonomous Systems

BGP is used between autonomous systems, as illustrated in Figure 6-2.

Figure 6-2. BGP-4 Is Used Between Autonomous Systems on the Internet.

[View full size image]



The main goal of BGP is to provide an interdomain routing system that guarantees the loop-free exchange of routing information between autonomous systems. BGP routers exchange information about paths to destination networks.

Note

BGP is a successor to Exterior Gateway Protocol (EGP). (Note the dual use of the EGP acronym.) The EGP protocol was developed to isolate networks from each other at the early stages of the Internet.

There is a distinction between an ordinary autonomous system and one that has been configured with BGP to implement a transit policy. The latter is called an ISP or a service provider.

Many RFCs relate to BGP-4, including those listed in Table 6-1.

Table 6-1. RFCs Relating to BGP-4

| RFC Number | RFC Title                                                                                                                                                  |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RFC 1772   | An Application of BGP on the Internet                                                                                                                      |
| RFC 1773   | Experience with the BGP-4 Protocol                                                                                                                         |
| RFC 1774   | BGP-4 Protocol Analysis                                                                                                                                    |
| RFC 1930   | Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)                                                                          |
| RFC 1997   | BGP Communities Attribute                                                                                                                                  |
| RFC 1998   | Application of the BGP Community Attribute in Multihome Routing                                                                                            |
| RFC 2042   | Registering New BGP Attribute Types                                                                                                                        |
| RFC 2385   | Protection of BGP Sessions via TCP MD5 Signature Option                                                                                                    |
| RFC 2439   | BGP Route Flap Damping                                                                                                                                     |
| RFC 2545   | Use of BGP-4 Multiprotocol Extensions for IPv6 Interdomain Routing                                                                                         |
| RFC 2918   | Route Refresh Capability for BGP-4                                                                                                                         |
| RFC 3107   | Carrying Label Information in BGP-4                                                                                                                        |
| RFC 4223   | Reclassification of RFC 1863 to Historic                                                                                                                   |
| RFC 4271   | A Border Gateway Protocol 4 (BGP-4)                                                                                                                        |
| RFC 4364   | <i>BGP/MPLS IP Virtual Private Networks (VPNs)</i><br>(updated by RFC 4577, RFC 4684, RFC 5462)                                                            |
| RFC 4456   | BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)                                                                                      |
| RFC 4577   | OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)                                                                |
| RFC 4684   | Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs) |
| RFC 4760   | Multiprotocol Extensions for BGP-4                                                                                                                         |
| RFC 4893   | BGP Support for Four-Octet AS Number Space                                                                                                                 |
| RFC 5065   | Autonomous System Confederations for BGP                                                                                                                   |
| RFC 5462   | Multiprotocol Label Switching (MPLS) Label Stack Entry: EXP Field Renamed to Traffic Class Field                                                           |
| RFC 5492   | Capabilities Advertisement with BGP-4                                                                                                                      |

Note

You can search for RFCs by number at <http://www.rfc-editor.org/rfcsearch.html>.

BGP-4 has many enhancements over earlier protocols. It is used extensively on the Internet today to connect ISPs and to interconnect enterprises to ISPs.

BGP-4 and its extensions are the only acceptable version of BGP available for use on the public Internet. BGP-4 carries a network mask for each advertised network and supports both variable-length subnet mask (VLSM) and classless interdomain routing (CIDR). BGP-4 predecessors did not support these capabilities, which are currently mandatory on the Internet. When CIDR is used on a core router for a major ISP, the IP routing table, which is composed mostly of BGP routes, has more than 300,000 CIDR blocks. Not using CIDR at the Internet level would cause the IP routing table to have more than 2,000,000 entries. Using CIDR, and, therefore, BGP-4, prevents the Internet routing table from becoming too large for interconnecting millions of users.

#### Comparison with Other Scalable Routing Protocols

Table 6-2 compares some of BGP's key characteristics to the other scalable IP routing protocols (including EIGRP and OSPF, which are discussed in this book).

Table 6-2. Comparison of Scalable Routing Protocols

| Protocol | Interior or Exterior | Type                     | Hierarchy Required? | Metric                    |
|----------|----------------------|--------------------------|---------------------|---------------------------|
| OSPF     | Interior             | Link state               | Yes                 | Cost                      |
| IS-IS    | Interior             | Link state               | Yes                 | Metric                    |
| EIGRP    | Interior             | Advanced distance vector | No                  | Composite                 |
| BGP      | Exterior             | Path vector              | No                  | Path vectors (attributes) |

As shown in Table 6-2, OSPF, IS-IS, and EIGRP are interior protocols, whereas BGP is an exterior protocol.

Chapter 1, "Routing Services," discusses the characteristics of distance vector and link-state routing protocols. OSPF and IS-IS are link-state protocols, whereas EIGRP is an advanced distance vector protocol. BGP is also a distance vector protocol, with many enhancements; it is also called a path vector protocol.

Most link-state routing protocols, including OSPF and IS-IS, require a hierarchical design as the network expands, especially to support proper address summarization. For OSPF and IS-IS this hierarchical design is implemented by separating a large internetwork into smaller internetworks called areas. EIGRP and BGP do not require a hierarchical topology.

BGP works differently than IGP. Internal routing protocols look at the path cost to get somewhere and choose the best path from one point in a corporate network to another based on certain metrics. RIP uses hop count and looks to cross the fewest Layer 3 devices to reach the destination network. OSPF uses cost, which on Cisco routers is based on bandwidth, as its metric. The IS-IS metric is typically based on bandwidth (but it defaults to 10 on all interfaces on Cisco routers). EIGRP uses a composite metric, with bandwidth and accumulated delay considered by default.

In contrast, BGP does not look at speed for the best path. Rather, BGP is a policy-based routing protocol that allows an autonomous system to control traffic flow using multiple BGP attributes. Routers running BGP exchange network reachability information, called path vectors or attributes, including a list of the full path of BGP autonomous system numbers that a router should take to reach a destination network. BGP allows an organization to fully use all of its bandwidth by manipulating these path attributes.

#### Connecting Enterprise Networks to an ISP

Modern corporate IP networks connect to the global Internet, use the Internet for some of their data transport needs, and provide services via the Internet to customers and business partners. To meet these needs, systems from web servers to mainframes to workstations are required to be accessible from anywhere in the world.

Requirements that must be determined for connecting an enterprise to an ISP include the following:

- **Public IP address space**— In the rare case that only one-way connectivity, from the clients to the Internet, is required, private IP addresses with Network Address Translation (NAT) are used, allowing clients on a private network to communicate with servers on the public Internet. Typically, though,

two-way connectivity is needed, such that clients external to the enterprise network can access resources in the enterprise network. In this case, both public and private address space is needed, as is routing.

- **Enterprise-to-ISP connection link type and bandwidth—** The type and bandwidth available depends on the ISP and may include leased line, Ethernet over fiber or copper, and various types of digital subscriber line (DSL) (also known as xDSL). The bandwidth provisioned should address the enterprise Internet connectivity requirements.
- **Routing protocol—** Either static or dynamic routing.
- **Connection redundancy—** The type of redundancy required for the enterprise network to ISP connectivity must be evaluated. Options include edge router redundancy, link redundancy, and ISP redundancy.

These requirements are discussed more fully in the following sections.

#### Public IP Address Space

Public IP addresses are used to translate client private addresses for those clients that need to access resources on the Internet. They are also used for enterprise servers that need to be accessible from the Internet; these servers are either configured with public addresses or with private addresses that are statically translated to public addresses. If the enterprise network needs to be independent of the selected ISPs, public IP address space should not be used from the ISP public address space but must instead be acquired from a regional Internet authority. Similarly, a public autonomous system number would be required, rather than using an autonomous system number assigned by the ISP.

#### Connection Link Type and Routing

Connecting an enterprise network to one or more ISPs requires routing information to be exchanged between them. How that routing information is exchanged depends on the requirements, such as the answer to the following questions:

- Does the routing need to respond to the changes in the network topology, such as when a link goes down?
- Will the enterprise network be connected to multiple ISPs?
- Does the routing need to support one link to an ISP or multiple links, to one or multiple ISPs?
- Is traffic load balancing over multiple links required?
- Does the ISP only offer a transport capability for connecting customer's locations, via Layer 2 technologies?
- Which routing options does the ISP offer?
- How much routing information needs to be exchanged with the ISP?

The following sections describe two examples of connecting an enterprise network to one or more ISPs, using Layer 2 circuit emulation and Layer 3 Multiprotocol Label Switching (MPLS) virtual private networks (VPNs). Following those sections, the use of static routes and BGP routing is explored.

#### Using Layer 2 Circuit Emulation

Layer 2 connectivity may be needed between two or more locations, such as in the following examples:

- The locations include data centers with geographically distributed clusters that require Layer2 connectivity to function properly.
- The enterprise is in the process of migrating to a Layer 3 solution, but still requires Layer 2 connectivity.
- The enterprise connects to another partner with which it requires Layer2 connectivity.

As described in Chapter 2, "Configuring the Enhanced Interior Gateway Routing Protocol," service providers offer many Layer 2 services, including Ethernet, Frame Relay, Point-to-Point Protocol (PPP), High-Level Data Link Control (HDLC), and ATM.

When MPLS VPNs were introduced, they provided a unified network for Layer 3 VPN services. For customers still wanting Layer 2 connections, Ethernet virtual LAN (VLAN) extensions across a metropolitan area or ATM services could be deployed. Any Transport over MPLS (AToM) was introduced to facilitate this Layer 2 connectivity across an MPLS backbone.

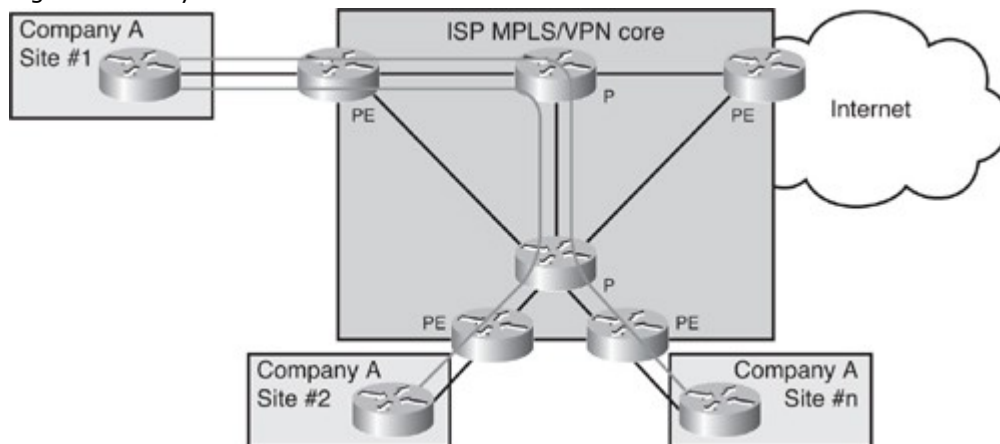
AToM enables sending Layer 2 frames across an MPLS backbone. It unifies Layer 2 and Layer 3 offerings over a common MPLS infrastructure. In AToM, virtual circuits (VCs) represent Layer 2 links, and MPLS labels identify VCs.

AToM allows ISPs that currently offer Layer 2 connectivity to customers with traditional offerings such as ATM, Frame Relay, and serial PPP services, and those specializing in Ethernet connectivity in metropolitan

areas, to expand their offerings. These Layer 2 VPN services appeal to the ISP's enterprise customers who may already run their own networks and desire only point-to-point connectivity between sites.

Figure 6-3 illustrates multiple sites of Company A connected via a Layer 2 MPLS VPN. This Layer 2 MPLS VPN provides a Layer 2 service across the backbone, where all of Company A's edge routers are connected together on the same IP subnet. From Company A's perspective, the ISP is providing a Layer 2 port (an Ethernet switch port). From the ISP's perspective, it is connecting two ports together. There is no routing exchange between the ISP and Company A.

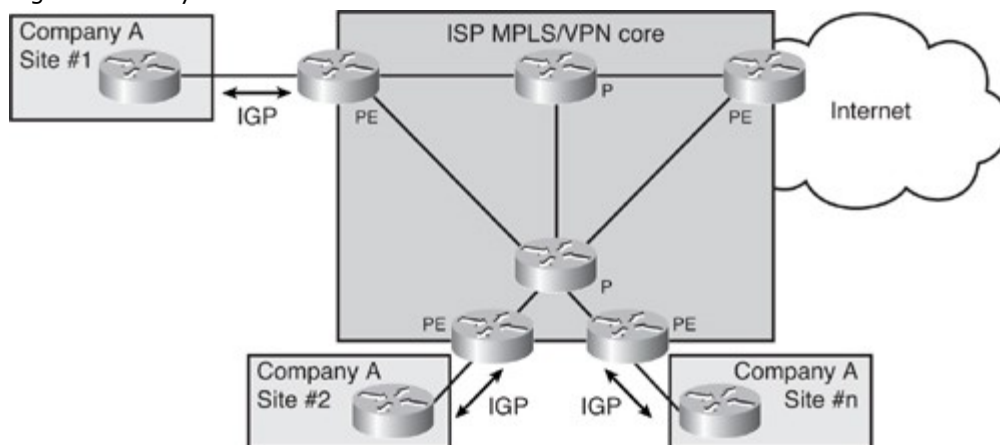
Figure 6-3. Layer 2 MPLS VPN Connection.



#### Using Layer 3 MPLS VPNs

Figure 6-4 illustrates multiple sites of Company A connected via a Layer 3 MPLS VPN. The Layer 3 MPLS VPN provides a Layer 3 service across the backbone, where all of Company A's edge routers are connected to ISP edge routers. A separate IP subnet is used for the connection of each edge router.

Figure 6-4. Layer 3 MPLS VPN Connection.



MPLS VPNs are used when a customer has multiple locations that need to be interconnected through an ISP and does not want to use expensive Layer 2 technologies such as leased lines.

With MPLS VPNs, the ISP uses a common IP-based core network enhanced with MPLS technology to provide secure and manageable connectivity for different customers to their geographically disperse sites. Traffic from different customers of the ISP shares the same physical infrastructure, but is tagged with MPLS labels so that the traffic cannot intermix.

When a customer uses MPLS VPN functionality, routing between the customer and ISP is required, to provide connectivity between the customer locations. The routing used depend on what the ISP supports, but options may include static routes or dynamic routing, including RIP, EIGRP, OSPF, IS-IS, or BGP. Although not typical, different locations could use different routing protocols.

The customer routers are configured for the IGP as if there is a corporate network between them. The ISP and the customer must agree on the IGP parameters; however, these are often governed by the ISP.



With an MPLS VPN deployment, the service provider can also offer Internet connectivity through the same MPLS core network, either through a special Internet VPN or through a global routing table. To exchange the Internet routing information, either BGP or default routes are used.

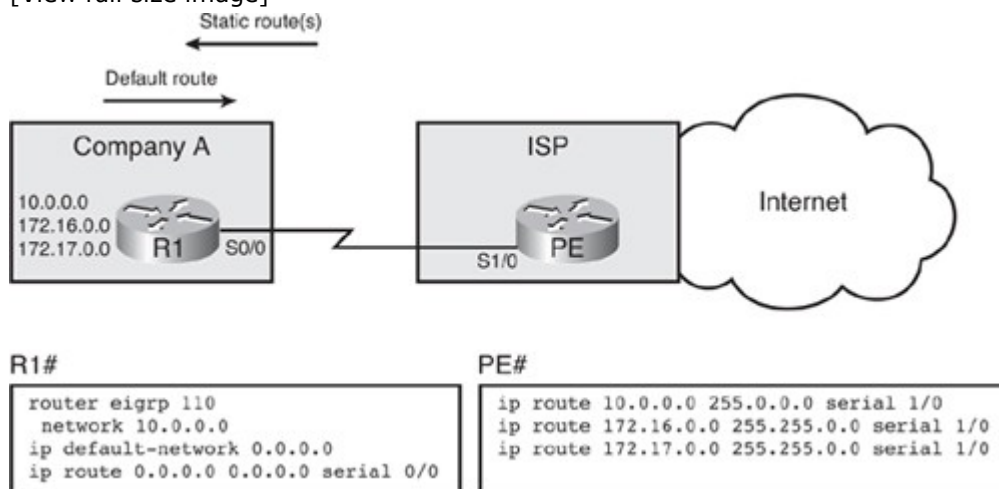
#### Using Static Routes

Configuring static routes between a customer's edge router and an ISP is the simplest way to implement packet forwarding with an ISP. The specific routes configured need to be agreed upon with the ISP and they should not need to be changed very often, because this change would have to be done manually.

Static routes are typically used for Internet connectivity when a customer is connected through a single connection to an ISP, and where that customer can use a default route toward the ISP, as shown in Figure 6-5. The ISP deploys static routes that encompass all the customer's public networks, and would typically also redistribute this information into its BGP routing protocol. Partial configurations of the routers are also shown in Figure 6-5.

Figure 6-5. Using Static Routes for ISP Connectivity.

[View full size image]

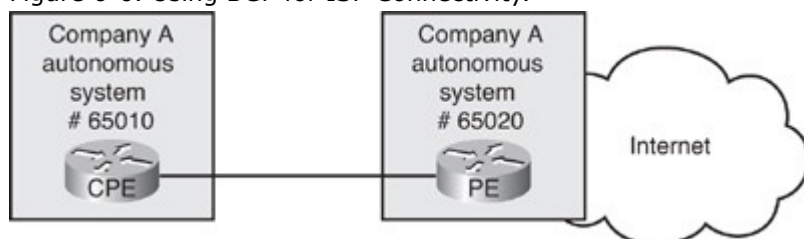


Static routes have drawbacks, especially in terms of flexibility and adaptability. For example, if there were a change in a network topology (beyond a directly connected link failure), the static routes would not adapt. The static routes could be combined with Cisco IOS IP Service Level Agreements (SLAs) functionality (described in Chapter 5, "Implementing Path Control"), which could declare a static route down if a certain condition is met. However, IP SLAs cannot react to all changes in the topology in the Internet. Alternatively, dynamic BGP routing can be used.

#### Using BGP

BGP dynamically exchanges routing information, and thus reacts to topology changes including those changes beyond a customer-to-ISP link failure. Figure 6-6 shows a simple example with a customer provider edge router (CPE) connecting to the ISP provider edge (PE) router.

Figure 6-6. Using BGP for ISP Connectivity.



This option is the focus of the rest of this chapter.

#### Connection Redundancy

When connecting an enterprise network to an ISP, redundancy can be achieved by deploying redundant links, deploying redundant devices, and using redundant components within a device, such as a router.



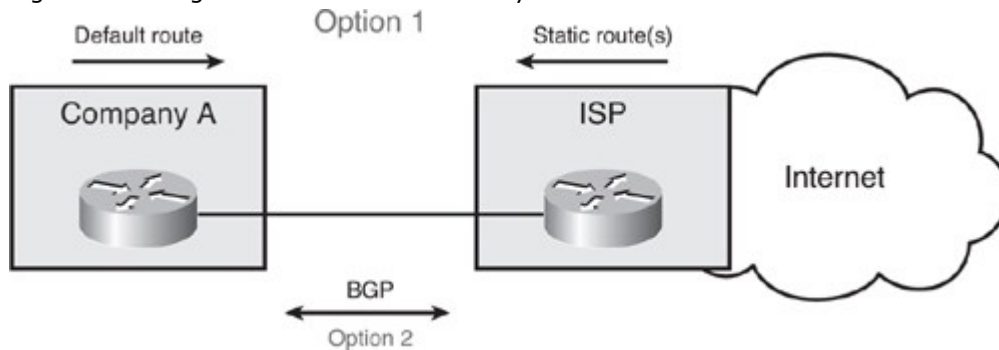
The ISP connection can also be made redundant. A customer can be connected to a single ISP or to multiple ISPs. There are various names for these different types of connections:

- With a connection to a single ISP when no link redundancy is used, the customer is *single homed*. If the ISP network fails, connectivity to the Internet is interrupted.
- With a connection to a single ISP, redundancy can be achieved if two links toward the same ISP are used effectively. This is called being *dual homed*.
- With connections to multiple ISPs, redundancy is built into the design. A customer connected to multiple ISPs is said to be *multihomed*, and is thus resistant to a single ISP failure.
- To enhance the resiliency further with connections to multiple ISPs, a customer can have two links toward each ISP. This solution is called being *dual multihomed*.

#### Single-Homed ISP Connectivity

Figure 6-7 is an example of single-homed ISP connectivity, used in cases when a loss in Internet connectivity is not problematic for a customer.

Figure 6-7. Single-Homed ISP Connectivity.



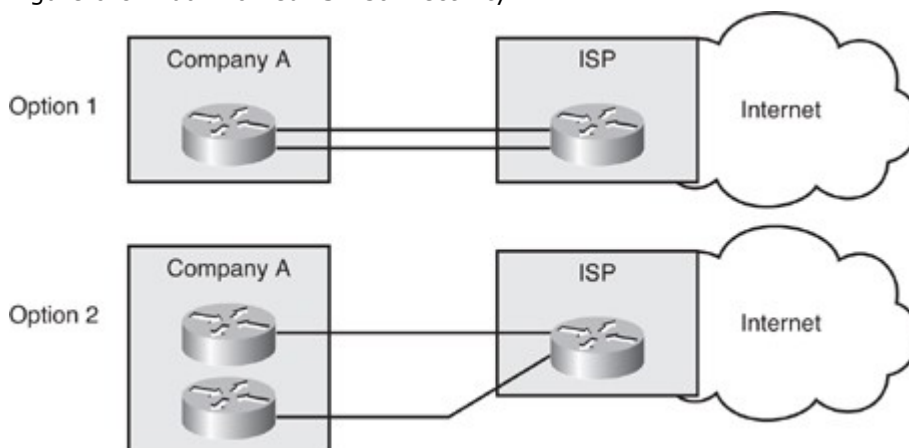
In this case, the customer uses a single connection to a single ISP. The connection type depends on the ISP offering, and can be, for example, a leased line, xDSL, or Ethernet. A failure of the link results in a no Internet connectivity.

As shown in Figure 6-7, single-homed Internet access does not require BGP. Rather, static routes are typically used, with a static default route from the customer to the ISP, and static routes in the ISP pointing toward customer networks (shown as Option 1 in the figure). If BGP is used (shown as Option 2 in the figure), the customer uses BGP to dynamically announce its public networks to the ISP, and the ISP announces only a default route to the customer, since that is sufficient to provide connectivity.

#### Dual-Homed ISP Connectivity

As illustrated in Figure 6-8, there are two options for dual homing when a customer has connections to a single ISP. Both links can be connected to one customer router (shown as Option 1 in the figure), or to enhance the resiliency further, the two links can terminate at separate routers in the customer's network (shown as Option 2 in the figure). In either case, routing must be properly configured to allow both links to be used.

Figure 6-8. Dual-Homed ISP Connectivity.



Depending on the SLA signed with the ISP, the routing deployed could achieve either of the following:

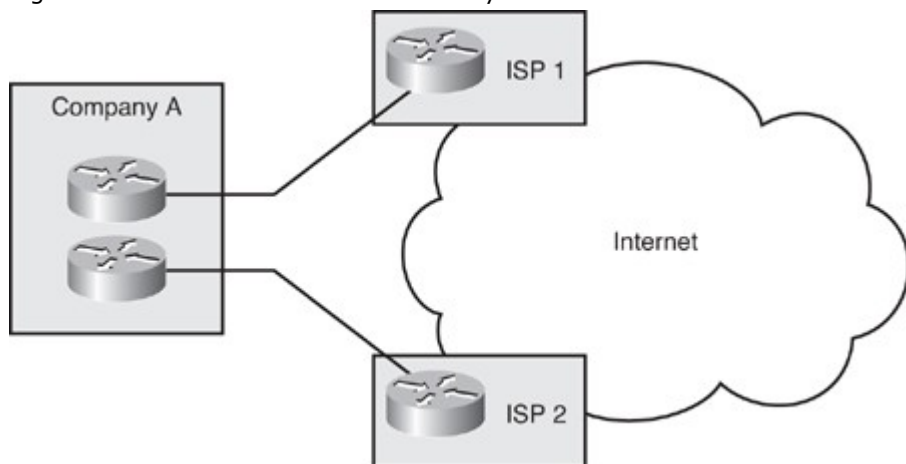
- Primary and backup link functionality where a single primary link is used to forward and receive traffic to and from ISP, and the secondary link is used only when the first one fails
- Load sharing between the links (achieved with Cisco Express Forwarding [CEF] switching)

In both cases, routing can be either static or dynamic (typically BGP).

#### Multihomed ISP Connectivity

Figure 6-9 illustrates a company connected to two different ISPs. The benefits of doing so include the following:

Figure 6-9. Multihomed ISP Connectivity.



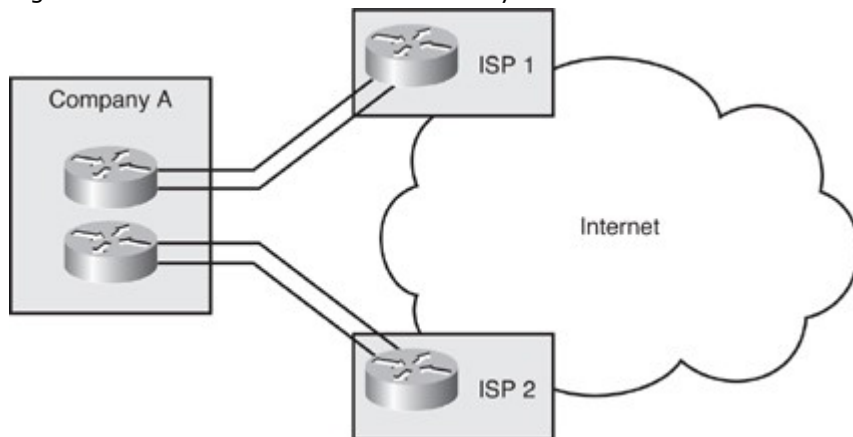
- Resistance to a failure beyond a directly connected link to a single ISP.
- Load sharing for different destination networks between ISPs, based on the network proximity.
- Scalability of the solution, beyond two ISPs.
- Achieving an ISP-independent solution. For example, although an ISP change would require an update to the routing and link configuration, and changing the link, the public IP address space used would remain the same.

Connections from different ISPs can terminate on the same router, or on different routers to further enhance the resiliency. The routing must be capable of reacting to dynamic changes; BGP is typically used.

#### Dual-Multihomed ISP Connectivity

Figure 6-10 illustrates a company that is dual multihomed, connected to two (or more) different ISPs with dual links per ISP. This configuration typically has multiple edge routers, one per ISP, and uses BGP.

Figure 6-10. Multihomed ISP Connectivity.



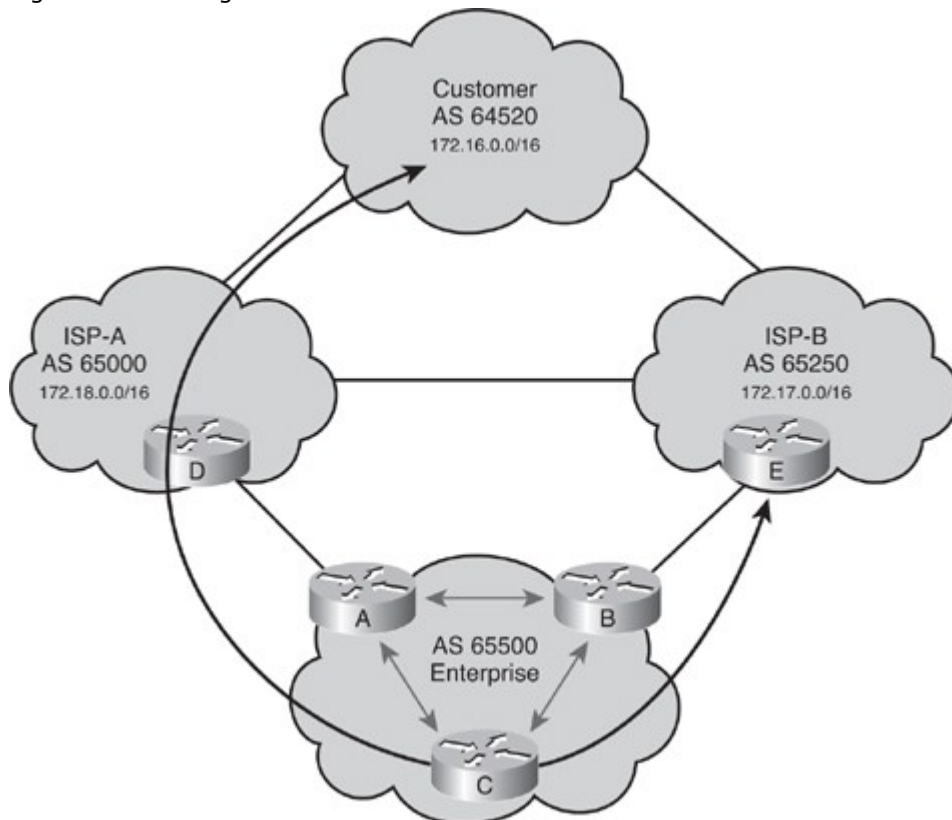
Dual-multihomed connectivity includes all the benefits of multihomed connectivity, with enhanced resiliency. Using BGP in an Enterprise Network

The Internet is a collection of autonomous systems that are interconnected to allow communication among them. BGP provides the routing between these autonomous systems.

Enterprises that want to connect to the Internet do so through one or more ISPs. If your organization has only one connection to one ISP, you probably do not need to use BGP. Instead you would use a default route. If you have multiple connections to one or to multiple ISPs, however, BGP might be appropriate because it allows manipulation of path attributes, facilitating selection of the optimal path.

When BGP is running between routers in different autonomous systems, it is called External BGP (EBGP). When BGP is running between routers in the same autonomous system, it is called Internal BGP (IBGP). Understanding how BGP works is important to avoid creating problems for your autonomous system as a result of running BGP. For example, enterprise autonomous system 65500 in Figure 6-11 is learning routes from both ISP-A and ISP-B via EBGP and is also running IBGP on all of its routers. Autonomous system 65500 learns about routes and chooses the best way to each one based on the configuration of the routers in the autonomous system and the BGP routes passed from the ISPs. If one of the connections to the ISPs goes down, traffic will be sent through the other ISP.

Figure 6-11. Using BGP to Connect to the Internet.



One of the routes that autonomous system 65500 learns from ISP-A is the route to 172.18.0.0/16. If that route is passed through autonomous system 65500 using IBGP and is mistakenly announced to ISP-B, ISP-B might decide that the best way to get to 172.18.0.0/16 is through autonomous system 65500, instead of through the Internet. Autonomous system 65500 would then be considered a transit autonomous system (an ISP); this would be a very undesirable situation. Autonomous system 65500 wants to have a redundant Internet connection, but does not want to act as a transit autonomous system between ISP-A and ISP-B. Careful BGP configuration is required to avoid this situation.

#### BGP Multihoming Options

As discussed earlier, multihoming is when an autonomous system has more than one connection to the Internet, via multiple ISPs (either multihoming and dual multihoming). Two typical reasons for multihoming are as follows:

- **To increase the reliability of the connection to the Internet**— If one connection fails, the other connection remains available.
- **To increase the performance of the connection**— Better paths can be used to certain destinations.

Note

Some Cisco documentation uses the term multihoming to refer to an autonomous system that has one or more connections to the Internet, whether via a single ISP or multiple ISPs. For consistency here, we distinguish between the various types of "homing" as described in the "Connection Redundancy" section, earlier in this chapter.

The benefits of BGP are apparent when an autonomous system has multiple EBGP connections to either a single ISP (which is called dual homed) or to multiple ISPs (one of the multihomed types). Having multiple connections enables an organization to have redundant connections to the Internet so that if a single path becomes unavailable, connectivity can still be maintained.

An organization can be connected to either a single ISP or to multiple ISPs. A drawback to having all your connections to a single ISP is that connectivity issues in that single ISP can cause your autonomous system to lose connectivity to the Internet. By having connections to multiple ISPs, an organization gains the following benefits:

- Has redundancy with the multiple connections
- Is not tied into the routing policy of a single ISP
- Has more paths to the same networks for better policy manipulation

A multihomed autonomous system will run EBGP with its external neighbors and might also run IBGP internally.

If an organization has determined that it will perform multihoming with BGP, three common ways to do this are as follows:

- **Each ISP passes only a default route to the autonomous system**— The default route is passed to the internal routers.
- **Each ISP passes only a default route and provider-owned specific routes to the autonomous system**— These routes can be passed to internal routers, or all internal routers in the transit path can run BGP and pass these routes between them.
- **Each ISP passes all routes to the autonomous system**— All internal routers in the transit path run BGP and pass these routes between them.

The sections that follow describe these options in greater detail.

#### Multihoming with Default Routes from All Providers

The first multihoming option is to receive only a default route from each ISP. This configuration requires the least resources within the autonomous system because a default route is used to reach any external destinations. The autonomous system sends all of its routes to the ISPs, which process and pass the routes on to other autonomous systems.

In this scenario, both edge routers learn a default route from their attached ISP and propagate the default route into their routing domain using the local IGP. If a router within the autonomous system learns about multiple default routes using the local IGP, it installs the best default route into its routing table. From the perspective of this router, it takes the default route with the least-cost IGP metric. This IGP default route will route packets that are destined to the external networks to an edge router of this autonomous system, which is running EBGP with the ISPs. The edge router will use the BGP default route to reach all external networks. The route that inbound packets take to reach the autonomous system is decided outside of the autonomous system (within the ISPs and other autonomous systems).

Regional ISPs that have multiple connections to national or international ISPs commonly implement this option. The regional ISPs do not need to use BGP for path manipulation. However, they do require the capability of adding new customers and the networks of the customers; BGP is ideal for this purpose. Consider if the regional ISP did not use BGP: Each time that the regional ISP needed to add a new set of networks, the customers would have to wait until the national ISPs added these networks to their BGP process and added static routes pointing to the regional ISP. Instead, by running EBGP with the national or international ISPs, the regional ISP needs only to add the new customer networks to its BGP process. These new networks automatically propagate across the Internet with minimal delay.

A customer that chooses to receive default routes from all providers must understand the following limitations of this option:

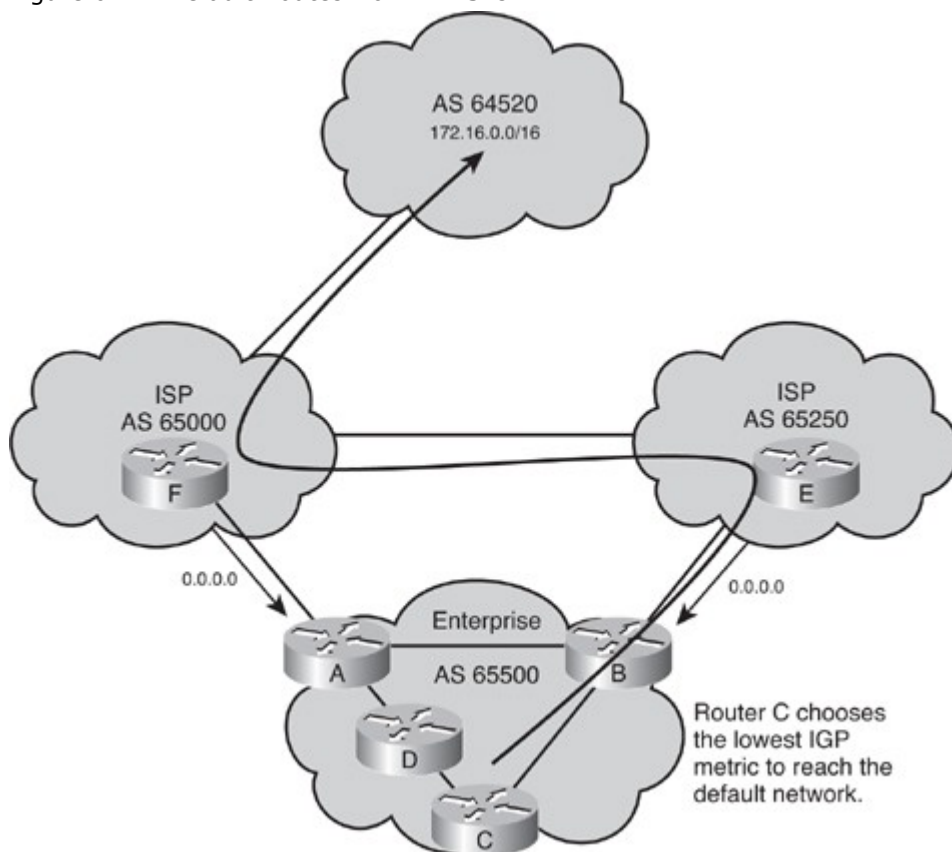
- Path manipulation cannot be performed because only a single route is being received from each ISP.
- Bandwidth manipulation is extremely difficult and can be accomplished only by manipulating the IGP metric of the default route.
- Diverting some of the traffic from one exit point to another is challenging because all destinations are using the same default route for path selection.

Note

Policy-based routing, as described in Chapter 5, could be used to select the ISP to which specific traffic should be routed.

Figure 6-12 illustrates an example. ISP autonomous system 65000 and ISP autonomous system 65250 send default routes into Enterprise autonomous system 65500. The ISP that a specific router within autonomous system 65500 uses to reach any external address is decided by the IGP metric that is used to reach the default route within the autonomous system. For example, if autonomous system 65500 uses RIP, Router C selects the route with the lowest hop count to the default route when sending packets to network 172.16.0.0. In this example, this appears to be suboptimal routing; the packets will go to ISP autonomous system 65250, but a more direct route is via ISP autonomous system 65000.

Figure 6-12. Default Routes from All ISPs.



#### Multihoming with Default Routes and Partial Table from All Providers

In the second design option for multihoming, all ISPs pass default routes plus select specific routes to the autonomous system.

An enterprise that is running EBGP with an ISP and that wants a partial routing table generally receives the networks that the ISP and its other customers own. The enterprise can also receive the routes from any other autonomous system.

Major ISPs are assigned large blocks of addresses (for example, they may get 2000 to 10,000 CIDR blocks of IP addresses, depending on their needs) from their RIR (which in turn gets address blocks from the IANA). The ISPs reassign their address blocks to their customers. If the ISP passes this information to a customer that wants only a partial BGP routing table, the customer could pass this information to internal routers using IBGP and might redistribute these routes into its IGP.

#### Caution

Redistributing a large number of BGP routes into an IGP could result in many issues. It is mentioned here only as something that *could* be done, but as you will see in this chapter, redistributing a large number of routes into an IGP is usually not the recommended solution.

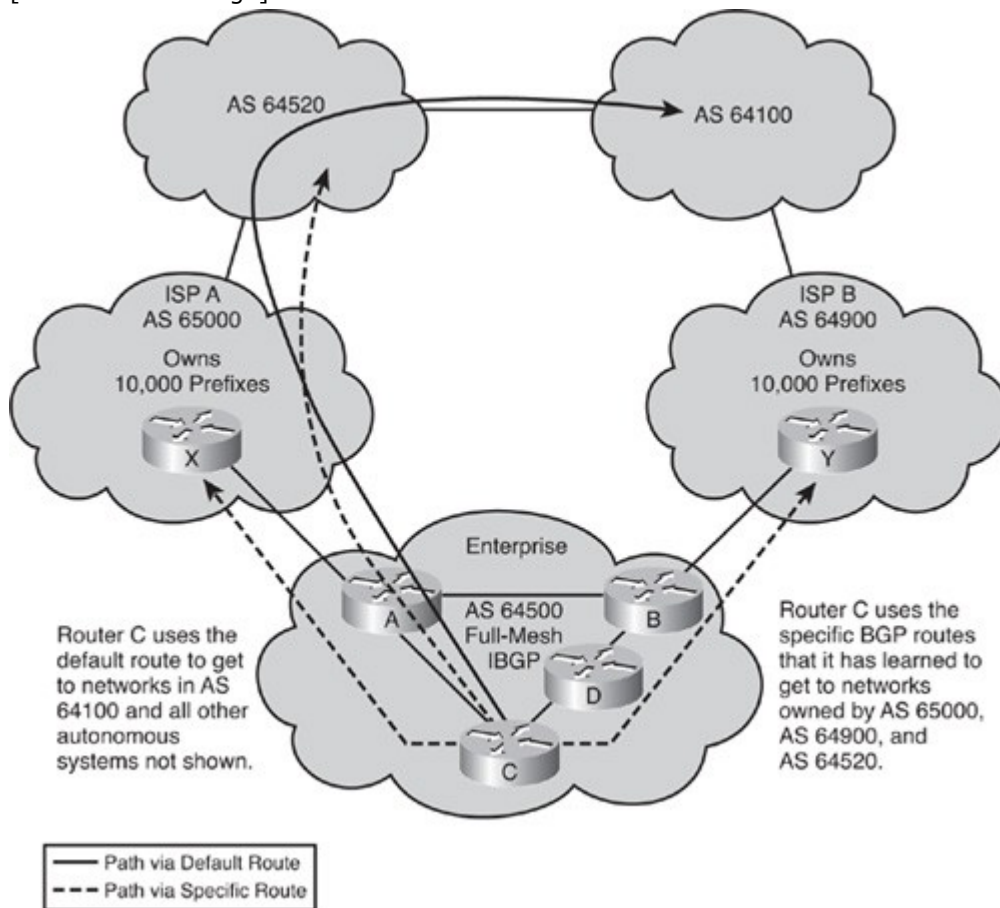
The internal routers of the customer (these routers are not running BGP) could then receive these routes via redistribution. They would take the nearest exit point based on the best metric of specific networks instead of taking the nearest exit point based on the default route.

Acquiring a partial BGP table from each provider would be beneficial for specific routes because path selection will be more predictable than when using a default route.

Figure 6-13 illustrates an example. ISP A autonomous system 65000 and ISP B autonomous system 64900 send default routes and the routes that each ISP owns to Enterprise autonomous system 64500. The enterprise (autonomous system 64500) asked both providers to also send routes to networks in autonomous system 64520 because of the amount of traffic between autonomous system 64520 and autonomous system 64500.

Figure 6-13. Default Routes and Partial Table from All ISPs.

[View full size image]



By running IBGP between the internal routers within autonomous system 64500 (or at least those along the transit path within the autonomous system), autonomous system 64500 can choose the optimal path to reach the customer networks (autonomous system 64520 in this case). (Where IBGP should be run is described in detail in the upcoming "IBGP on All Routers in a Transit Path" section in this chapter.) The routes to autonomous system 64100 and to other autonomous systems (not shown in the figure) that are not specifically advertised to autonomous system 64500 by ISP A and ISP B are decided by the IGP metric that is used to reach the default route within the autonomous system this may again result in suboptimal routing to these destinations.

#### Multihoming with Full Routes from All Providers

In the third multihoming option, all ISPs pass all routes to the autonomous system, and IBGP is run on at least all the routers in the transit path in this autonomous system. This option allows the internal routers of the autonomous system to take the path through the best ISP for each route.

This configuration requires a lot of resources within the autonomous system because it must process all the external routes.

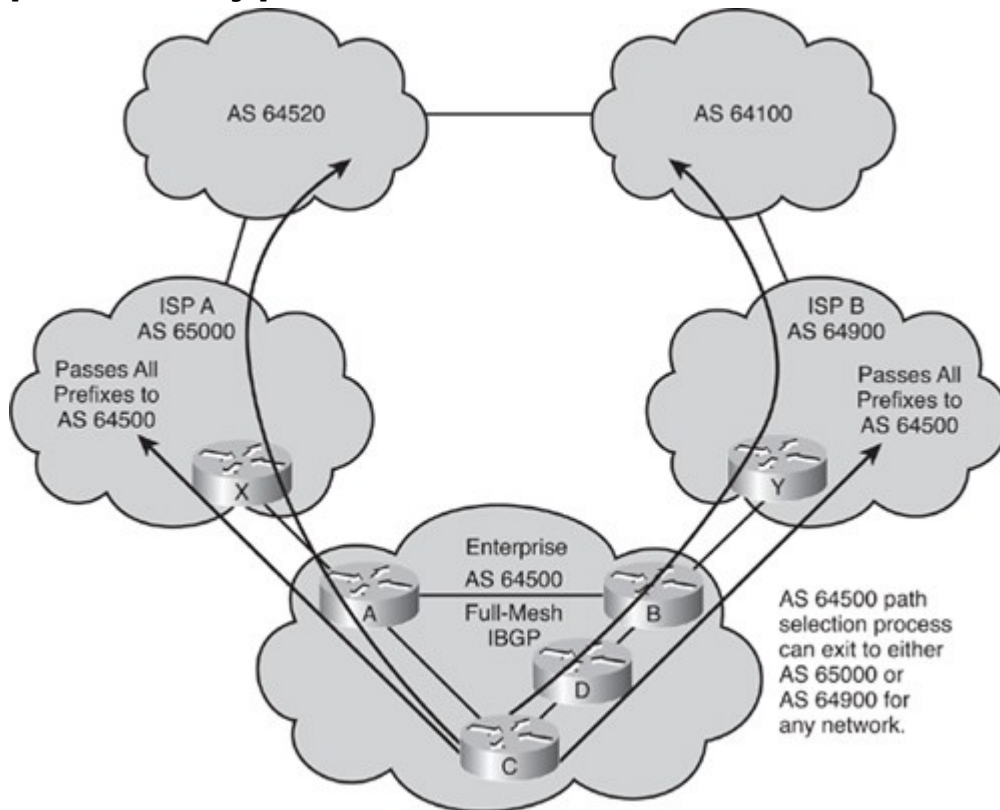


The autonomous system sends all of its routes to the ISPs, which process the routes and pass them to other autonomous systems.

Figure 6-14 illustrates an example. ISP A autonomous system 65000 and ISP B autonomous system 64900 send all routes into Enterprise autonomous system 64500. The ISP that a specific router within autonomous system 64500 uses to reach the external networks is determined by BGP. The routers in autonomous system 64500 can be configured to influence the path to certain networks. For example, Router A and Router B can influence the outbound traffic from autonomous system 64500.

Figure 6-14. Full Routes from All ISPs.

[View full size image]

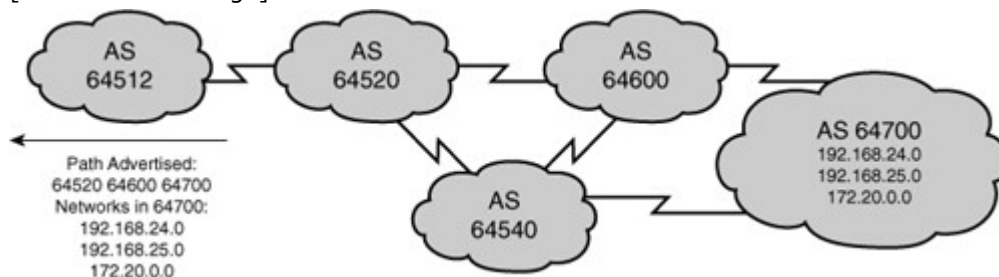


### BGP Path Vector Characteristics

Internal routing protocols announce a list of networks and the metrics to get to each network. In contrast, BGP routers exchange network reachability information, called path vectors, made up of path attributes, as illustrated in Figure 6-15. The path vector information includes a list of the full path of BGP autonomous system numbers (hop by hop) necessary to reach a destination network. Other attributes include the IP address to get to the next autonomous system (the next-hop attribute) and how the networks at the end of the path were introduced into BGP (the origin code attribute). The "BGP Attributes" section, later in this chapter, describes all the BGP attributes in detail.

Figure 6-15. BGP Uses Path Vector Routing.

[View full size image]





This autonomous system path information is used to construct a graph of loop-free autonomous systems and to identify routing policies so that restrictions on routing behavior can be enforced based on the autonomous system path.

The BGP autonomous system path is guaranteed to always be loop free: A router running BGP does not accept a routing update that already includes its autonomous system number in the path list, because the update has already passed through its autonomous system, and accepting it again will result in a routing loop.

BGP is designed to scale to huge internetworks, such as the Internet.

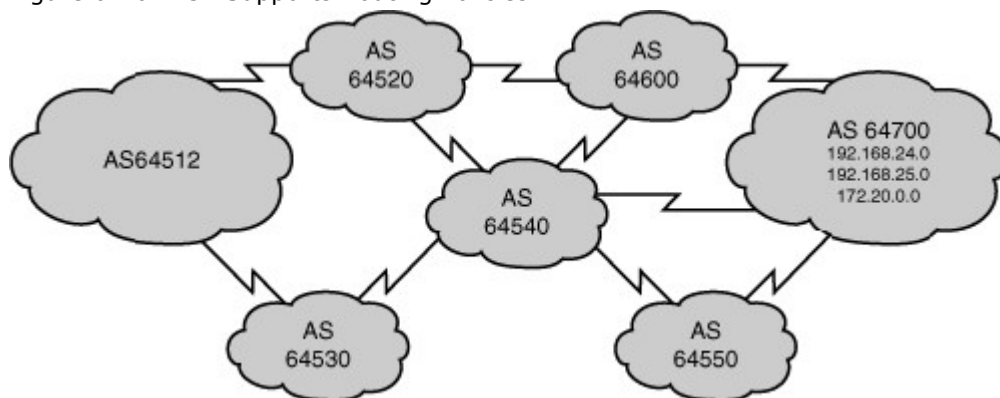
BGP allows routing-policy decisions to be applied to the path of BGP autonomous system numbers so that routing behavior can be enforced at the autonomous system level and to determine how data will flow through the autonomous system. These policies can be implemented for all networks owned by an autonomous system, for a certain CIDR block of network numbers (prefixes), or for individual networks or subnetworks. The policies are based on the attributes carried in the routing information and configured on the routers.

BGP specifies that a BGP router can advertise to its peers (neighbors) in neighboring autonomous systems only those routes that it uses. This rule reflects the hop-by-hop routing paradigm generally used throughout the current Internet. Some policies cannot be supported by the hop-by-hop routing paradigm. For example, BGP does not allow one autonomous system to send traffic to a neighboring autonomous system, intending that the traffic take a different route from that taken by traffic originating in that neighboring autonomous system. In other words, you cannot influence how a neighboring autonomous system will route your traffic, but you can influence how your traffic gets to a neighboring autonomous system. However, BGP can support any policy conforming to the hop-by-hop routing paradigm.

Because the current Internet uses only the hop-by-hop routing paradigm, and because BGP can support any policy that conforms to that paradigm, BGP is highly applicable as an inter-autonomous system routing protocol for the current Internet.

For example, in Figure 6-16, the following are some of the paths possible for autonomous system 64512 to reach networks in autonomous system 64700, through autonomous system 64520:

Figure 6-16. BGP Supports Routing Policies.



- 64520 64600 64700
- 64520 64600 64540 64550 64700
- 64520 64540 64600 64700
- 64520 64540 64550 64700

Autonomous system 64512 does not see all these possibilities. Autonomous system 64520 advertises to autonomous system 64512 only its best path, in this case 64520 64600 64700, the same way that IGP's announce only their best least-cost routes. This path is the only path through autonomous system 64520 that autonomous system 64512 sees. All packets that are destined for 64700 via 64520 take this path, because it is the autonomous system-by-autonomous system (hop-by-hop) path that autonomous system 64520 uses to reach the networks in autonomous system 64700. Autonomous system 64520 does not announce the other paths, such as 64520 64540 64600 64700, because it does not choose any of those paths as the best path, based on the BGP routing policy in autonomous system 64520.

Autonomous system 64512 does not learn of the second-best path, or any other paths from 64520, unless the best path through autonomous system 64520 becomes unavailable.

Even if autonomous system 64512 were aware of another path through autonomous system 64520 and wanted to use it, autonomous system 64520 would not route packets along that other path, because

autonomous system 64520 selected 64520 64600 64700 as its best path, and all autonomous system 64520 routers will use that path as a matter of BGP policy. BGP does not let one autonomous system send traffic to a neighboring autonomous system, intending that the traffic take a different route from that taken by traffic originating in the neighboring autonomous system.

To reach the networks in autonomous system 64700, autonomous system 64512 can choose to use the path through autonomous system 64520 or it can choose to go through the path that autonomous system 64530 is advertising. Autonomous system 64512 selects the best path to take based on its own BGP routing policies.

#### When to Use BGP

BGP use in an autonomous system is most appropriate when the effects of BGP are well understood and at least one of the following conditions exists:

- The autonomous system allows packets to transit through it to reach other autonomous systems (for example, it is a service provider).
- The autonomous system has multiple connections to other autonomous systems.
- Routing policy and route selection for traffic entering and leaving the autonomous system must be manipulated.

If an enterprise wants its traffic to be differentiated from its ISP's traffic on the Internet, the enterprise must connect to its ISP using BGP. If, instead, an enterprise is connected to its ISP with a static route, traffic from that enterprise on the Internet is indistinguishable from traffic from the ISP.

BGP was designed to allow ISPs to communicate and exchange packets. These ISPs have multiple connections to one another and have agreements to exchange updates. BGP is the protocol that is used to implement these agreements between two or more autonomous systems. If BGP is not properly controlled and filtered, it has the potential to allow an outside autonomous system to affect the traffic flow to your autonomous system. For example, if you are a customer connected to ISP A and ISP B (for redundancy), you want to implement a routing policy to ensure that ISP A does not send traffic to ISP B via your autonomous system. You want to be able to receive traffic destined for your autonomous system through each ISP, but you do not want to waste valuable resources and bandwidth within your autonomous system to route traffic for your ISPs. This chapter focuses on how BGP operates and how to configure it properly to prevent this from happening.

#### When Not to Use BGP

BGP is not always the appropriate solution to interconnect autonomous systems. For example, if there is only one exit path from the autonomous system, a default or static route is appropriate. Using BGP will not accomplish anything except to use router CPU resources and memory. If the routing policy that will be implemented in an autonomous system is consistent with the policy implemented in the ISP autonomous system, it is not necessary or even desirable to configure BGP in that autonomous system. The only time BGP will be required is when the local policy differs from the ISP policy.

Do not use BGP if one or more of the following conditions exist:

- A single connection to the Internet or another autonomous system
- Lack of memory or processor power on edge routers to handle constant BGP updates
- You have a limited understanding of route filtering and the BGP path-selection process

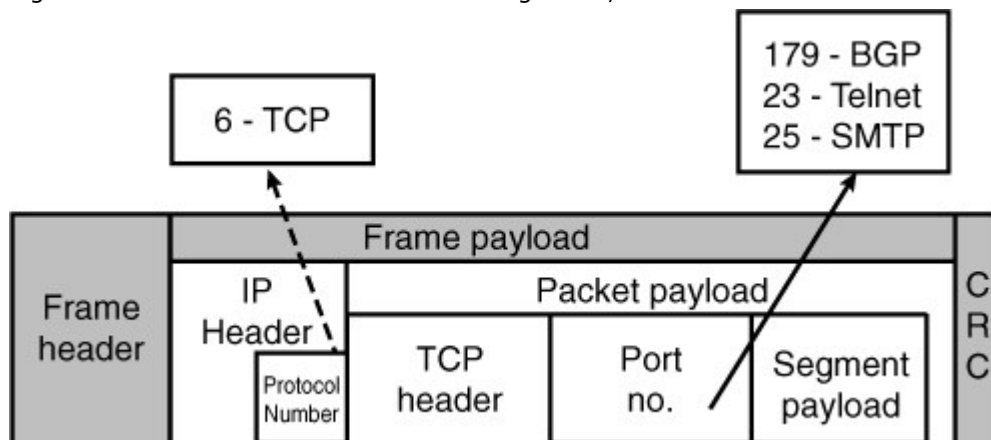
In these cases, use static or default routes instead, as discussed in Chapter 1.

#### BGP Characteristics

What type of protocol is BGP? Chapter 1 covers the characteristics of distance vector and link-state routing protocols. BGP is sometimes categorized as an advanced distance vector protocol, but it is actually a path vector protocol. BGP has many differences from standard distance vector protocols, such as RIP.

BGP uses the TCP as its transport protocol, which provides connection-oriented reliable delivery. In this way, BGP assumes that its communication is reliable and, therefore, BGP does not have to implement any retransmission or error-recovery mechanisms, like EIGRP does. BGP information is carried inside TCP segments using protocol 179; these segments are carried inside IP packets. Figure 6-17 illustrates this concept.

Figure 6-17. BGP Is Carried Inside TCP Segments, Which Are Inside IP Packets.



#### Note

BGP is the only IP routing protocol to use TCP as its transport layer. OSPF and EIGRP reside directly above the IP layer. IS-IS is at the network layer. RIP uses the User Datagram Protocol (UDP) for its transport layer.

Two routers speaking BGP establish a TCP connection with one another and exchange messages to open and confirm the connection parameters. These two routers are called BGP *peer* routers or BGP *neighbors*.

After the TCP connection is made, the routers exchange their full BGP routing tables (described later in the "BGP Tables" section). However, because the connection is reliable, BGP routers need to send only changes (incremental updates) after that. Periodic routing updates are not required on a reliable link, so triggered updates are used. BGP sends keepalive messages, similar to the hello messages sent by OSPF, IS-IS, and EIGRP.

OSPF and EIGRP have their own internal functions to ensure that update packets are explicitly acknowledged. These protocols use a one-for-one window so that if either OSPF or EIGRP has multiple packets to send, the next packet cannot be sent until an acknowledgment from the first update packet is received. This process can be very inefficient and cause latency issues if thousands of update packets must be exchanged over relatively slow serial links. However, OSPF and EIGRP rarely have thousands of update packets to send. For example, EIGRP can hold more than 100 networks in one EIGRP update packet, so 100 EIGRP update packets can hold up to 10,000 networks. Most organizations do not have 10,000 subnets.

BGP, on the other hand, has more than 300,000 networks (and growing) on the Internet to advertise, and it uses TCP to handle the acknowledgment function. TCP uses a dynamic window, which allows for up to 65,536 bytes to be outstanding before it stops and waits for an acknowledgment. For example, if 1000-byte packets are being sent and the maximum window size is being used, BGP would have to stop and wait for an acknowledgment only when 65 packets had not been acknowledged.

#### Note

The CIDR report, at <http://www.cidr-report.org/>, is a good reference site to see the current size of the Internet routing tables and other related information.

TCP is designed to use a sliding window, where the receiver sends an acknowledgment before the number of octets specified by the window have been received (such as at the halfway point of the sending window). This method allows any TCP application, such as BGP, to continue streaming packets without having to stop and wait, as OSPF or EIGRP would require.

#### BGP Neighbor Relationships

No single router can handle communications with the tens of thousands of the routers that run BGP and are connected to the Internet, representing more than 33,000 autonomous systems. A BGP router forms a direct neighbor relationship with a limited number of other BGP routers. Through these BGP neighbors, a BGP router learns of the paths through the Internet to reach any advertised network.

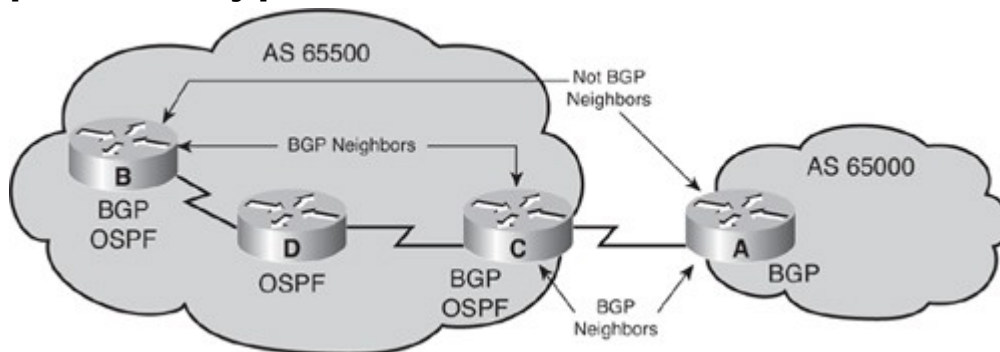
Any router that runs BGP is called a BGP speaker.

A BGP peer, also known as a BGP neighbor, is a BGP speaker that is configured to form a neighbor relationship with another BGP speaker for the purpose of directly exchanging BGP routing information with one another.

A BGP speaker has a limited number of BGP neighbors with which it peers and forms a TCP-based relationship, as illustrated in Figure 6-18. BGP peers can be either internal or external to the autonomous system.

Figure 6-18. Routers That Have Formed a BGP Connection Are BGP Neighbors or Peers.

[View full size image]



#### Note

A BGP peer must be configured under the BGP process with a **neighbor** command. This command instructs the BGP process to establish a relationship with the neighbor at the address listed in the command and to exchange BGP routing updates with that neighbor.

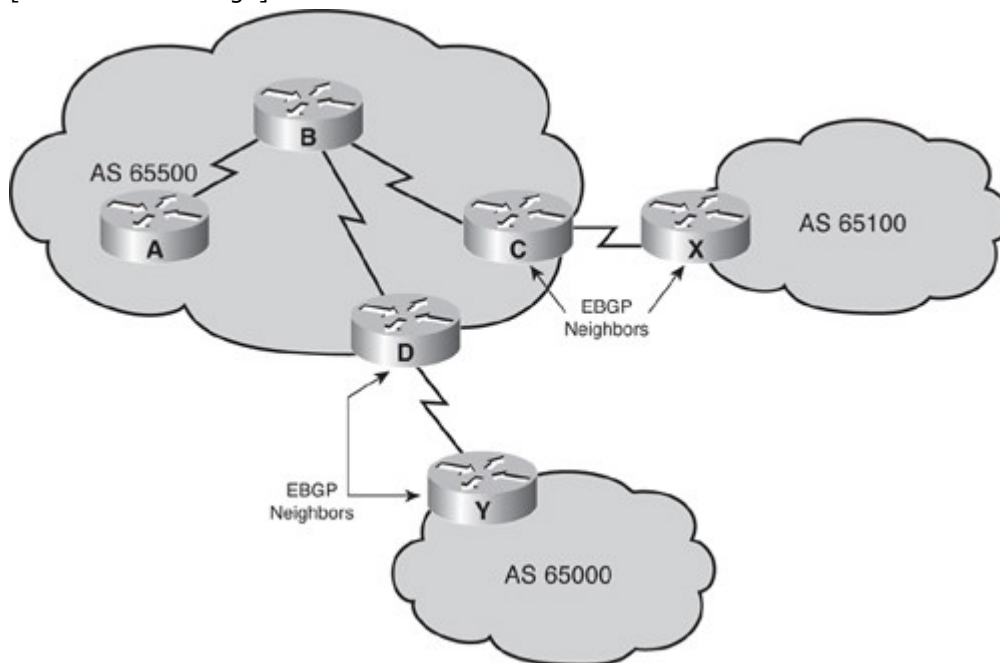
BGP configuration is described later, in the “Configuring BGP” section.

#### External BGP Neighbors

Recall that when BGP is running between routers in different autonomous systems, it is called EBGP. Routers running EBGP are usually directly connected to each other, as shown in Figure 6-19.

Figure 6-19. EBGP Neighbors Belong to Different Autonomous Systems.

[View full size image]



An EBGP neighbor is a router running in a different autonomous system. An IGP is not run between the EBGP neighbors. For two routers to exchange BGP routing updates, the TCP reliable transport layer on each side must successfully pass the TCP three-way handshake before the BGP session can be established. Therefore, the IP address used in the **neighbor** command must be reachable without using an IGP. This can be

accomplished by pointing at an address that can be reached through a directly connected network or by using static routes to that IP address. Generally, the neighbor address used is the address of the directly connected network.

An enterprise network can have a connection to one or several ISPs, and the ISPs themselves might be connected to several other ISPs. For each such connection between different autonomous systems, there is an EBGp session required between EBGp neighboring routers. In Figure 6-19, an EBGp relationship is established between Routers D and Y, and another EBGp relationship is established between Routers C and X. There are several requirements for EBGp neighborship:

- **Different autonomous system number**— EBGp neighbors must reside in different autonomous systems to be able to form an EBGp relationship.
- **Define neighbors**— A TCP session must be established before starting BGP routing update exchanges.
- **Reachability**— The IP addresses used in the **neighbor** command must be reachable; EBGp neighbors are usually directly connected.

#### Internal BGP Neighbors

Recall that when BGP is running between routers within the same autonomous system, it is called IBGP. IBGP is run within an autonomous system to exchange BGP information so that all internal BGP speakers have the same BGP routing information about outside autonomous systems and so this information can be passed to other autonomous systems.

There are several requirements for IBGP neighborship:

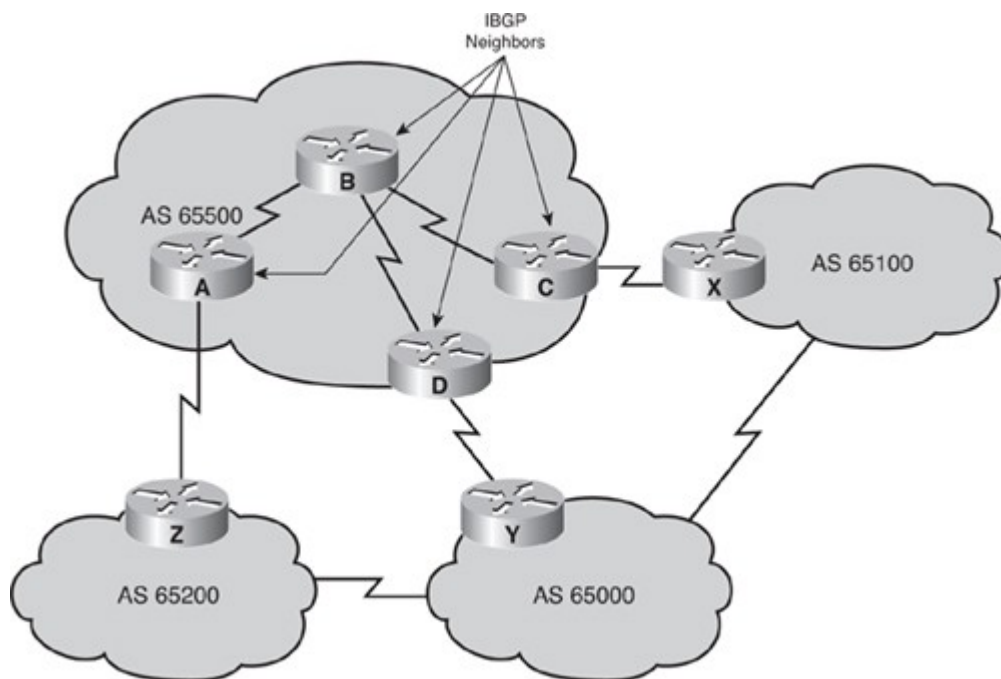
- **Same autonomous system number**— IBGP neighbors must reside in the same autonomous system to be able to form an IBGP relationship.
- **Define neighbors**— A TCP session must be established between neighbors before they start exchanging BGP routing updates.
- **Reachability**— IBGP neighbors must be reachable. An IGP typically runs inside the autonomous system.

Routers running IBGP do not have to be directly connected to each other, as long as they can reach each other so that TCP handshaking can be performed to set up the BGP neighbor relationships. The IBGP neighbor can be reached by a directly connected network, static routes, or an internal routing protocol. Because multiple paths generally exist within an autonomous system to reach other routers, a loopback address is usually used in the BGP **neighbor** command to establish the IBGP sessions.

For example, in Figure 6-20, Routers A, D, and C learn the paths to the external autonomous systems from their respective EBGp neighbors (Routers Z, Y, and X). If the link between Routers D and Y goes down, Router D must learn new routes to the external autonomous systems. Other BGP routers within autonomous system 65500 that were using Router D to get to external networks must also be informed that the path through Router D is unavailable. Those BGP routers within autonomous system 65500 need to have the alternative paths through Routers A and C in their BGP topology database.

Figure 6-20. IBGP Neighbors Are in the Same Autonomous System.

[View full size image]



As described in the next section, you must set up IBGP sessions between all routers in the transit path in autonomous system 65500 so that each router in the transit path within the autonomous system learns about paths to the external networks via IBGP.

#### IBGP on All Routers in a Transit Path

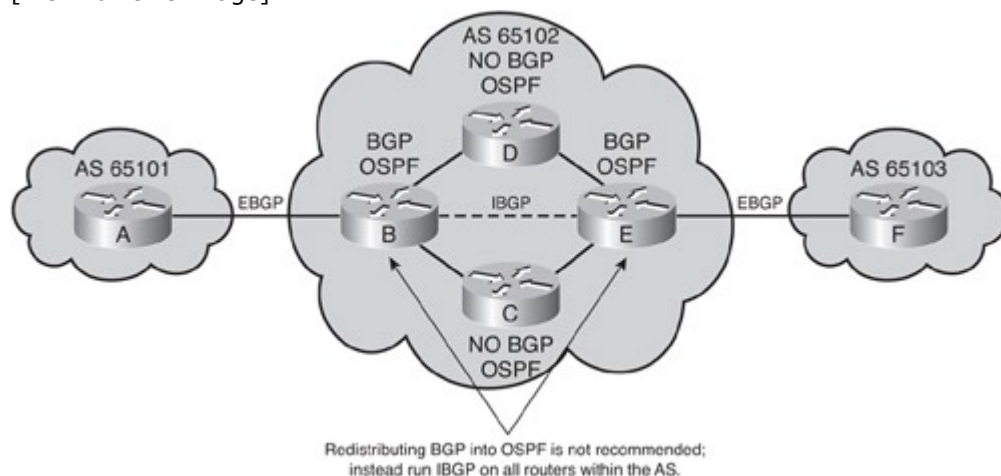
This section explains why IBGP route propagation requires all routers in the transit path in an autonomous system to run IBGP.

#### IBGP in a Transit Autonomous System

BGP was originally intended to run along the borders of an autonomous system, with the routers in the middle of the autonomous system ignorant of the details of BGP—hence the name Border Gateway Protocol. A transit autonomous system, such as autonomous system 65102 in Figure 6-21, is an autonomous system that routes traffic from one external autonomous system to another external autonomous system. As mentioned earlier, transit autonomous systems are typically ISPs. All routers in a transit autonomous system must have complete knowledge of external routes. Theoretically, one way to achieve this goal is to redistribute BGP routes into an IGP at the edge routers; however, this approach has problems.

Figure 6-21. BGP in a Transit Autonomous System.

[View full size image]





Because the current Internet routing table is very large, redistributing all the BGP routes into an IGP is not a scalable way for the interior routers within an autonomous system to learn about the external networks. Another method that you can use is to run IBGP on all routers within the autonomous system.

#### IBGP in a Nontransit Autonomous System

A nontransit autonomous system, such as an organization that is multihoming with two ISPs, does not pass routes between the ISPs. To make proper routing decisions, however, the BGP routers within the autonomous system still require knowledge of all BGP routes passed to the autonomous system.

As discussed, BGP does not work in the same manner as IGPs. Because the designers of BGP could not guarantee that an autonomous system would run BGP on all routers, a method had to be developed to ensure that IBGP speakers could pass updates to one another while ensuring that no routing loops would exist.

To avoid routing loops within an autonomous system, BGP specifies that routes learned through IBGP are never propagated to other IBGP peers. Recall that the **neighbor** command enables BGP updates between BGP speakers. By default, each BGP speaker is assumed to have a **neighbor** statement for all other IBGP speakers in the autonomous system—this is known as *full-mesh IBGP*.

If the sending IBGP neighbor is not fully meshed with each IBGP router, the routers that are not peering with this router will have different IP routing tables than the routers that are peering with it. The inconsistent routing tables can cause routing loops or routing black holes, because the default assumption by all routers running BGP within an autonomous system is that each BGP router exchanges IBGP information directly with all other BGP routers in the autonomous system.

When all IBGP neighbors are fully meshed and a change is received from an external autonomous system, the receiving BGP router in the local autonomous system is responsible for informing all other IBGP neighbors of the change. IBGP neighbors that receive this update do not send it to any other IBGP neighbor because they assume that the sending IBGP neighbor is fully meshed with all other IBGP speakers and has sent each IBGP neighbor the update.

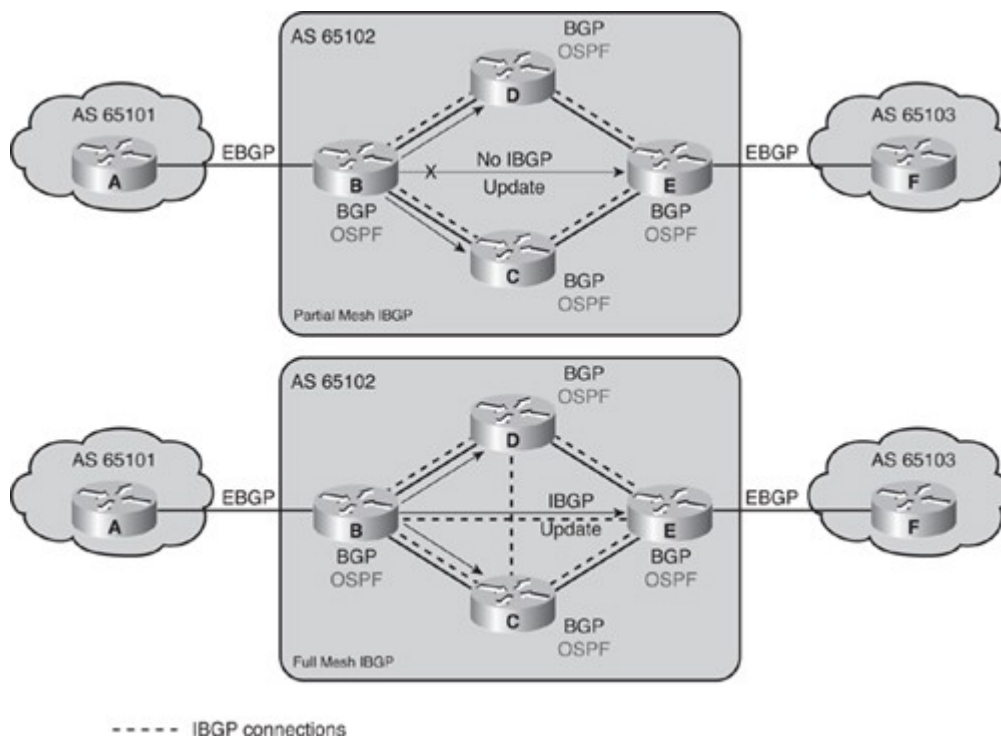
#### BGP Partial-Mesh and Full-Mesh Examples

The top network in Figure 6-22 illustrates IBGP update behavior in a partially meshed neighbor environment. Router B receives an EBGP update from Router A. Router B has two IBGP neighbors, Routers C and D, but does not have an IBGP neighbor relationship with Router E. Therefore, Routers C and D learn about any networks that were added or withdrawn behind Router B. Even if Routers C and D have IBGP neighbor sessions with Router E, they assume that the autonomous system is fully meshed for IBGP and do not replicate the update and send it to Router E. Sending the IBGP update to Router E is Router B's responsibility because it is the router with firsthand knowledge of the networks in and beyond autonomous system 65101. So, Router E does not learn of any networks through Router B and does not use Router B to reach any networks in autonomous system 65101 or other autonomous systems behind autonomous system 65101.

Figure 6-22. Partial-Mesh Versus Full-Mesh IBGP.

[View full size image]





In the lower portion of Figure 6-22, IBGP is fully meshed. When Router B receives an EBGP update from Router A, it updates all three of its IBGP peers, Router C, Router D, and Router E. OSPF, the IGP, is used to route the TCP segment containing the BGP update from Router A to Router E, because these two routers are not directly connected. The update is sent once to each neighbor and not duplicated by any other IBGP neighbor, which reduces unnecessary traffic. In fully meshed IBGP, each router assumes that every other internal router has a neighbor statement that points to each IBGP neighbor.

#### TCP and Full Mesh

TCP was selected as the transport layer for BGP because TCP can move a large volume of data reliably. With the very large full Internet routing table changing constantly, using TCP for windowing and reliability was determined to be the best solution, as opposed to developing a BGP one-for-one windowing capability like OSPF or EIGRP.

TCP sessions cannot be multicast or broadcast because TCP has to ensure the delivery of packets to each recipient. Because TCP cannot use broadcasting, BGP cannot use it either.

Because each IBGP router needs to send routes to all the other IBGP neighbors in the same autonomous system (so that they all have a complete picture of the routes sent to the autonomous system) and they cannot use broadcast, they must use fully meshed BGP (TCP) sessions. In other words, an IBGP neighbor relationship must be configured between each pair of routers.

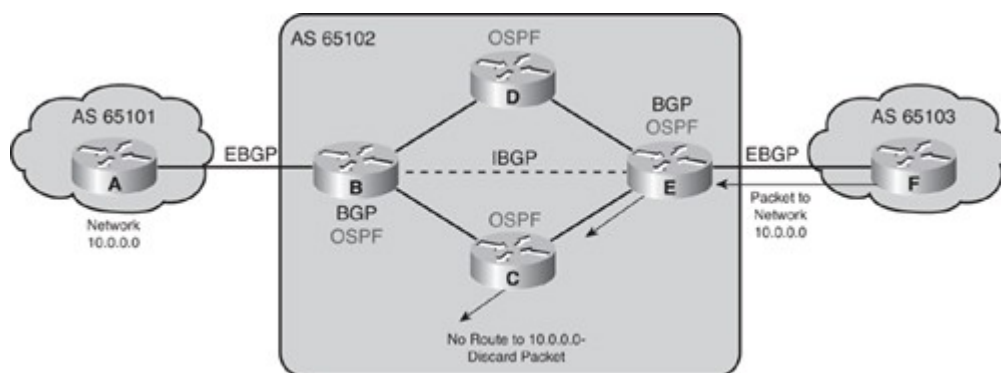
When all routers running BGP in an autonomous system are fully meshed and have the same database as a result of a consistent routing policy, they can apply the same path-selection formula. The path-selection results will therefore be uniform across the autonomous system. Uniform path selection across the autonomous system means no routing loops and a consistent policy for exiting and entering the autonomous system.

#### Routing Issues If BGP Not on in All Routers in a Transit Path

Figure 6-23 illustrates how routing might not work if all routers in a transit path are not running BGP.

Figure 6-23. Routing Might Not Work If BGP Is Not Run on All Routers in a Transit Path.

[View full size image]



In this example, Routers A, B, E, and F are the only ones running BGP. Router B has an EBGP **neighbor** statement for Router A and an IBGP **neighbor** statement for Router E. Router E has an EBGP **neighbor** statement for Router F and an IBGP **neighbor** statement for Router B. Routers C and D are not running BGP. Routers B, C, D and E are running OSPF as their IGP.

Network 10.0.0.0 is owned by autonomous system 65101 and is advertised by Router A to Router B via an EBGP session. Router B advertises it to Router E via an IBGP session. Routers C and D never learn about this network because it is not redistributed into the local routing protocol (OSPF in this example), and Routers C and D are not running BGP. If Router E advertises this network to Router F in autonomous system 65103, and Router F starts forwarding packets to network 10.0.0.0 through autonomous system 65102, where would Router E send the packets?

Router E would send the packets to its BGP peer, Router B. To get to Router B, however, the packets must go through Router C or D, but those routers do not have an entry in their routing tables for network 10.0.0.0. Thus, if Router E forwards packets with a destination address in network 10.0.0.0 to either Routers C or D, those routers discard the packets.

#### Note

BGP route reflectors are an alternative to running full-mesh IBGP and are discussed in Appendix C, "BGP Supplement."

Appendix C is available at this book's companion website <http://www.ciscopress.com/title/9781587058820>.

Even if Routers C and D have a default route going to the exit points of the autonomous system (Routers B and E), there is a good chance that when Router E sends a packet for network 10.0.0.0 to Routers C or D, those routers may send it back to Router E, which will forward it again to Routers C or D, causing a routing loop. To solve this problem, BGP must be implemented on Routers C and D.

In conclusion, it is important to remember that all routers in the path between IBGP neighbors within an autonomous system, known as the transit path, must also be running BGP. These IBGP sessions must be fully meshed.

#### BGP Synchronization

The BGP synchronization rule states that a BGP router should not use, or advertise to an external neighbor, a route learned by IBGP, unless that route is local or is learned from the IGP.

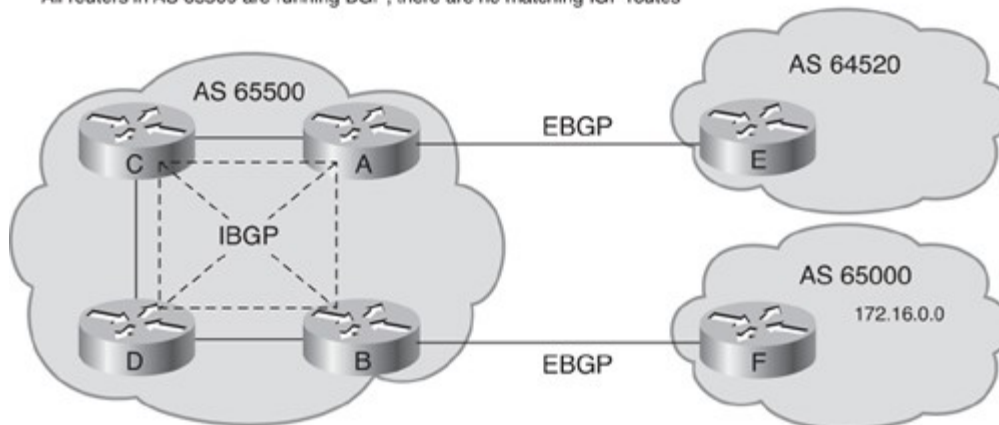
If there were a small enough number of BGP routes so that they could be redistributed into an IGP running in an autonomous system, IBGP would not be needed in every router in the transit path. However, synchronization would be needed to make sure that packets did not get lost. In the past, synchronization was on by default in the Cisco IOS. If synchronization is enabled and your autonomous system is passing traffic from one autonomous system to another, BGP should not advertise a route before all routers in your autonomous system have learned about the route via IGP. In other words, BGP and the IGP must be synchronized before networks learned from an IBGP neighbor can be used.

The modern Internet has too many routes in the BGP table to consider redistributing them into IGPs. The best practice is to not redistribute BGP into the IGP, but instead use IBGP on all routers in the transit path. In this case, synchronization is not needed. Therefore, it is now off by default in the Cisco IOS.

BGP synchronization is disabled by default in Cisco IOS Software Release 12.2(8)T and later. It was on by default in earlier Cisco IOS Software releases. With the default of synchronization disabled, BGP can use and advertise to external BGP neighbors routes learned from an IBGP neighbor that are not present in the local routing table. (BGP synchronization should only be disabled, though, if all routers in the transit path in the autonomous system are running full-mesh IBGP, for the reasons discussed in the previous section, or if the autonomous system is not a transit autonomous system.)

Figure 6-24 illustrates an example. Routers A, B, C, and D are all running IBGP and an IGP with each other (there is full-mesh IBGP). There are no matching IGP routes for the BGP routes (Routers A and B are not redistributing the BGP routes into the IGP). Routers A, B, C, and D have IGP routes to the internal networks of autonomous system 65500 but do not have IGP routes to external networks, such as 172.16.0.0.

All routers in AS 65500 are running BGP; there are no matching IGP routes



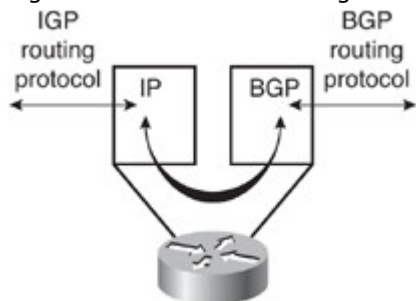
As described earlier, in modern autonomous systems, because the size of the Internet routing table is large, redistributing from BGP into an IGP is not scalable. Therefore, most modern autonomous systems run full-mesh IBGP and do not require synchronization. Some advanced BGP configuration methods, such as route

reflectors and confederations, reduce the IBGP full-mesh requirements. (As mentioned earlier, route reflectors are discussed in Appendix C.)

#### BGP Tables

As shown in Figure 6-25, a router running BGP keeps its own table for storing BGP information received from and sent to other routers.

Figure 6-25. Router Running BGP Keeps a BGP Table, Separate from the IP Routing Table.



This table of BGP information is known by many names in various documents, including the following:

- BGP table
- BGP topology table
- BGP topology database
- BGP routing table
- BGP forwarding database

It is important to remember that this BGP table is separate from the IP routing table in the router.

The router offers the best routes from the BGP table to the IP routing table and can be configured to share information between the two tables (by redistribution).

BGP also keeps a neighbor table containing a list of neighbors with which it has a BGP connection.

For BGP to establish an adjacency, you must configure it explicitly for each neighbor. BGP forms a TCP relationship with each of the configured neighbors and keeps track of the state of these relationships by periodically sending a BGP/TCP keepalive message.

#### Note

BGP sends BGP/TCP keepalives by default every 60 seconds.

After establishing an adjacency, the neighbors exchange their best BGP routes. Each router collects these routes from each neighbor with which it successfully established an adjacency and places them in its BGP forwarding database. All routes that have been learned from each neighbor are placed in the BGP forwarding database. The best routes for each network are selected from the BGP forwarding database using the BGP route-selection process (discussed in the section “The Route-Selection Decision Process,” later in this chapter) and then are offered to the IP routing table. (As described in the referenced section, one of the criteria for being selected as the best BGP route is that the next-hop IP address is reachable. Therefore, BGP routes with an unreachable next hop will not be propagated to other routers.)

#### Note

As described earlier, if synchronization is enabled, BGP will exchange routes only if those routes are also in the IP routing table. With the default of synchronization disabled, though, the routes do not have to be in the IP routing table for BGP to advertise them.

Each router compares the offered BGP routes to any other possible paths to those networks in its IP routing table, and the best route, based on administrative distance, is installed in the IP routing table. EBGP routes (BGP routes learned from an external autonomous system) have an administrative distance of 20. IBGP routes (BGP routes learned from within the autonomous system) have an administrative distance of 200.

A router may have a best BGP route to a destination, but that route might not be installed in the IP routing table because it has a higher administrative distance than another route. That best BGP route will still be propagated to other BGP routers, though.

#### BGP Message Types

BGP defines the following message types, as described in this section:

- Open

- Keepalive
- Update
- Notification

#### Note

Keepalive messages have a length of 19 bytes. Other messages may be between 19 and 4096 bytes long.

#### Open and Keepalive Messages

After a TCP connection is established, the first message sent by each side is an open message. If the open message is acceptable, a keepalive message confirming the open message is sent back by the side that received the open message.

When the open is confirmed, the BGP connection is established, and update, keepalive, and notification messages can be exchanged.

BGP peers initially exchange their full BGP routing tables. From then on, incremental updates are sent as the routing table changes. Keepalive packets are sent to ensure that the connection is alive between the BGP peers, and notification packets are sent in response to errors or special conditions.

An open message includes the following information:

- **Version**— This 8-bit field indicates the message's BGP version number. The highest common version that both routers support is used. BGP implementations today use the current version, BGP-4.
- **My autonomous system**— This 16-bit field indicates the sender's autonomous system number. The peer router verifies this information; if it is not the autonomous system number expected, the BGP session is torn down.
- **Hold time**— This 16-bit field indicates the maximum number of seconds that can elapse between the successive keepalive or update messages from the sender. Upon receipt of an open message, the router calculates the value of the hold timer to use with this neighbor by using the smaller of its configured hold time (which has a default of 180 seconds) and the hold time received in the open message.
- **BGP router identifier (router ID)**— This 32-bit field indicates the sender's BGP identifier. The BGP router ID is an IP address assigned to that router and is determined at startup. The BGP router ID is chosen the same way the OSPF router ID is chosen: It is the highest active IP address on the router, unless a loopback interface with an IP address exists, in which case it is the highest such loopback IP address. Alternatively, the router ID can be statically configured, overriding the automatic selection.
- **Optional parameters**— A length field indicates the total length of the optional parameters field in octets. These parameters are Type, Length, and Value (TLV)-encoded. An example of an optional parameter is session authentication.

BGP does not use any transport protocol-based keepalive mechanism to determine whether peers can be reached. Instead, BGP keepalive messages are exchanged between peers often enough to keep the hold timer from expiring. If the negotiated hold time interval is 0, periodic keepalive messages are not sent. Keepalive messages consist of only a message header and have a length of 19 bytes; they are sent every 60 seconds by default.

#### Update Messages

An update message has information on one path only; multiple paths require multiple messages. All the attributes in the update message refer to that path, and the networks are those that can be reached through that path. An update message might include the following fields:

- **Withdrawn routes**— A list of IP address prefixes for routes that are being withdrawn from service, if any.
- **Path attributes**— The AS-path, origin, local preference, and so forth, as discussed in the next section. Each path attribute includes the attribute type, attribute length, and attribute value (TLV). The attribute type consists of the attribute flags, followed by the attribute type code.
- **Network layer reachability information (NLRI)**— A list of networks (IP address prefixes and their prefix lengths) that can be reached by this path.

#### Notification Messages

A BGP router sends a notification message when it detects an error condition. The BGP router closes the BGP connection immediately after sending the notification message. Notification messages include an error code, an error subcode, and data related to the error.

### BGP Neighbor States

BGP is a state machine that takes a router through the following states with its neighbors:

- Idle
- Connect
- Active
- Open sent
- Open confirm
- Established

Only when the connection is in the established state are update, keepalive, and notification messages exchanged.

Neighbor states are discussed in more detail in the “Understanding and Troubleshooting BGP Neighbor States” section, later in this chapter.

### BGP Attributes

BGP routers send BGP update messages about destination networks to other BGP routers. As described in the previous section, update messages can contain network layer reachability information, which is a list of one or more networks (IP address prefixes and their prefix lengths), and path attributes, which are a set of BGP metrics describing the path to these networks (routes). BGP uses the path attributes to determine the best path to the networks. The following are some terms defining how these attributes are implemented:

- An attribute is either well-known or optional, mandatory or discretionary, and transitive or nontransitive. An attribute might also be partial.
- Not all combinations of these characteristics are valid; path attributes fall into four separate categories:
  - Well-known mandatory
  - Well-known discretionary
  - Optional transitive
  - Optional nontransitive
- Only optional transitive attributes might be marked as partial.

These characteristics are described in the following sections.

### BGP Path Attribute Format

A BGP update message includes a variable-length sequence of path attributes describing the route. A path attribute is of variable length and consists of three fields:

- Attribute type, which consists of a 1-byte attribute flags field and a 1-byte attribute-type code field
- Attribute length
- Attribute value

The first bit of the attribute flags field indicates whether the attribute is optional or well known. The second bit indicates whether an optional attribute is transitive or nontransitive. The third bit indicates whether a transitive attribute is partial or complete. The fourth bit indicates whether the attribute length field is 1 or 2 bytes. The rest of the flag bits are unused and are set to 0.

### Well-Known Attributes

A well-known attribute is one that all BGP implementations must recognize and propagate to BGP neighbors.  
Note

If a well-known attribute is missing from an update message, a notification error is generated. This ensures that all BGP implementations agree on a standard set of attributes.

There are two types of well-known attributes:

- **Well-known mandatory attribute**— A well-known mandatory attribute *must* appear in all BGP update messages.



- **Well-known discretionary attribute**— A well-known discretionary attribute does not have to be present in all BGP update messages. (In other words, it is recognized by all BGP implementations but does not have to be in every update message.)

#### Optional Attributes

Attributes that are not well-known are called optional. BGP routers that implement an optional attribute might propagate it to other BGP neighbors, depending on its meaning. Optional attributes are either transitive or nontransitive, as follows:

- **Optional transitive**— BGP routers that do not implement an optional transitive attribute should pass it to other BGP routers untouched and mark the attribute as partial.
- **Optional nontransitive**— BGP routers that do not implement an optional nontransitive attribute must delete the attribute and must not pass it to other BGP routers.

#### Defined BGP Attributes

The attributes defined by BGP include the following:

- Well-known mandatory attributes
  - AS-path
  - Next hop
  - Origin
- Well-known discretionary attributes
  - Local preference
  - Atomic aggregate
- Optional transitive attributes
  - Aggregator
  - Community
- Optional nontransitive attribute
  - Multiexit-discriminator (MED)

In addition, Cisco has defined a weight attribute for BGP. The weight is configured locally on a router and is not propagated to any other BGP routers.

The AS-path, next-hop, origin, local preference, community, MED, and weight attributes are discussed more fully in the following sections. The atomic aggregate attribute informs the neighbor autonomous system that the originating router has aggregated (summarized) the routes. The aggregator attribute specifies the BGP router ID and autonomous system number of the router that performed the route aggregation. Both of these attributes are discussed in Appendix C, as is BGP community configuration.

#### Note

Appendix C describes how BGP route summarization is configured, using both the network router configuration command (which is detailed in the “Defining the Networks That BGP Advertises” section later in this chapter) and the aggregate-address ip-address mask [summary-only] [as-set] router configuration command.

#### Note

“The Route Selection Decision Process” section, later in this chapter, describes how the attributes are used to determine the best BGP path.

#### BGP Attribute Type Codes

Cisco uses the following attribute type codes:

- Origin—Type code 1
- AS-path—Type code 2
- Next-hop—Type code 3
- MED—Type Code 4
- Local-preference—type code 5
- Atomic-aggregate—type code 6



- Aggregator—Type code 7
- Community—Type code 8 (Cisco-defined)
- Originator-ID—Type code 9 (Cisco-defined)
- Cluster list—Type code 10 (Cisco-defined)

The originator ID and cluster list attributes are discussed in Appendix C.

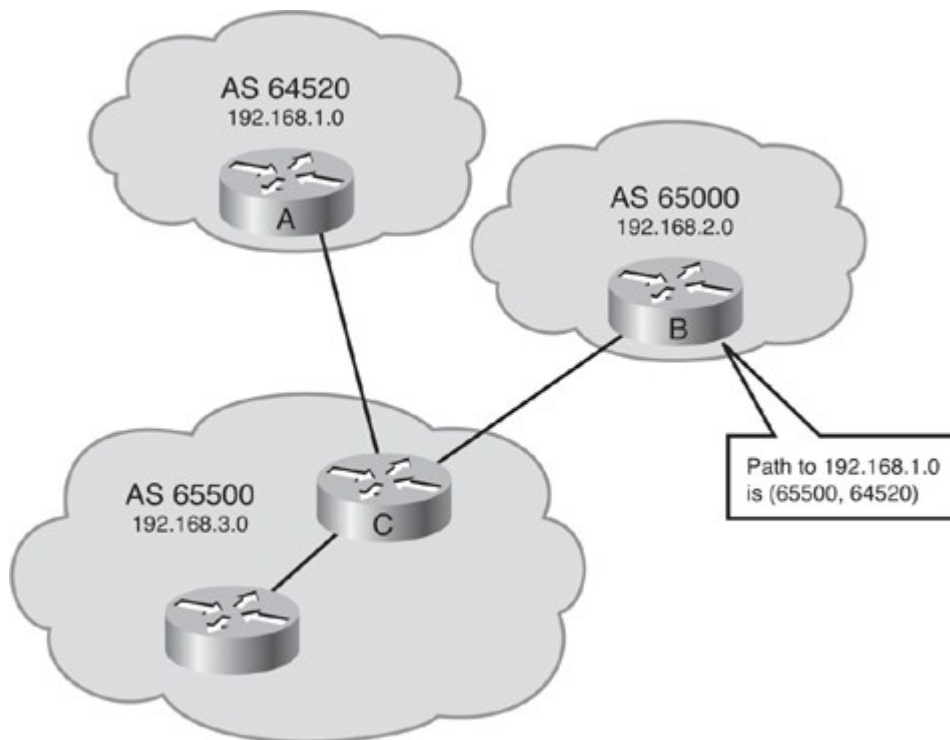
### The AS-Path Attribute

The AS-path attribute is the list of autonomous system numbers that a route has traversed to reach a destination, with the number of the autonomous system that originated the route at the end of the list.

The AS-path attribute is a well-known mandatory attribute. Whenever a route update passes through an autonomous system, the autonomous system number is *prepended* to that update (in other words, it is put at the beginning of the list) when it is advertised to the next EBGP neighbor.

In Figure 6-26, Router A in autonomous system 64520 advertises network 192.168.1.0. When that route traverses autonomous system 65500, Router C prepends its own autonomous system number to it. When the route to 192.168.1.0 reaches Router B, it has two autonomous system numbers attached to it. From Router B's perspective, the path to reach 192.168.1.0 is (65500, 64520).

Figure 6-26. Router C Prepends Its Own Autonomous System Number as It Passes Routes from Router A to Router B.



The same applies for 192.168.2.0 and 192.168.3.0. Router A's path to 192.168.2.0 is (65500 65000)—it traverses autonomous system 65500 and then autonomous system 65000. Router C has to traverse path (65000) to reach 192.168.2.0 and path (64520) to reach 192.168.1.0.

BGP routers use the AS-path attribute to ensure a loop-free environment. If a BGP router receives a route in which its own autonomous system is part of the AS-path attribute, it does not accept the route.

Autonomous system numbers are prepended only by routers advertising routes to EBGP neighbors. Routers advertising routes to IBGP neighbors do not change the AS-path attribute.

### The Next-Hop Attribute

The BGP next-hop attribute is a well-known mandatory attribute that indicates the next-hop IP address that is to be used to reach a destination.

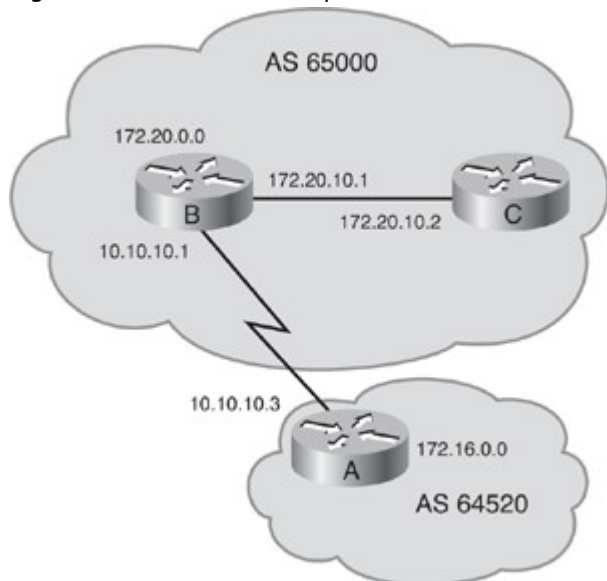
BGP, like IGPs, is a hop-by-hop routing protocol. However, unlike IGPs, BGP routes autonomous system by autonomous system, not router by router, and the default next-hop is the next autonomous system. The

next-hop address for a network from another autonomous system is an IP address of the entry point of the next autonomous system along the path to that destination network.

Therefore, for EBGp, the next-hop address is the IP address of the neighbor that sent the update.

This is illustrated in Figure 6-27. Router A advertises 172.16.0.0 to Router B, with a next hop of 10.10.10.3, and Router B advertises 172.20.0.0 to Router A, with a next hop of 10.10.10.1. Therefore, Router A uses 10.10.10.1 as the next-hop attribute to get to 172.20.0.0, and Router B uses 10.10.10.3 as the next-hop attribute to get to 172.16.0.0.

Figure 6-27. BGP Next-Hop Attribute.



For IBGP, however, the protocol states that the next hop advertised by EBGp should be carried into IBGP.

Because of this IBGP rule, Router B in Figure 6-27 advertises 172.16.0.0 to its IBGP peer Router C, with a next hop of 10.10.10.3 (Router A's address). Therefore, Router C knows that the next hop to reach 172.16.0.0 is 10.10.10.3, not 172.20.10.1, as you might expect.

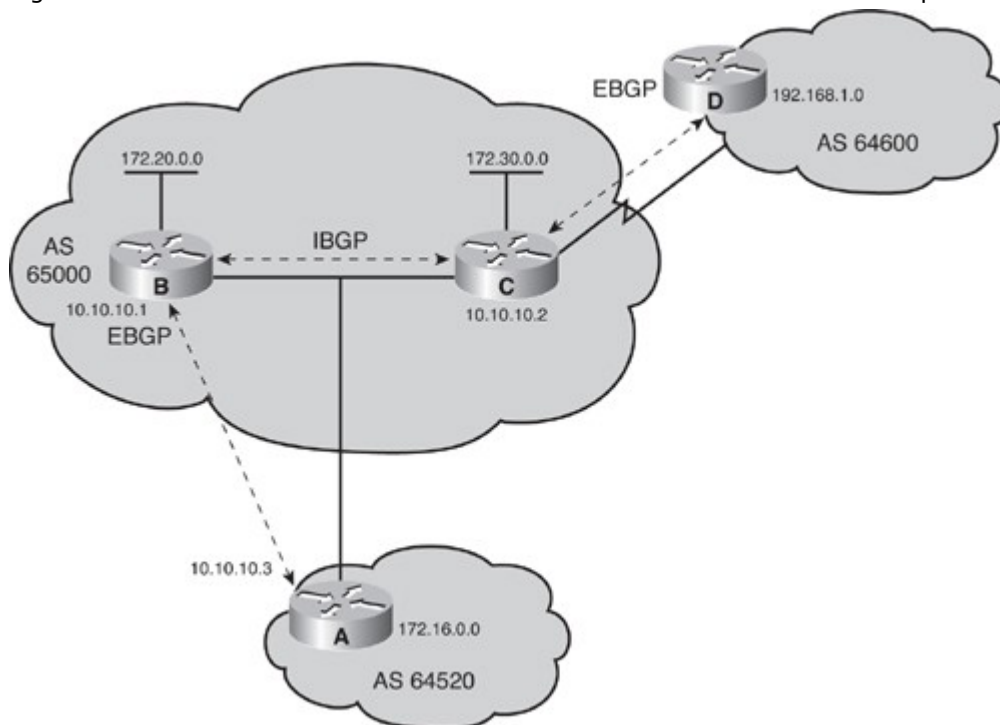
It is important, therefore, that Router C knows how to reach the 10.10.10.0 subnet, either via an IGP or a static route. Otherwise, it will drop packets destined for 172.16.0.0 because it will not be able to get to the next-hop address for that network.

The IBGP neighboring router performs a recursive lookup to find out how to reach the BGP next-hop address by using its IGP entries in the routing table. For example, Router C in Figure 6-27 learns in a BGP update about network 172.16.0.0/16 from the route source 172.20.10.1, Router B, with a next hop of 10.10.10.3, Router A. Router C installs the route to 172.16.0.0/16 in the routing table with a next hop of 10.10.10.3. Assuming that Router B announces network 10.10.10.0/24 using its IGP to Router C, Router C installs that route in its routing table with a next hop of 172.20.10.1. An IGP uses the source IP address of a routing update (route source) as the next-hop address, whereas BGP uses a separate field for each network to record the next-hop address. If Router C has a packet to send to 172.16.100.1, it looks up the network in the routing table and finds a BGP route with a next hop of 10.10.10.3. Because it is a BGP entry, Router C completes a recursive lookup in the routing table for a path to network 10.10.10.3; there is an IGP route to network 10.10.10.0 in the routing table with a next hop of 172.20.10.1. Router C then forwards the packet destined for 172.16.100.1 to 172.20.10.1.

When running BGP over a multiaccess network such as Ethernet, a BGP router uses the appropriate address as the next-hop address (by changing the next-hop attribute) to avoid inserting additional hops into the path. This feature is sometimes called a *third-party next hop*.

For example, in Figure 6-28, assume that Routers B and C in autonomous system 65000 are running an IGP, so that Router B can reach network 172.30.0.0 via 10.10.10.2. Router B is also running EBGp with Router A. When Router B sends a BGP update to Router A about 172.30.0.0, it uses 10.10.10.2 as the next hop, not its own IP address (10.10.10.1). This is because the network among the three routers is a multiaccess network, and it makes more sense for Router A to use Router C as a next hop to reach 172.30.0.0, rather than making an extra hop via Router B.

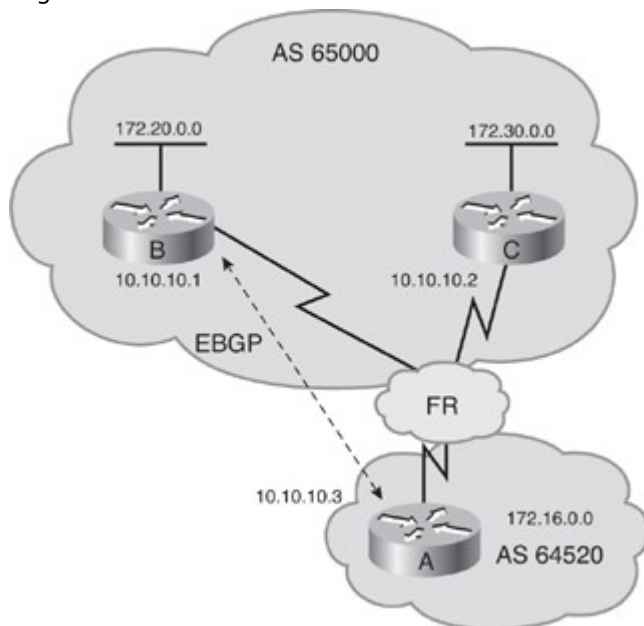
Figure 6-28. Multiaccess Network: Router A Has 10.10.10.2 as the Next-Hop Attribute to Reach 172.30.0.0.



The third-party next-hop address issue also makes sense when you review it from an ISP perspective. A large ISP at a public peering point has multiple routers peering with different neighboring routers; it is not possible for one router to peer with every neighboring router at the major public peering points. For example, in Figure 6-28, Router B might peer with autonomous system 64520, and Router C might peer with autonomous system 64600. However, each router must inform the other IBGP neighbor of reachable networks from other autonomous systems. From the perspective of Router A, it must transit autonomous system 65000 to get to networks in and behind autonomous system 64600. Router A has a neighbor relationship with only Router B in autonomous system 65000. However, Router B does not handle traffic going to autonomous system 64600. Router B gets to autonomous system 64600 through Router C, 10.10.10.2, and Router B must advertise the networks for autonomous system 64600 to Router A, 10.10.10.3. Router B notices that Routers A and C are on the same subnet, so Router B tells Router A to install the autonomous system 64600 networks with a next hop of 10.10.10.2, not 10.10.10.1. However, if the common medium between routers is a nonbroadcast multiaccess (NBMA) medium, complications might occur.

For example, in Figure 6-29, Routers A, B, and C are connected by Frame Relay. Router B can reach network 172.30.0.0 via 10.10.10.2. When Router B sends a BGP update to Router A about 172.30.0.0, it uses 10.10.10.2 as the next hop, not its own IP address (10.10.10.1). A problem arises if Routers A and C do not know how to communicate directly—in other words, if Routers A and C do not have a Frame Relay map entry to reach each other, Router A does not know how to reach the next-hop address on Router C.

Figure 6-29. NBMA Network: Router A Has 10.10.10.2 as the Next-Hop Attribute to Reach 172.30.0.0, but It Might Be Unreachable.



This behavior can be overridden in Router B by configuring it to advertise itself as the next-hop address for routes sent to Router A; this configuration is described later in the section “Changing the Next-Hop Attribute.”

#### Note

Routers can be configured to change the next-hop attribute in many scenarios, not only for NBMA networks, as described in the referenced section, later in this chapter.

#### The Origin Attribute

The origin is a well-known mandatory attribute that defines the origin of the path information. The origin attribute can be one of three values:

- **IGP**— The route is interior to the originating autonomous system. This normally happens when a **network** command is used to advertise the route via BGP. An origin of IGP is indicated with an *i* in the BGP table.
- **EGP**— The route is learned via EGP. This is indicated with an *e* in the BGP table. EGP is considered a historic routing protocol and is not supported on the Internet because it performs only classful routing and does not support CIDR.
- **Incomplete**— The route’s origin is unknown or is learned via some other means. This usually occurs when a route is redistributed into BGP. (Redistribution is discussed in Chapter 4, “Manipulating Routing Updates,” and in Appendix C.) An incomplete origin is indicated with a *?* in the BGP table.

#### The Local Preference Attribute

Local preference is a well-known discretionary attribute that indicates to routers in the autonomous system which path is preferred to exit the autonomous system.

A path with a *higher* local preference is preferred.

The term *local* refers to inside the autonomous system. The local preference attribute is sent only to IBGP neighbors; it is not passed to EBGP peers. Thus, local preference is an attribute that is configured on a router and exchanged only among routers within the same autonomous system. The default value for local preference on a Cisco router is 100.

#### Note

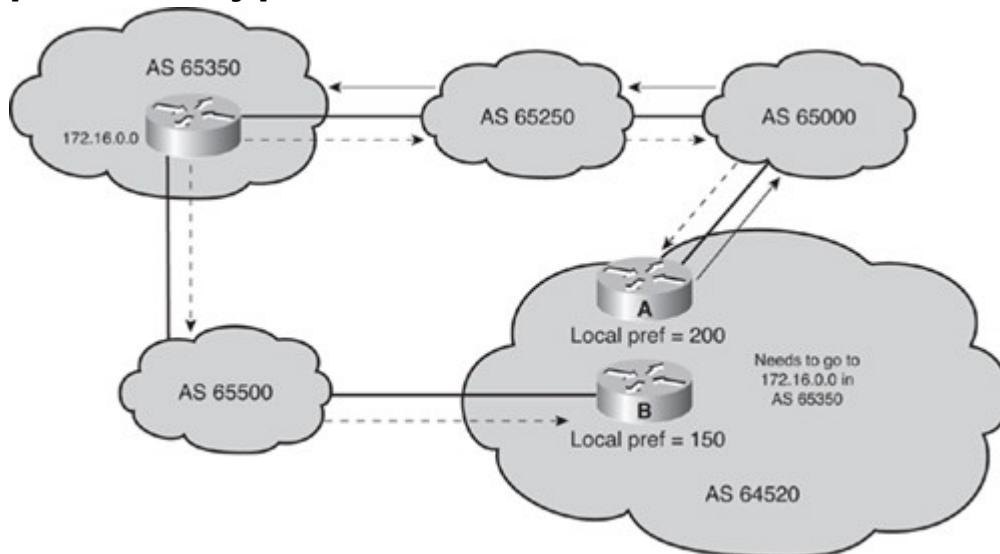
One way to reduce the IBGP mesh is to divide an autonomous system into multiple sub-autonomous systems and group them into a single *confederation*. To the outside world, the confederation looks like a single autonomous system. Each sub-autonomous system is fully meshed within itself, and has a few connections to other sub-autonomous systems in the same confederation. Even though the peers in different sub-autonomous systems have EBGP sessions with each other, they exchange routing information as if they were

IBGP peers. Specifically, the next-hop, MED, and local preference attributes are preserved. When configuring a BGP confederation, you specify a confederation identifier. To the outside world, the group of sub-autonomous systems will look like a single autonomous system, with the confederation identifier as the autonomous system number.

For example, in Figure 6-30, autonomous system 64520 receives updates about network 172.16.0.0 from two directions. Router A and Router B are IBGP neighbors. Assume that the local preference on Router A for network 172.16.0.0 is set to 200 and that the local preference on Router B for network 172.16.0.0 is set to 150. Because the local preference information is exchanged within autonomous system 64520, all traffic in autonomous system 64520 addressed to network 172.16.0.0 is sent to Router A as an exit point from autonomous system 64520.

Figure 6-30. Local Preference Attribute: Router A Is the Preferred Router to Get to 172.16.0.0.

[View full size image]



### The Community Attribute

BGP communities are one way to filter incoming or outgoing routes. BGP communities allow routers to *tag* routes with an indicator (the *community*) and allow other routers to make decisions based on that tag. Any BGP router can tag routes in incoming and outgoing routing updates, or when doing redistribution. Any BGP router can filter routes in incoming or outgoing updates or can select preferred routes based on communities (the tag).

BGP communities are used for destinations (routes) that share some common properties and, therefore, share common policies; routers act on the community rather than on individual routes. Communities are not restricted to one network or one autonomous system, and they have no physical boundaries.

Communities are optional transitive attributes. If a router does not understand the concept of communities, it defers to the next router. However, if the router does understand the concept, it must be configured to propagate the community; otherwise, communities are dropped by default.

#### Note

BGP community configuration is detailed in Appendix C.

### The MED Attribute

The MED attribute, also called the *metric*, is an optional nontransitive attribute.

#### Note

The MED was known as the inter-autonomous system attribute in BGP-3.

#### Note

The MED attribute is called the metric in the Cisco IOS. In the output of the **show ip bgp** command for example, the MED is displayed in the *metric* column.

The MED indicates to *external* neighbors the preferred path *into* an autonomous system. This is a dynamic way for an autonomous system to try to influence another autonomous system as to which way it should choose to reach a certain route if there are multiple entry points into the autonomous system.

A *lower* metric value is preferred.

Unlike local preference, the MED is exchanged between autonomous systems. The MED is sent to EBGp peers; those routers propagate the MED within their autonomous system, and the routers within the autonomous system use the MED, but do not pass it on to the next autonomous system. When the same update is passed on to another autonomous system, the metric will be set back to the default of 0.

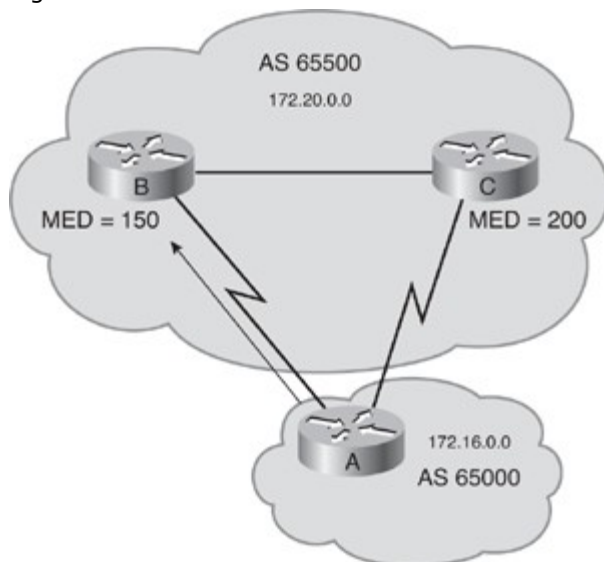
It is important to note the difference between MED and local preference: MED influences inbound traffic to an autonomous system, whereas local preference influences outbound traffic from an autonomous system.

By default, a router compares the MED attribute only for paths from neighbors in the same autonomous system.

By using the MED attribute, BGP is the only protocol that can affect the path used to send traffic into an autonomous system.

For example, in Figure 6-31, Router B has set the MED attribute to 150, and Router C has set the MED attribute to 200. When Router A receives update messages from Routers B and C (which include the path attributes), it picks Router B as the best next hop to get to autonomous system 65500, because the MED from Router B of 150 is less than the MED from Router C of 200.

Figure 6-31. MED Attribute: Router B Is the Best Next Hop to Get to Autonomous System 65500.



#### Note

By default, the MED comparison is done only if the neighboring autonomous system is the same for all routes considered. For the router to compare metrics from neighbors coming from different autonomous systems, the **bgp always-compare-med** router configuration command must be configured on the router.

#### The Weight Attribute (Cisco Only)

The weight attribute is a Cisco-defined attribute used for the path-selection process. The weight attribute is configured locally and provides local routing policy only; it is *not* propagated to *any* BGP neighbors.

Routes with a *higher* weight are preferred when multiple routes to the same destination exist.

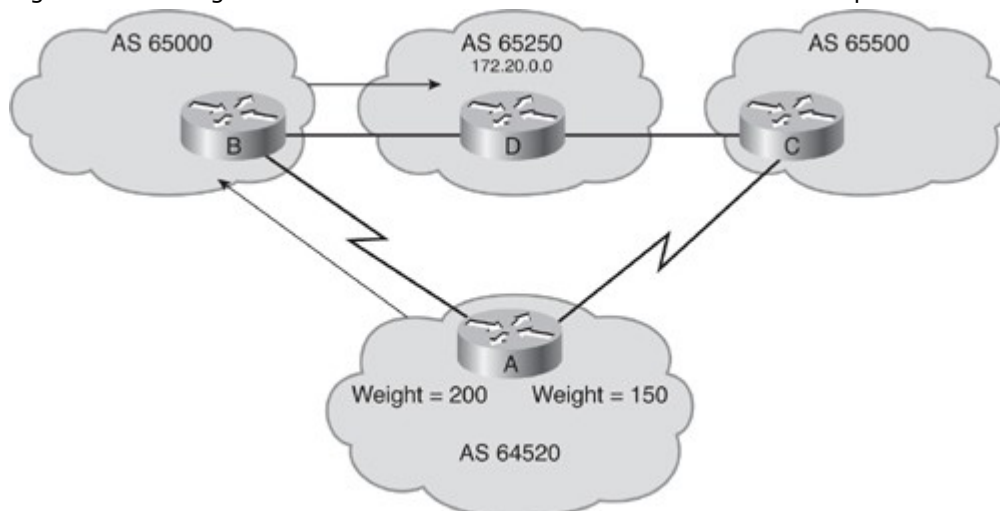
The weight can have a value from 0 to 65535. Paths that the router originates have a weight of 32768 by default, and other paths have a weight of 0 by default.

The weight attribute applies when using one router with multiple exit points out of an autonomous system, as compared to the local preference attribute, which is used when two or more routers provide multiple exit points.

In Figure 6-32, Routers B and C learn about network 172.20.0.0 from autonomous system 65250 and propagate the update to Router A. Router A has two ways to reach 172.20.0.0 and must decide which way to go. In the example, Router A is configured to set the weight of updates coming from Router B to 200 and the weight of those coming from Router C to 150. Because the weight for Router B's updates is higher than the weight for Router C's updates, Router A uses Router B as a next hop to reach 172.20.0.0.



Figure 6-32. Weight Attribute: Router A Uses Router B as the Next Hop to Reach 172.20.0.0.



#### The Route-Selection Decision Process

After BGP receives updates about different destinations from different autonomous systems, it decides which path to choose to reach each specific destination. Multiple paths might exist to reach a given network. These are kept in the BGP table. As paths for the network are evaluated, those determined not to be the best path are eliminated from the selection criteria but kept in the BGP table in case the best path becomes inaccessible.

BGP chooses only a single best path to reach a specific destination.

#### Note

The "Multiple Path Selection" sidebar later in this chapter describes the use of the maximum-paths command to allow multiple paths to be kept in the IP routing table; but BGP still selects one best path for the BGP table.

BGP is not designed to perform load balancing; paths are chosen because of policy, not based on bandwidth. The BGP selection process eliminates any multiple paths until a single best path is left.

The best path is submitted to the routing table manager process and is evaluated against any other routing protocols that can also reach that network. The route from the routing protocol with the lowest administrative distance is installed in the routing table.

#### BGP Route-Selection Process

The decision process is based on the attributes discussed earlier in the "BGP Attributes" section. When faced with multiple routes to the same destination, BGP chooses the best route for routing traffic toward the destination. A path is not considered if it is internal, synchronization is on, and the route is not synchronized (in other words, the route is not in the IGP routing table), or if the path's next-hop address cannot be reached. Therefore, to choose the best route, BGP considers only synchronized routes with no autonomous system loops and a valid next-hop address. The following process summarizes how BGP chooses the best route on a Cisco router:

- Step 1. Prefer the route with the highest weight. (Recall that the weight is Cisco proprietary and is local to the router only.)
- Step 2. If multiple routes have the same weight, prefer the route with the highest local preference. (Recall that the local preference is used within an autonomous system.)
- Step 3. If multiple routes have the same local preference, prefer the route that was originated by the local router. (A locally originated route has a next hop of 0.0.0.0 in the BGP table.)
- Step 4. If none of the routes were originated by the local router, prefer the route with the shortest AS-path.
- Step 5. If the AS-path length is the same, prefer the lowest-origin code (IGP < EGP < incomplete).
- Step 6. If all origin codes are the same, prefer the path with the lowest MED. (Recall that the MED is exchanged between autonomous systems.)



The MED comparison is done only if the neighboring autonomous system is the same for all routes considered, unless the **bgp always-compare-med** router configuration command is enabled.

#### Note

The most recent Internet Engineering Task Force (IETF) decision about BGP MED assigns a value of infinity to a missing MED, making a route lacking the MED variable the least preferred. The default behavior of BGP routers running Cisco IOS Software is to treat routes without the MED attribute as having a MED of 0, making a route lacking the MED variable the most preferred. To configure the router to conform to the IETF standard, use the **bgp bestpath med missing-as-worst** router configuration command.

- Step 7. If the routes have the same MED, prefer external paths (EBGP) over internal paths (IBGP).
- Step 8. If synchronization is disabled (the default) and only internal paths remain, prefer the path through the closest IGP neighbor. This means that the router prefers the shortest internal path within the autonomous system to reach the destination (the shortest path to the BGP next hop).
- Step 9. For EBGP paths, select the oldest route, to minimize the effect of routes going up and down (flapping).
- Step 10. Prefer the route with the lowest neighbor BGP router ID value.
- Step 11. If the BGP router IDs are the same, prefer the route with the lowest neighbor IP address.

Only the best path is entered in the routing table and propagated to the router's BGP neighbors.

#### Note

The route-selection decision process summarized here does not cover all cases, but it is sufficient for a basic understanding of how BGP selects routes.

For example, suppose that there are seven paths to reach network 10.0.0.0. All paths have no autonomous system loops and valid next-hop addresses, so all seven paths proceed to Step 1, which examines the weight of the paths. All seven paths have a weight of 0, so they all proceed to Step 2, which examines the paths' local preference. Four of the paths have a local preference of 200, and the other three have a local preference of 100, 100, and 150. The four with a local preference of 200 continue the evaluation process to the next step. The other three remain in the BGP forwarding table but are currently disqualified as the best path.

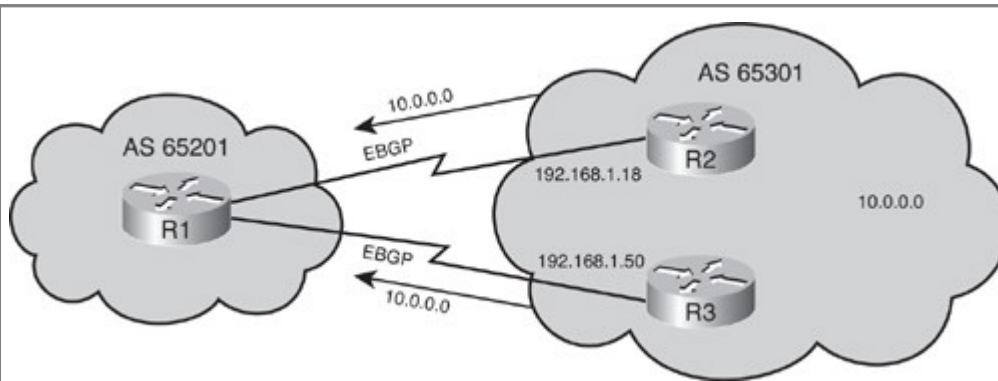
BGP continues the evaluation process until only a single best path remains. The single best path that remains is offered to the IP routing table as the best BGP path.

#### Multiple Path Selection

BGP chooses only a single best path for each destination.

The maximum-paths router configuration command for BGP works if your router has multiple parallel paths to different routers in the same remote autonomous system. However, this command affects only the number of routes kept in the IP routing table, not the number of paths selected as best by BGP. For BGP, the paths parameter defaults to one. For example, consider three routers: R1 is in autonomous system 65201, and both R2 and R3 are in autonomous system 65301, as shown in Figure 6-33. R1 is running EBGP to R2 and R3. R2 and R3 are advertising network 10.0.0.0. Without the maximum-paths command under the router bgp 65201 command on R1, there is only one path to 10.0.0.0 in R1's routing table. After the maximum-paths 2 command is added to the R1 BGP configuration, both paths appear in the IP routing table, as shown in Example 6-1. However, as also shown in Example 6-1, only one path is still selected as the best in the BGP table (as indicated by the > symbol); this is the path the router advertises to its BGP neighbors. This feature is referred to as BGP multipath.

Figure 6-33. BGP Maximum Paths Example.



Example 6-1. Output from Testing of the *maximum-paths* Command for BGP

Code View: Scroll / Show All

R1#**show ip route bgp**

```
B 10.0.0.0/8 [20/0] via 192.168.1.18, 00:00:41
 [20/0] via 192.168.1.50, 00:00:41
```

R1#**show ip bgp**

BGP table version is 3, local router ID is 192.168.1.49

Status codes: s suppressed, d damped, h history, \* valid, > best, i -> internal

Origin codes: i - IGP, e - EGP, ? - incomplete

| Network     | Next Hop     | Metric | LocPrf | Weight | Path    |
|-------------|--------------|--------|--------|--------|---------|
| *> 10.0.0.0 | 192.168.1.18 | 0      | 0      | 0      | 65301 i |
| * 10.0.0.0  | 192.168.1.50 | 0      | 0      | 0      | 65301 i |

#### The Path-Selection Decision Process with a Multihomed Connection

An autonomous system rarely implements BGP with only one EBGP connection, so generally multiple paths exist for each network in the BGP forwarding database.

##### Note

If you are running BGP in a network with only one EBGP connection, it is loop free. If synchronization is disabled or BGP is synchronized with the IGP for IBGP connections, and the next hop can be reached, the path is submitted to the IP routing table. Because there is only path, there is no benefit to manipulating its attributes.

Using the 11-step route-selection process, only the best path is put in the routing table and propagated to the router's BGP neighbors. Without route manipulation, the most common reason for path selection is Step 4, the preference for the shortest AS-path.

Step 1 looks at weight, which by default is set to 0 for routes that were not originated by this router.

Step 2 compares local preference, which by default is set to 100 for all networks. Both Step 1 and Step 2 have an effect only if the network administrator configures the weight or local preference to a nondefault value.

Step 3 looks at networks that are owned by this autonomous system. If one of the routes is injected into the BGP table by the local router, the local router prefers it to any routes received from other BGP routers.

Step 4 selects the path that has the fewest autonomous systems to cross. This is the most common reason a path is selected in BGP. If a network administrator does not like the path with the fewest autonomous

systems, he or she needs to manipulate weight or local preference to change which outbound path BGP chooses.

Step 5 looks at how a network was introduced into BGP. This introduction is usually either with **network** commands (*i* for an origin code) or through redistribution (? for an origin code).

Step 6 looks at MED to judge where the neighbor autonomous system wants this autonomous system to send packets for a given network. The Cisco IOS Software sets the MED to 0 by default. Therefore, MED does not participate in path selection unless the network administrator of the neighbor autonomous system manipulates the paths using MED.

If multiple paths have the same number of autonomous systems to traverse, the second most common decision point is Step 7, which states that an externally learned path from an EBGp neighbor is preferred over a path learned from an IBGP neighbor. A router in an autonomous system prefers to use the ISP's bandwidth to reach a network rather than using internal bandwidth to reach an IBGP neighbor on the other side of its own autonomous system.

If the autonomous system path length is equal and the router in an autonomous system has no EBGp neighbors for that network (only IBGP neighbors), it makes sense to take the quickest path to the nearest exit point. Step 8 looks for the closest IBGP neighbor; the IGP metric determines what closest means. (For example, RIP uses hop count, and OSPF uses the least cost, which by default is based on bandwidth in the Cisco IOS.)

If the autonomous system path length is equal and the costs via all IBGP neighbors are equal, or if all neighbors for this network are EBGp, the oldest path (Step 9) is the next common reason for selecting one path over another. EBGp neighbors rarely establish sessions at the exact same time. One session is likely older than another, so the paths through that older neighbor are considered more stable because they have been up longer.

If all these criteria are equal, the next most common decision is to take the neighbor with the lowest BGP router ID, which is Step 10.

If the BGP router IDs are the same (for example, if the paths are to the same BGP router), Step 11 states that the route with the lowest neighbor IP address is used.

## Configuring BGP

This section covers the commands used to configure some of the BGP features discussed in this chapter. After a discussion of planning a BGP implementation, the concept of peer groups is described first, because peer groups appear in many of the configuration commands.

### Note

The syntax of some BGP configuration commands is similar to the syntax of commands used to configure internal routing protocols. However, there are significant differences in how BGP functions.

## Planning BGP Implementations

Before BGP configuration, a network administrator must define the network requirements including the internal connectivity (for IBGP) and the external connectivity to the ISP (for EBGp).

The next step is to gather the parameters needed to provide the BGP configuration details. For basic BGP, these details include the following:

- The autonomous system numbers (of your own network and of all remote autonomous systems)
- The IP addresses of all the neighbors (peers) involved
- The networks that are to be advertised into BGP

Basic EBGp configuration requires the following main steps:

- Define the BGP process
- Establish the neighbor relationships
- Advertise the networks into BGP

## Peer Groups

In BGP, many neighbors are often configured with the same update policies (for example, they have the same filtering applied). On a Cisco IOS router, neighbors with the same update policies can be grouped into peer groups to simplify configuration and, more importantly, to make updating more efficient and improve performance. When a BGP router has many peers, this approach is highly recommended.

A BGP peer group is a group of BGP neighbors of the router being configured that all have the same update policies.

Instead of separately defining the same policies for each neighbor, a peer group can be defined with these policies assigned to the peer group. Individual neighbors are then made members of the peer group. The policies of the peer group are similar to a template. The template is then applied to the individual members of the peer group.

Members of the peer group inherit all the peer group's configuration options. The router can also be configured to override these options for some members of the peer group if these options do not affect outbound updates. In other words, only options that affect the inbound updates can be overridden.

#### Note

Some earlier IOS releases had a restriction that all EBGp neighbors in a peer group had to be reachable over the same interface. This is because the next-hop attribute would be different for EBGp neighbors accessible on different interfaces. You can get around this restriction by configuring a loopback source address for EBGp peers. This restriction was removed starting in Cisco IOS Software Releases 11.1(18)CC, 11.3(4), and 12.0.

Peer groups are more efficient than defining the same policies for each neighbor, because updates are generated only once per peer group rather than repetitiously for each neighboring router. The generated update is replicated for each neighbor that is part of the peer group.

Thus, peer groups save processing time in generating the updates for all IBGP neighbors and make the router configuration easier to read and manage.

The **neighbor peer-group-name peer-group** router configuration command is used to create a BGP peer group. The *peer-group-name* is the name of the BGP peer group to be created. The *peer-group-name* is local to the router on which it is configured; it is not passed to any other router.

Another syntax form of the neighbor peer-group command, the **neighbor ip-address peer-group peer-group-name** router configuration command, is used to assign neighbors as part of the group after the group has been created. Table 6-3 provides details of the parameters of this command. Using this command allows you to type the peer group name instead of typing the IP addresses of individual neighbors in other commands (for example, to link a policy to the group of neighboring routers). (Note that you must enter the neighbor peer-group-name peer-group command before the router will accept this second command.)

Table 6-3. *neighbor peer-group* Command Description

| Parameter       | Description                                                                         |
|-----------------|-------------------------------------------------------------------------------------|
| ip-address      | The IP address of the neighbor that is to be assigned as a member of the peer group |
| peer-group-name | The name of the BGP peer group                                                      |

A neighboring router can be part of only one peer group.

#### Note

Release 12.0(24)S of Cisco IOS Software introduced the BGP Dynamic Update Peer-Groups feature using peer templates to dynamically optimize update-groups of neighbors for shared outbound policies. You can find more information about this feature at <http://www.cisco.com>.

The **clear ip bgp peer-group peer-group-name EXEC** command is used to reset the BGP connections for all members of a BGP peer group. The *peer-group-name* is the name of the BGP peer group for which connections are to be cleared.

#### Caution

Resetting BGP sessions will disrupt routing. See the "Resetting BGP Sessions" section, later in this chapter, for more information about how the clear ip bgp commands operate.

#### Entering BGP Configuration Mode

Use the **router bgp autonomous-system** global configuration command to enter BGP configuration mode, and identify the local autonomous system in which this router belongs. In the command, *autonomous-system* identifies the local autonomous system. The BGP process needs to be informed of its autonomous system so that when BGP neighbors are configured it can determine whether they are IBGP or EBGp neighbors.

The **router bgp** command alone does not activate BGP on a router. You must enter at least one subcommand under the **router bgp** command to activate the BGP process on the router.

Only one instance of BGP can be configured on a router at a time. For example, if you configure your router in autonomous system 65000 and then try to configure the **router bgp 65100** command, the router informs you that you are currently configured for autonomous system 65000.

#### Defining BGP Neighbors and Activating BGP Sessions

Use the `neighbor {ip-address | peer-group-name} remote-as autonomous-system` router configuration command to activate a BGP session for external and internal neighbors and to identify a peer router with which the local router will establish a session, as described in Table 6-4.

Table 6-4. *neighbor remote-as* Command Description

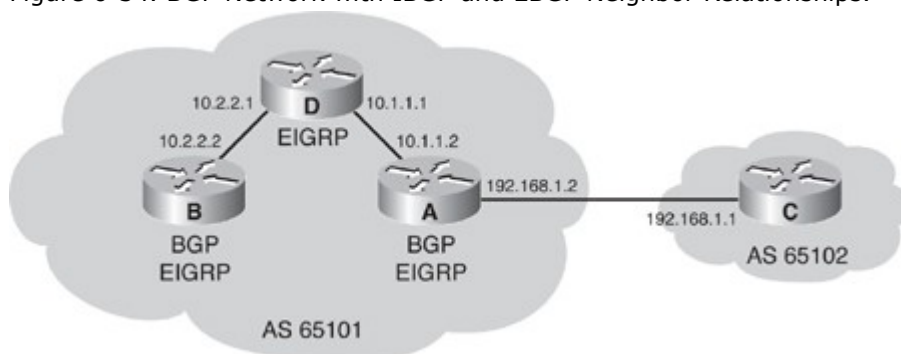
| Parameter         | Description                                    |
|-------------------|------------------------------------------------|
| ip-address        | Identifies the peer router                     |
| peer-group-name   | Identifies the name of a BGP peer group        |
| autonomous-system | Identifies the peer router's autonomous system |

The IP address used in the **neighbor remote-as** command is the destination address for all BGP packets going to this neighboring router. For a BGP relationship to be established, this address must be reachable, because BGP attempts to establish a TCP session and exchange BGP updates with the device at this IP address.

The value placed in the *autonomous-system* field of the **neighbor remote-as** command determines whether the communication with the neighbor is an EBGP or IBGP session. If the *autonomous-system* field configured in the **router bgp** command is identical to the field in the **neighbor remote-as** command, BGP initiates an internal session, and the IP address specified does not have to be directly connected. If the field values are different, BGP initiates an external session, and the IP address specified must be directly connected, by default.

The network shown in Figure 6-34 uses the BGP neighbor commands. The configurations of Routers A, B, and C are shown in Examples 6-2, 6-3, and 6-4. Router A in autonomous system 65101 has two neighbor statements. In the first statement, neighbor 10.2.2.2 (Router B) is in the same autonomous system as Router A (65101); this neighbor statement defines Router B as an IBGP neighbor. autonomous system 65101 runs EIGRP between all internal routers. Router A has an EIGRP path to reach IP address 10.2.2.2. As an IBGP neighbor, Router B can be multiple routers away from Router A.

Figure 6-34. BGP Network with IBGP and EBGP Neighbor Relationships.



Example 6-2. Configuration of Router A in Figure 6-34

```
router bgp 65101
 neighbor 10.2.2.2 remote-as 65101
 neighbor 192.168.1.1 remote-as 65102
```

Example 6-3. Configuration of Router B in Figure 6-34

```
router bgp 65101
 neighbor 10.1.1.2 remote-as 65101
```

Example 6-4. Configuration of Router C in Figure 6-34

```
router bgp 65102
 neighbor 192.168.1.2 remote-as 65101
```

Router A in Figure 6-34 knows that Router C is an external neighbor because the neighbor statement for Router C uses autonomous system 65102, which is different from the autonomous system number of Router A, autonomous system 65101. Router A can reach autonomous system 65102 via 192.168.1.1, which is directly connected to Router A.

#### Note

The network in Figure 6-34 is used just to illustrate the difference between configuring IBGP and EBGP sessions. As mentioned earlier, if Router B connects to another autonomous system then all routers in the transit path (Routers A, D, and B in this figure) should be running fully meshed BGP.

#### Shutting Down a BGP Neighbor

To disable (administratively shut down) an existing BGP neighbor or peer group, use the **neighbor {ip-address | peer-group-name} shutdown** router configuration command. To enable a previously existing neighbor or peer group that had been disabled using the **neighbor shutdown** command, use the **no neighbor {ip-address | peer-group-name} shutdown** router configuration command. If you want to implement major policy changes to a neighboring router and you change multiple parameters, you must administratively shut down the neighboring router, implement the changes, and then bring the neighboring router back up.

Using the **neighbor shutdown** command not only terminates the session, but also removes all associated routing information.

#### Defining the Source IP Address

The BGP **neighbor** statement tells the BGP process the destination IP address of each update packet. The router must decide which IP address to use as the source IP address in the BGP routing update.

When a router creates a packet, whether it is a routing update, a ping, or any other type of IP packet, the router does a lookup in the routing table for the destination address. The routing table lists the appropriate interface to get to the destination address. The address of this outbound interface is used as that packet's source address by default.

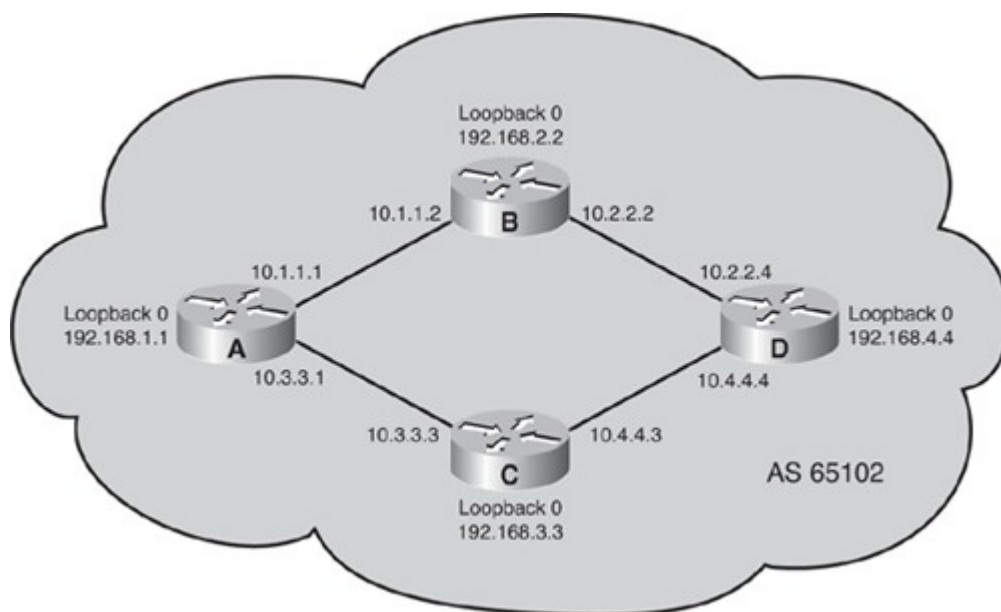
For BGP packets, this source IP address must match the address in the corresponding **neighbor** statement on the other router. (In other words, the other router must have a BGP relationship with the packet's source IP address.) Otherwise, the routers will not be able to establish the BGP session, and the packet will be ignored. BGP does not accept unsolicited updates; it must be aware of every neighboring router and have a **neighbor** statement for it.

For example, in Figure 6-35, assume that Router D uses the neighbor 10.3.3.1 remote-as 65102 command to establish a relationship with Router A. If Router A sends the BGP packets to Router D via Router B, the source IP address of the packets is 10.1.1.1. When Router D receives a BGP packet from 10.1.1.1, it does not recognize the sender of the BGP packet, because 10.1.1.1 is not configured as a neighbor of Router D. Therefore, the IBGP session between Router A and Router D is not established.

Figure 6-35. BGP Source Address Must Match the Address in the *neighbor* Command.

[View full size image]





A solution to this problem is to establish the IBGP session using a loopback interface when multiple paths exist between the IBGP neighbors.

If the IP address of a loopback interface is used in the **neighbor** command, some extra configuration must be done on the neighbor router. You must tell BGP to use a loopback interface address rather than a physical interface address as the source address for all BGP packets, including those that initiate the BGP neighbor TCP connection. Use the **neighbor {ip-address | peer-group-name} update-source loopback interface-number** router configuration command to cause the router to use the address of the specified loopback interface as the source address for BGP connections to this neighbor.

The **update-source** option in the **neighbor** command overrides the default source IP address selection for BGP packets. This peering arrangement also adds resiliency to the IBGP sessions because they are not tied into a physical interface, which might go down for any number of reasons. For example, if a BGP router is using a neighbor address that is assigned to a specific physical interface on another router, and that interface goes down, the router pointing to that address loses its BGP session with its BGP neighbor. If, instead, the router peers with the loopback interface of the other router, the BGP session is not lost, because the loopback interface is always available as long as the router itself does not fail.

To peer with the loopback interface of an IBGP neighbor, configure each router with a **neighbor** command using the neighbor's loopback address. Both routers must have a route to the loopback address of the other neighbor in their routing table; check to ensure that both routers are announcing their loopback addresses into the IGP. The **neighbor update-source** command is also necessary for both routers.

For example, in Figure 6-36, Router B has Router A as an EBGP neighbor and Router C as an IBGP neighbor. The configurations for Routers B and C are shown in Examples 6-5 and 6-6. The only reachable address that Router B can use to establish a BGP neighbor relationship with Router A is the directly connected address 172.16.1.1. However, Router B has multiple paths to reach Router C, an IBGP neighbor. All networks, including the IP network for Router C's loopback interface, can be reached from Router B because these two routers exchange EIGRP updates. (Router B and Router A do not exchange EIGRP updates.) The neighbor relationship between Routers B and C is not tied to a physical interface, because each router peers with the loopback interface on the other router and uses its loopback address as the BGP source IP address. If Router B instead peered with 10.1.1.2 on Router C and that interface went down, the BGP neighbor relationship would be lost.

Figure 6-36. BGP Sample Network Using Loopback Addresses.

[View full size image]





Example 6-5. Configuration of Router B in Figure 6-36

```
router bgp 65101
 neighbor 172.16.1.1 remote-as 65100
 neighbor 192.168.3.3 remote-as 65101
 neighbor 192.168.3.3 update-source loopback0
!

router eigrp 1
 network 10.0.0.0
 network 192.168.2.0
```

Example 6-6. Configuration of Router C in Figure 6-36

```
router bgp 65101
 neighbor 192.168.1.1 remote-as 65102
 neighbor 192.168.2.2 remote-as 65101
 neighbor 192.168.2.2 update-source loopback0
!

router eigrp 1
 network 10.0.0.0
 network 192.168.3.0
```

If Router B points at loopback address 192.168.3.3 on Router C and Router C points at loopback address 192.168.2.2 on Router B, but neither uses the **neighbor update-source** command, a BGP session is not established between these routers. Without this command, Router B will send a BGP open packet to Router C with a source IP address of either 10.1.1.1 or 10.2.2.1. Router C will examine the source IP address and attempt to match it against its list of known neighbors. Because Router C will not find a match, it will not respond to the open message from Router B.

#### EBGP Multihop

When peering with an external neighbor, the only address that an EBGP router can reach without further configuration is the interface that is directly connected to that EBGP router. Because IGP routing information is not exchanged with external peers, the router must by default point to a directly connected address for external neighbors. A loopback interface is never directly connected. Therefore, if you want to peer with a loopback interface instead, you have to add a static route to the loopback pointing to the physical address of the directly connected network (the next-hop address). You must also enable multihop EBGP, with the **neighbor {ip-address | peer-group-name} ebgp-multihop [ttl]** router configuration command.

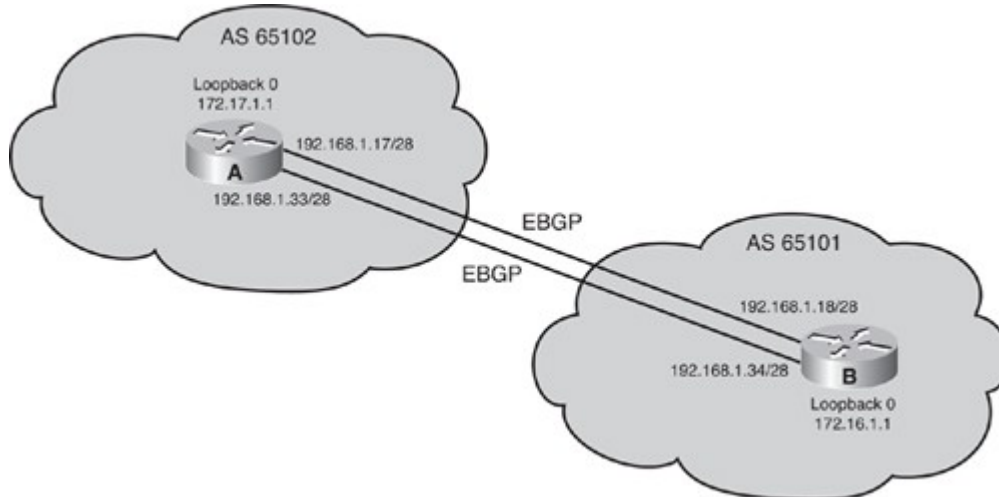
This command allows the router to accept and attempt BGP connections to external peers residing on networks that are not directly connected. This command increases the default of one hop for EBGP peers by changing the default Time To Live (TTL) value of 1 (with the *ttl* parameter) and therefore allowing routes to the EBGP loopback address. By default, the TTL is set to 255 with this command. This command is useful when redundant paths exist between EBGP neighbors.

For example, in Figure 6-37, Router A in autonomous system 65102 has two paths to Router B in autonomous system 65101. If Router A uses a `singleneighbor` statement that points to 192.168.1.18 on

Router B and that link goes down, the BGP session between these autonomous systems is lost, and no packets pass from one autonomous system to the next, even though another link exists. This problem can be solved if Router A uses two neighbor statements pointing to 192.168.1.18 and 192.168.1.34 on Router B. However, every BGP update that Router A receives will be sent to Router B twice because of the two neighbor statements.

Figure 6-37. EBGP Multihop Is Required If Loopback Is Used Between External Neighbors.

[View full size image]



The configurations of Routers A and B are shown in Examples 6-7 and 6-8. As these configurations show, each router instead points to the loopback address of the other router and uses its loopback address as the source IP address for its BGP updates. Because an IGP is not used between autonomous systems, neither router can reach the loopback of the other router without assistance. Each router needs to use two static routes to define the paths available to reach the loopback address of the other router. The neighbor ebgp-multihop command must also be configured to change the default setting of BGP and inform the BGP process that this neighbor IP address is more than one hop away. In these examples, the commands used on Routers A and B inform BGP that the neighbor address is two hops away (one hop to the other router and one hop through the router to the loopback interface).

Example 6-7. Configuration of Router A in Figure 6-37

```
router bgp 65102
 neighbor 172.16.1.1 remote-as 65101
 neighbor 172.16.1.1 update-source loopback0
 neighbor 172.16.1.1 ebgp-multihop 2
 ip route 172.16.1.1 255.255.255.255 192.168.1.18
 ip route 172.16.1.1 255.255.255.255 192.168.1.34
```

Example 6-8. Configuration of Router B in Figure 6-37

```
router bgp 65101
 neighbor 172.17.1.1 remote-as 65102
 neighbor 172.17.1.1 update-source loopback0
 neighbor 172.17.1.1 ebgp-multihop 2
 ip route 172.17.1.1 255.255.255.255 192.168.1.17
 ip route 172.17.1.1 255.255.255.255 192.168.1.33
```

#### Note

Recall that BGP is not designed to perform load balancing. Paths are chosen because of policy, not based on bandwidth. BGP will choose only a single best path. Using the loopback addresses and the **neighbor ebgp-multihop** command as shown in this example allows load balancing, and redundancy, across the two paths between the autonomous systems.

## Changing the Next-Hop Attribute

As discussed in the section “The Next-Hop Attribute,” earlier in this chapter, it is sometimes necessary (for example, in an NBMA environment) to override a router’s default behavior and force it to advertise itself as the next-hop address for routes sent to a neighbor.

An internal protocol, such as RIP, EIGRP, or OSPF, always uses the source IP address of a routing update as the next-hop address for each network from that update that is placed in the routing table. The neighbor {ip-address | peer-group-name} next-hop-self router configuration command is used to force BGP to use the source IP address of the update as the next hop for each network it advertises to the neighbor, rather than letting the protocol choose the next-hop address to use. This command is described in Table 6-5.

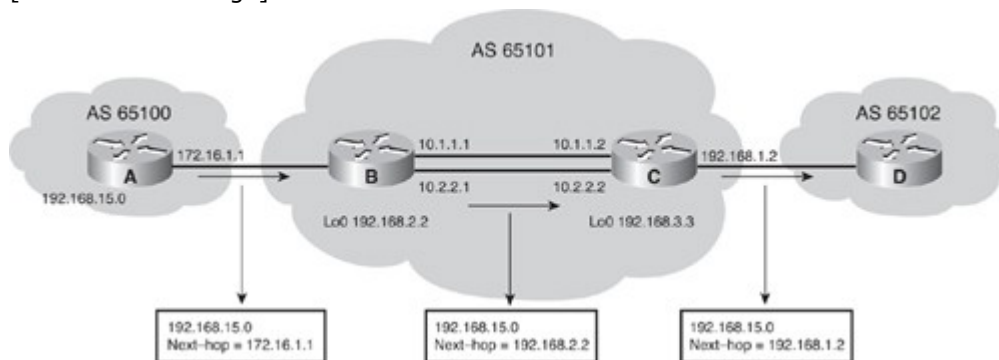
Table 6-5. *neighbor next-hop-self* Command Description

| Parameter       | Description                                                                                                          |
|-----------------|----------------------------------------------------------------------------------------------------------------------|
| ip-address      | Identifies the peer router to which advertisements will be sent, with this router identified as the next hop         |
| peer-group-name | Gives the name of a BGP peer group to which advertisements will be sent, with this router identified as the next hop |

For example, in Figure 6-38, Router B views all routes learned from autonomous system 65100 as having a next hop of 172.16.1.1, which is the entrance to autonomous system 65100 for Router B. When Router B announces those networks to its IBGP neighbors in autonomous system 65101, the BGP default setting is to announce that the next hop to reach each of those networks is the entrance to autonomous system 65100 (172.16.1.1), because BGP is an autonomous system-by-autonomous system routing protocol. With the default settings, a BGP router needs to reach the 172.16.1.1 next hop to reach networks in or behind autonomous system 65100. Therefore, the network that represents 172.16.1.1 will have to be advertised in the internal routing protocol.

Figure 6-38. *neighbor next-hop-self* Command Allows Router B to Advertise Itself as the Next Hop.

[View full size image]



In this example, however, the configuration for Router B is as shown in Example 6-9. Router B uses the neighbor next-hop-self command to change the default BGP next-hop settings. After this command is configured, Router B advertises a next hop of 192.168.2.2 (the IP address of its loopback interface) to its IBGP neighbor because that is the source IP address of the routing update to its IBGP neighbor (set with the neighbor update-source command).

Example 6-9. Configuration of Router B in Figure 6-38

```
router bgp 65101
 neighbor 172.16.1.1 remote-as 65100
 neighbor 192.168.3.3 remote-as 65101
 neighbor 192.168.3.3 update-source loopback0
 neighbor 192.168.3.3 next-hop-self
```

```
!

router eigrp 1
 network 10.0.0.0
 network 192.168.2.0
```

When Router C announces networks that are in or behind autonomous system 65101 to its EBGp neighbors, such as Router D in autonomous system 65102, Router C, by default, uses its outbound interface address 192.168.1.2 as the next-hop address. This address is the default next-hop address for Router D to use to reach any networks in or behind autonomous system 65101.

#### Defining the Networks That BGP Advertises

Two options are available to advertise networks into BGP: using the `network` command, and redistributing routes from an IGP into BGP. Redistribution is described in Chapter 4. The `network` command is described in the next section.

#### Note

Redistributing from an IGP into BGP is not recommended because any change in the IGP routes—for example, if a link goes down—might cause a BGP update. This method could result in unstable BGP tables. If redistribution is used, care must be taken that only local routes are redistributed. For example, routes learned from other autonomous systems (that were learned by redistributing BGP into the IGP) must not be sent out again from the IGP; otherwise, routing loops could result. Configuring this filtering can be complex. Using a **redistribute** command into BGP results in an incomplete origin attribute for the route, as indicated by the ? in the **show ip bgp** command output.

Use the `network network-number [mask network-mask] [route-map map-tag]` router configuration command to permit BGP to advertise a network if it is present in the IP routing table, as described in Table 6-6.

Table 6-6. *network* Command Description

| Parameter                       | Description                                                                                                                                                     |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>network-number</code>     | Identifies an IP network to be advertised by BGP.                                                                                                               |
| <b>mask</b> <i>network-mask</i> | (Optional) Identifies the subnet mask to be advertised by BGP. If the network mask is not specified, the default mask is the classful mask.                     |
| <b>route-map</b> <i>map-tag</i> | (Optional) Identifies a configured route map. The route map is examined to filter the networks to be advertised. If not specified, all networks are advertised. |

It is important to note that the BGP **network** command determines which networks this router advertises. This is a different concept from what you are used to when configuring IGPs. Unlike for IGPs, the **network** command does not start BGP on specific interfaces. Rather, it indicates to BGP which networks it should originate from this router. The **mask** parameter indicates that BGP-4 can handle subnetting and supernetting. The list of **network** commands must include all networks in your autonomous system that you want to advertise, not just those locally connected to your router.

The **network** command allows classless prefixes; the router can advertise individual subnets, networks, or supernets. It is also important to note that the prefix must *exactly* match (address and mask) an entry in the *IP routing table*. A static route to null 0 might be used to create a supernet entry in the IP routing table.

#### Note

Before Cisco IOS Software Release 12.0, there was a limit of 200 `network` commands per BGP router. This limit has now been removed. The router's resources, such as the configured NVRAM or RAM, determine the maximum number of `network` commands that you can now use.

Notice the difference between the **neighbor** command and the **network** command:

The **neighbor** command tells BGP where to advertise. The **network** command tells BGP what to advertise.

The sole purpose of the **network** command is to notify BGP which networks to advertise. If the mask is not specified, this command announces only the classful network number; at least one subnet of the specified major network must be present in the IP routing table to allow BGP to start announcing the classful network as a BGP route. However, if you specify the *network-mask*, an exact match to the network (both address and mask) must exist in the routing table for the network to be advertised.

Before BGP announces a route, it checks to see whether it can reach it. For example, if you want to advertise the 192.168.0.0/24 that is in your network, and by mistake you configure **network 192.168.0.0 mask 255.255.0.0**, instead of **network 192.168.0.0 mask 255.255.255.0**, BGP looks for 192.168.0.0/16 in the routing table. In this case it would find 192.168.0.0/24 but will not find 192.168.0.0/16. Because the routing table does not contain a specific match to the network, BGP does not announce the 192.168.0.0/24 network to any neighbors.

If you want to advertise the CIDR block 192.168.0.0/16, you might try configuring **network 192.168.0.0 mask 255.255.0.0**. Again, BGP looks for 192.168.0.0/16 in the routing table and if it never finds 192.168.0.0/16, BGP does not announce the 192.168.0.0/16 network to any neighbors. In this case, you can configure a static route to the CIDR block toward the null interface, with the **ip route 192.168.0.0 255.255.0.0 null0** command, so that BGP can find an exact match in the routing table. After finding an exact match in the routing table, BGP announces the 192.168.0.0/16 network to its neighbors.

To summarize the relationship between the BGP table, the IP routing table and the **network** command: The **network** command allows a BGP router to advertise a network that is in its IP routing table to its neighbors. The neighbor router that receives that network information puts the information in its BGP table and selects its best BGP route for that network. The best route is offered to its IP routing table. BGP neighbors exchange their best BGP routes. Routes learned by IBGP do not have to be in the IP routing table for BGP to use or advertise them (with the default of synchronization disabled).

Note

The BGP **auto-summary** router configuration command determines how BGP handles redistributed routes. With BGP summarization enabled (with **auto-summary**), all redistributed subnets are summarized to their classful boundaries in the BGP table. When disabled (with **no auto-summary**), all redistributed subnets are present in their original form in the BGP table, so only those subnets would be advertised.

In Cisco IOS Software Release 12.2(8)T, the default behavior of the **auto-summary** command was changed to disabled (**no auto-summary**); before that the default was enabled (**auto-summary**).

BGP route summarization is covered in Appendix C.

## BGP Neighbor Authentication

As described for EIGRP and OSPF earlier in this book, you can configure BGP neighbor authentication on a router so that the router authenticates the source of each routing update packet that it receives. This is accomplished by the exchange of an authenticating key (sometimes referred to as a password) that is known to both the sending and the receiving router. BGP supports message digest 5 (MD5) neighbor authentication. MD5 sends a "message digest" (also called a "hash"), which is created using the key and a message. The message digest is then sent instead of the key. The key itself is not sent, preventing it from being read by someone eavesdropping on the line while it is being transmitted.

To enable MD5 authentication on a TCP connection between two BGP peers, use the **neighbor {ip-address | peer-group-name} password string** router configuration command. Table 6-7 describes the neighbor password command parameters.

Table 6-7. *neighbor password* Command Description

| Parameter       | Description                                                                                                                                                                                                                                        |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ip-address      | IP address of the BGP-speaking neighbor.                                                                                                                                                                                                           |
| peer-group-name | Name of a BGP peer group.                                                                                                                                                                                                                          |
| string          | Case-sensitive password of up to 25 characters. The first character cannot be a number. The string can contain any alphanumeric characters, including spaces. You cannot specify a password in the format <i>number-space-anything</i> . The space |

| Parameter | Description                                        |
|-----------|----------------------------------------------------|
|           | after the number can cause authentication to fail. |

#### Note

If the **service password-encryption** command is not used when configuring BGP authentication, the password will be stored as plain text in the router configuration. If you configure the **service password-encryption** command, the password will be stored and displayed in an encrypted form. *When it is displayed*, there will be an *encryption type of 7* specified before the encrypted password.

When MD5 authentication is configured between two BGP peers, each segment sent on the TCP connection between the peers is verified. MD5 authentication must be configured with the same password on both BGP peers. Otherwise, the connection between them will not be made. Configuring MD5 authentication causes the Cisco IOS Software to generate and check the MD5 digest of every segment sent on the TCP connection.

#### Caution

If the authentication string is configured incorrectly, the BGP peering session will not be established. You should enter the authentication string carefully and verify that the peering session is established after authentication has been configured.

If a router has a password configured for a neighbor, but the neighbor router does not have a password configured, a message such as the following will appear on the console when the routers attempt to send BGP messages between themselves:

```
%TCP-6-BADAUTH: No MD5 digest from 10.1.0.2(179) to 10.1.0.1(20236)
```

Similarly, if the two routers have different passwords configured, a message such as the following will appear on the screen:

```
%TCP-6-BADAUTH: Invalid MD5 digest from 10.1.0.1(12293) to 10.1.0.2(179)
```

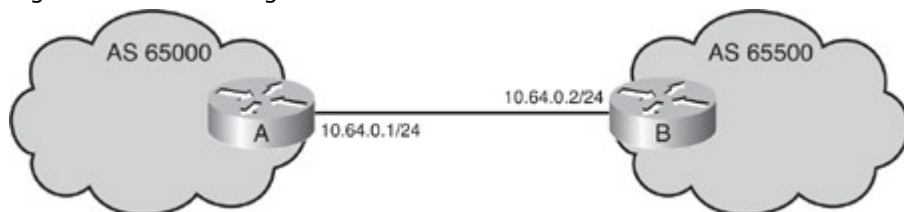
If you configure or change the password or key used for MD5 authentication between two BGP peers, the local router will not tear down the existing session after you configure the password. The local router will attempt to maintain the peering session using the new password until the BGP hold-down timer (with a default of 180 seconds) expires. If the password is not entered or changed on the remote router before the hold-down timer expires, the session will time out.

#### Note

Configuring a new timer value for the hold-down timer will take effect only after the session has been reset. So, it is not possible to change the configuration of the hold-down timer to avoid resetting the BGP session.

Examples 6-10 and 6-11 show the configurations for the routers in Figure 6-39. MD5 authentication is configured for the BGP peering session between Routers A and B. The same password must be configured on the remote peer before the hold-down timer expires.

Figure 6-39. BGP Neighbor Authentication.



Example 6-10. Configuration of Router A in Figure 6-39

```
router bgp 65000
 neighbor 10.64.0.2 remote-as 65500
 neighbor 10.64.0.2 password v61ne0qke1336
```



#### Example 6-11. Configuration of Router B in Figure 6-39

```
router bgp 65500
 neighbor 10.64.0.1 remote-as 65000
 neighbor 10.64.0.1 password v61ne0qke1336
```

#### Configuring BGP Synchronization

Recall that the BGP synchronization rule states that a BGP router should not use, or advertise to an external neighbor, a route learned by IBGP, unless that route is local or is learned from the IGP.

BGP synchronization is disabled by default in Cisco IOS Software Release 12.2(8)T and later; it was on by default in earlier Cisco IOS Software releases. With the default of synchronization disabled, BGP can use and advertise to an external BGP neighbor routes learned from an IBGP neighbor that are not present in the local routing table.

Use the **synchronization** router configuration command to enable BGP synchronization so that a router will not advertise routes in BGP until it learns them in an IGP. The **no synchronization** router configuration command disables synchronization if it was enabled.

#### Resetting BGP Sessions

BGP can potentially handle huge volumes of routing information. When a BGP policy configuration change occurs (such as when access lists, timers, or attributes are changed), the router cannot go through the huge table of BGP information and recalculate which entry is no longer valid in the local table. Nor can the router determine which route or routes, already advertised, should be withdrawn from a neighbor. There is an obvious risk that the first configuration change will immediately be followed by a second, which would cause the whole process to start all over again. To avoid such a problem, the Cisco IOS Software applies changes on only those updates received or sent *after* the BGP policy configuration change has been performed. The new policy, enforced by the new filters, is applied only on routes received or sent after the change.

If the network administrator wants the policy change to be applied on all routes, he or she must trigger an update to force the router to let all routes pass through the new filter. If the filter is applied to outgoing information, the router has to resend the BGP table through the new filter. If the filter is applied to incoming information, the router needs its neighbor to resend its BGP table so that it passes through the new filter.

There are three ways to trigger an update:

- Hard reset
- Soft reset
- Route refresh

The sections that follow detail all three methods of triggering an update.

#### Hard Reset of BGP Sessions

Resetting a session is a method of informing the neighbor or neighbors of a policy change. If BGP sessions are reset, all information received on those sessions is invalidated and removed from the BGP table. The remote neighbor detects a BGP session down state and, likewise, invalidates the received routes. After a period of 30 to 60 seconds, the BGP sessions are reestablished automatically, and the BGP table is exchanged again, but through the new filters. However, resetting the BGP session disrupts packet forwarding.

Use the **clear ip bgp \*** or **clear ip bgp {neighbor-address}** privileged EXEC command to cause a hard reset of the BGP neighbors involved, where *\** indicates all sessions and the *neighbor-address* identifies the address of a specific neighbor for which the BGP sessions will be reset. A "hard reset" means that the router issuing either of these commands will close the appropriate TCP connections, reestablish those TCP sessions as appropriate, and resend all information to each of the neighbors affected by the particular command that is used.

#### Caution

Clearing the BGP table and resetting BGP sessions will disrupt routing, so do not use these commands unless you have to.

The **clear ip bgp \*** command causes the BGP forwarding table on the router on which this command is issued to be completely deleted; all networks must be relearned from every neighbor. If a router has multiple neighbors, this action is a very dramatic event. This command forces all neighbors to resend their entire tables simultaneously.

For example, assume that Router A has eight neighbors and that each neighbor sends Router A the full Internet table (assume that is about 32 MB in size). If the **clear ip bgp \*** command is issued on Router A, all eight routers resend their 32 MB table at the same time. To hold all of these updates, Router A will need a lot



of RAM (possibly 256 MB) and will also need to be able to process all of this information. Processing this many updates will take a considerable number of CPU cycles for Router A, further delaying the routing of user data.

If, instead, the **clear ip bgp neighbor-address** command is used, one neighbor is reset at a time. The impact is less severe on the router issuing this command. However, it takes longer to change policy to all the neighbors, because each must be done individually rather than all at once as it is with the **clear ip bgp \*** command. The **clear ip bgp neighbor-address** command still performs a hard reset and must reestablish the TCP session with the specified address used in the command, but this command affects only a single neighbor at a time, not all neighbors simultaneously.

#### Soft Reset of BGP Sessions Outbound

Use the **clear ip bgp { \* | neighbor-address } [soft out]** privileged EXEC command to cause BGP to do a soft reset for outbound updates. The router on which this command is issued does not reset the BGP session. Instead, the router creates a new update and sends the whole table to the specified neighbors. This update includes withdrawal commands for networks that the neighbor will not see anymore based on the new outbound policy.

#### Note

The **soft** keyword of this command is optional; **clear ip bgp out** does a soft reset for outbound updates.

Outbound BGP soft configuration does not have any memory overhead. This command is highly recommended when you are changing an outbound policy, but does not help if you are changing an inbound policy.

#### Soft Reset of BGP Sessions Inbound

There are two ways to perform an inbound soft reconfiguration: using stored routing update information and dynamically.

##### Inbound Soft Reset Using Stored Information

To use stored information, first enter the **neighbor {ip-address} soft-reconfiguration inbound** router configuration command to cause BGP to save all updates that were learned from the neighbor specified. The BGP router retains an unfiltered table of what that neighbor has sent. When the inbound policy is changed, use the **clear ip bgp { \* | neighbor-address } soft in** privileged EXEC command to cause the router to use the stored unfiltered table to generate new inbound updates (in other words, updates from neighbors). The new results are placed in the BGP forwarding database. Therefore, if you make changes, you do not have to force the other side to resend everything.

##### Route Refresh: Dynamic Inbound Soft Reset

Cisco IOS Software Releases 12.0(2)S and 12.0(6)T introduced a BGP soft reset enhancement feature, also known as route refresh, that provides automatic support for dynamic soft reset of inbound BGP routing table updates that is not dependent on stored routing table update information. This new method requires no preconfiguration (the **neighbor soft-reconfiguration inbound** command is not required) and requires significantly less memory than the previous soft reset method for inbound routing table updates. The **clear ip bgp { \* | neighbor-address } [soft in | in]** privileged EXEC command is the only command required for this dynamic soft reconfiguration.

The **soft in** option generates new inbound updates without resetting the BGP session, but it can be memory intensive. BGP does not allow a router to force another BGP speaker to resend its entire table. If you change the inbound BGP policy and you do not want to complete a hard reset, use this command to cause the router to perform a soft reconfiguration.

#### Note

To determine whether a BGP router supports this route refresh capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the router supports the route refresh capability:

Received route refresh capability from peer.

If all BGP routers support the route refresh capability, use the **clear ip bgp { \* | address | peer-group-name } in** command. You need not use the **soft** keyword, because soft reset is automatically assumed when the route refresh capability is supported.

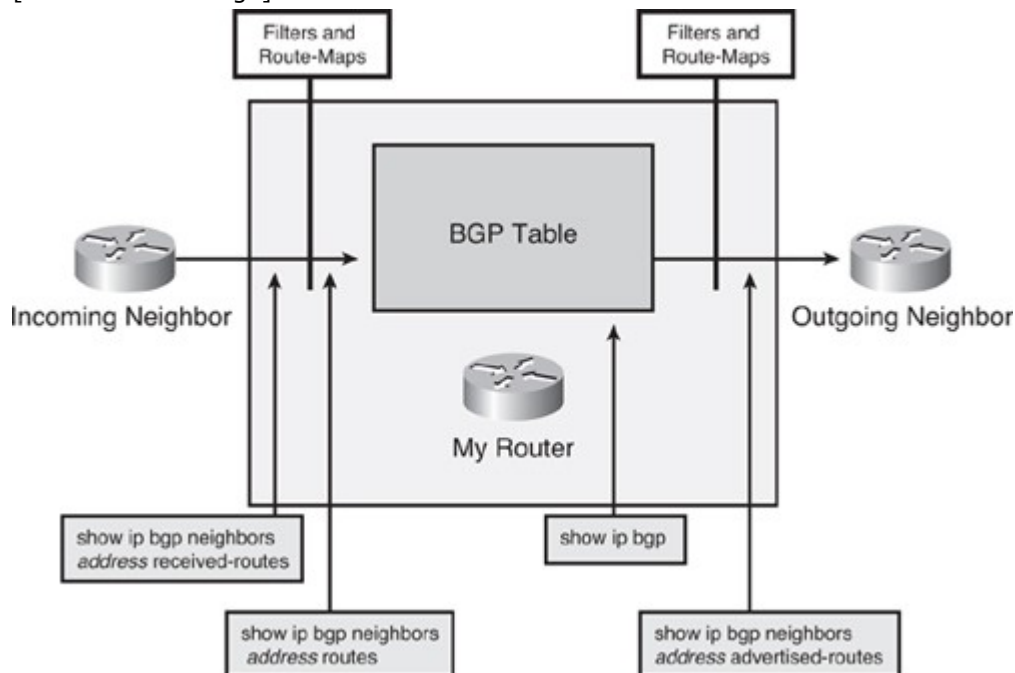
#### Note

The **clear ip bgp soft** command performs a soft reconfiguration of both inbound and outbound updates.

When a BGP session is reset using soft reconfiguration, the following commands can be useful for monitoring the BGP routes received, sent, or filtered, as illustrated in Figure 6-40:

Figure 6-40. Monitoring Soft Reconfiguration.

[View full size image]



- **show ip bgp neighbors {address} received-routes**— Displays all received routes (both accepted and rejected) from the specified neighbor.
- **show ip bgp neighbors {address} routes**— Displays all routes that are received and accepted from the specified neighbor. This output is a subset of the output displayed by the **received-routes** keyword.
- **show ip bgp**— Displays entries in the BGP table.
- **show ip bgp neighbors {address} advertised-routes**— Displays all BGP routes that have been advertised to neighbors

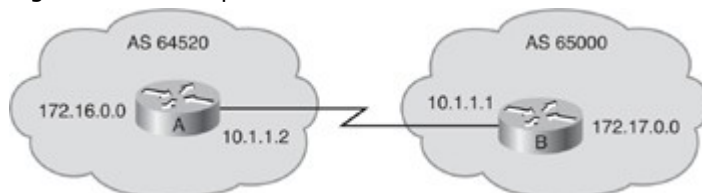
#### BGP Configuration Examples

This section provides some configuration examples using the commands discussed.

##### Basic BGP Examples

Figure 6-41 shows a sample BGP network. Example 6-12 provides the configuration of Router A, and Example 6-13 provides the configuration of Router B.

Figure 6-41. Sample BGP Network.



Example 6-12. Configuration of Router A in Figure 6-41

```
router bgp 64520
 neighbor 10.1.1.1 remote-as 65000
 network 172.16.0.0
```

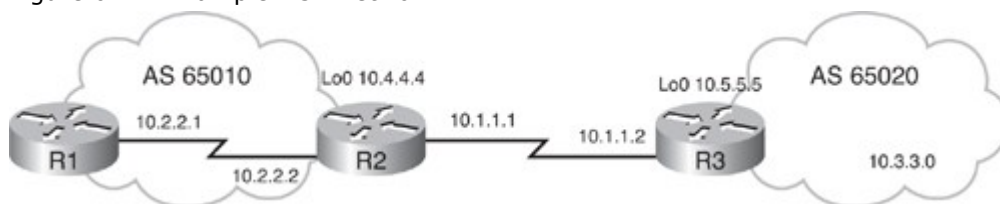
Example 6-13. Configuration of Router B in Figure 6-41

```
router bgp 65000
 neighbor 10.1.1.2 remote-as 64520
 network 172.17.0.0
```

In this example, Routers A and B define each other as BGP neighbors and start an EBGP session. Router A advertises the network 172.16.0.0/16, and Router B advertises the network 172.17.0.0/16.

Figure 6-42 illustrates another network. The configuration of router R2 is provided in Example 6-14. Router R2 is in autonomous system 65010. The autonomous system used in the neighbor remote-as command is different from the local autonomous system, so the neighbor is an EBGP neighbor and router R2 establishes a TCP session and starts exchanging BGP updates with router R3. Router R2 is configured to advertise two networks to its neighbor. The exact network entries 10.2.2.0/24 and 10.4.4.0/24 must exist in the routing table to be advertised by BGP.

Figure 6-42. Example BGP Network.



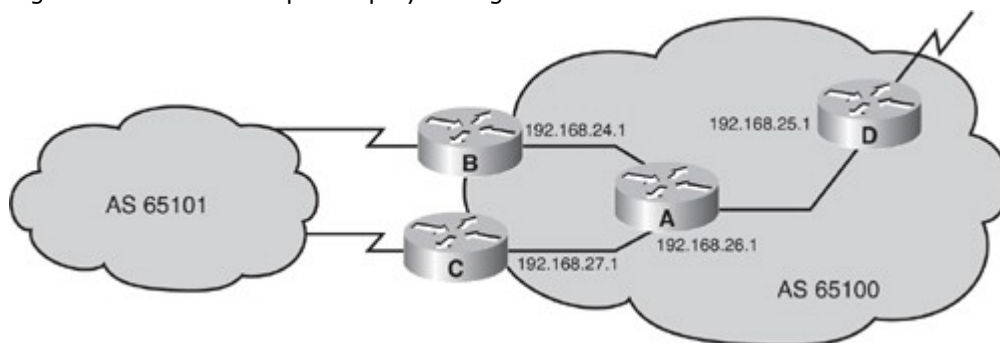
Example 6-14. Configuration of Router R2 in Figure 6-42

```
router bgp 65010
 neighbor 10.1.1.2 remote-as 65020
 network 10.2.2.0 mask 255.255.255.0
 network 10.4.4.0 mask 255.255.255.0
```

#### Peer Group Example

In Figure 6-43, autonomous system 65100 has four routers running IBGP. All of these IBGP neighbors are peering with each others' loopback 0 interface (the addresses of which are shown in the figure) and are using the IP address of their loopback 0 interface as the source IP address for all BGP packets. Each router is using one of its own IP addresses as the next-hop address for each network advertised through BGP. These are outbound policies.

Figure 6-43. Peer Groups Simplify Configuration.



Example 6-15 shows the BGP configuration of Router C when it is not using a peer group.

Example 6-15. Configuration of Router C in Figure 6-43 Without Using a Peer Group

```
router bgp 65100
 neighbor 192.168.24.1 remote-as 65100
 neighbor 192.168.24.1 update-source loopback 0
 neighbor 192.168.24.1 next-hop-self
```

```
neighbor 192.168.24.1 distribute-list 20 out
neighbor 192.168.25.1 remote-as 65100
neighbor 192.168.25.1 update-source loopback 0
neighbor 192.168.25.1 next-hop-self
neighbor 192.168.25.1 distribute-list 20 out
neighbor 192.168.26.1 remote-as 65100
neighbor 192.168.26.1 update-source loopback 0
neighbor 192.168.26.1 next-hop-self
neighbor 192.168.26.1 distribute-list 20 out
```

Router C has an outbound distribution list associated with each IBGP neighbor. This outbound filter performs the same function as the **distribute-list** command you use for internal routing protocols; however, when used for BGP, it is linked to a specific neighbor. For example, the ISP behind Router C might be announcing private address space to Router C, and Router C does not want to pass these networks to other routers running BGP in autonomous system 65100.

To accomplish this, **access-list 20**, referenced in the **distribute-list** command, might look like the following:

- access-list 20 deny 10.0.0.0 0.255.255.255
- access-list 20 deny 172.16.0.0 0.31.255.255
- access-list 20 deny 192.168.0.0 0.0.255.255
- access-list 20 permit any

As shown in Example 6-15, all IBGP neighbors have the outbound distribution list linked to them individually. If Router C receives a change from autonomous system 65101, it must generate an individual update for each IBGP neighbor and run each update against distribute list 20. If Router C has a large number of IBGP neighbors, the processing power needed to inform the IBGP neighbors of the changes in autonomous system 65101 could be extensive.

Example 6-16 shows the configuration of Router C when it is using a peer group called internal. The neighbor remote-as, neighbor update-source, neighbor next-hop-self, and neighbor distribute-list 20 out commands are all linked to peer group internal, which in turn is linked to each of the IBGP neighbors. If Router C receives a change from autonomous system 65101, it creates a single update and processes it through distribute list 20 once. The update is replicated for each neighbor that is part of the internal peer group. This action saves processing time in generating the updates for all IBGP neighbors. Therefore, the use of peer groups can improve efficiency when processing updates for BGP neighbors that have a common outbound BGP policy.

Example 6-16. Configuration of Router C in Figure 6-43 Using a Peer Group

```
router bgp 65100
neighbor internal peer-group
neighbor internal remote-as 65100
neighbor internal update-source loopback 0
neighbor internal next-hop-self
neighbor internal distribute-list 20 out
neighbor 192.168.24.1 peer-group internal
neighbor 192.168.25.1 peer-group internal
neighbor 192.168.26.1 peer-group internal
```

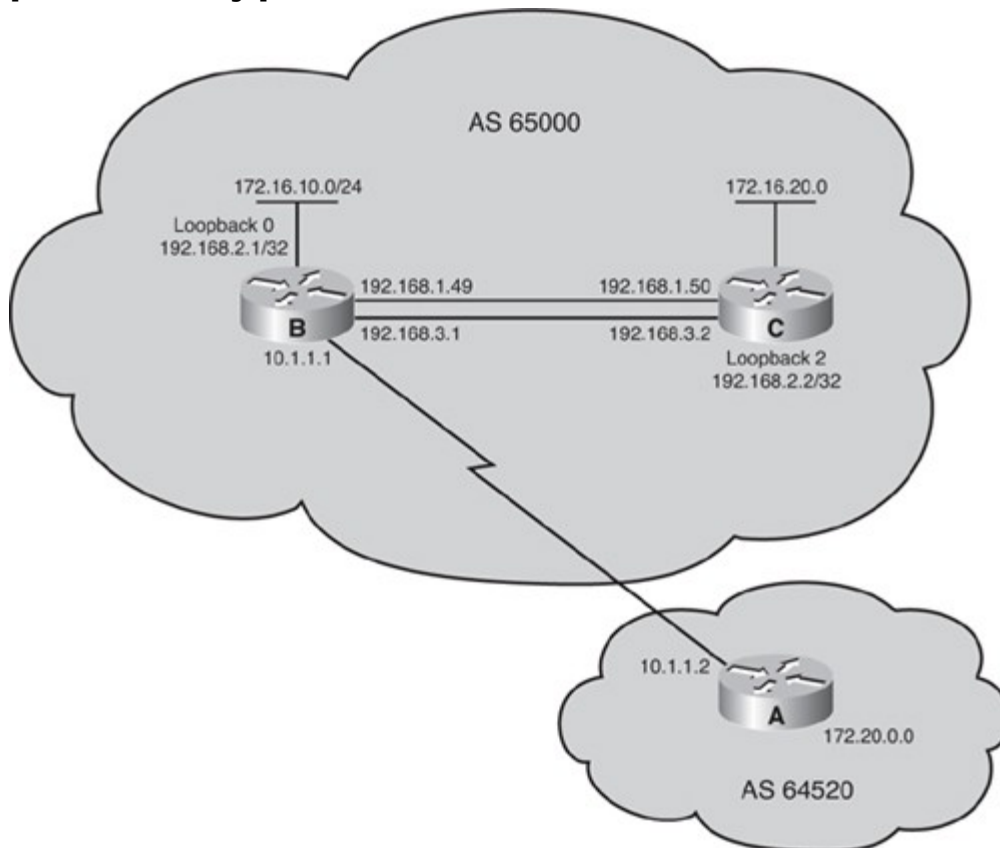
Adding a new neighbor with the same policies as the other IBGP neighbors to Router C when it is using a peer group requires adding only a single **neighbor** statement to link the new neighbor to the peer group. Adding that same neighbor to Router C if it does not use a peer group requires four **neighbor** statements.

Using a peer group also makes the configuration easier to read and change. If you need to add a new policy, such as a route map, to all IBGP neighbors on Router C, and you are using a peer group, you need only to link the route map to the peer group. If Router C does not use a peer group, you need to add the new policy to each neighbor.

IBGP and EBGp Examples

Figure 6-44 shows another BGP example.

Figure 6-44. IBGP and EBGp Example.  
 [View full size image]



Example 6-17 shows the configuration for Router B in Figure 6-44; note that line numbers have been added to this example to simplify the discussion. The first two commands under the router bgp 65000 command establish that Router B has the following two BGP neighbors:

- Router A in autonomous system 64520
- Router C in autonomous system 65000

Example 6-17. Configuration of Router B in Figure 6-44

```

1. router bgp 65000
2. neighbor 10.1.1.2 remote-as 64520
3. neighbor 192.168.2.2 remote-as 65000
4. neighbor 192.168.2.2 update-source loopback 0
5. neighbor 192.168.2.2 next-hop-self
6. network 172.16.10.0 mask 255.255.255.0
7. network 192.168.1.0
8. network 192.168.3.0
9. no synchronization

```

From the perspective of Router B, Router A is an EBGp neighbor, and Router C is an IBGP neighbor. The **neighbor** statement on Router B for Router A is pointing to the directly connected IP address to reach Router A. However, the **neighbor** statement on Router B for Router C points to Router C's loopback interface. Router B has multiple paths to reach Router C. If Router B pointed to the 192.168.3.2 IP address of Router C and that interface went down, Router B would be unable to reestablish the BGP session until the link came back up. By pointing to the loopback interface of Router C, the link stays established as long as any path to

Router C is available. (Router C should also point to Router B's loopback address in its configuration, and both Routers B and C will need a static or IGP route to each other's loopback interface.)

Line 4 in the configuration forces Router B to use its loopback 0 address, 192.168.2.1, as the source IP address when sending an update to Router C, 192.168.2.2.

In line 5, the next-hop address for networks that can be reached through Router B is changed from its default of 10.1.1.1 with the **neighbor next-hop-self** command. This command sets the next-hop address to the source IP address of the routing update, which is Router B's loopback 0 address, as set by the **neighbor update-source** command.

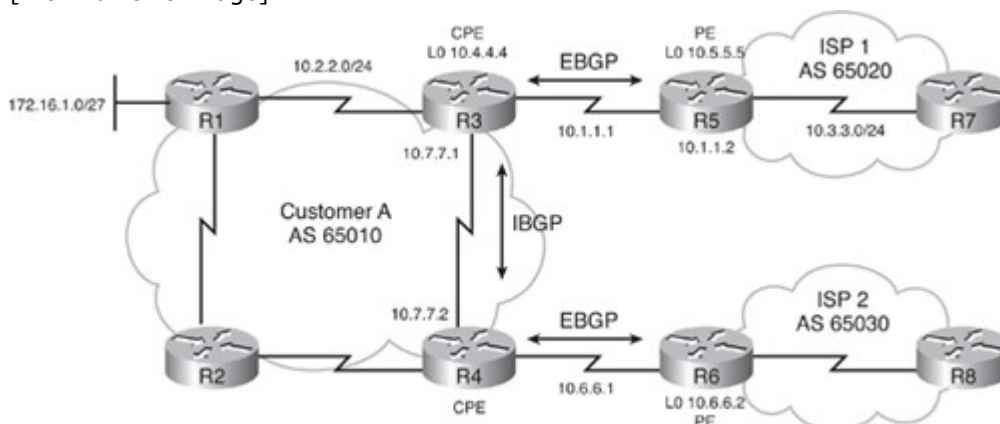
Lines 6, 7, and 8 tell BGP which networks to advertise. Line 6 contains a subnet of a Class B address using the **mask** option. Lines 7 and 8 contain two **network** statements for the two Class C networks that connect Routers B and C. The default mask for these networks is 255.255.255.0, so it is not needed in the commands.

In line 9, synchronization is disabled (this command is not needed if the router is running Cisco IOS Software Release 12.2(8)T or later, because then synchronization is off by default). If Router A is advertising 172.20.0.0 in BGP, Router B receives that route and advertises it to Router C. In this network, synchronization can be off because all the routers within the transit path in autonomous system are running IBGP.

Figure 6-45 illustrates another example BGP network in which two CPE routers are connected to ISP PE routers, in two different ISPs. EBGP is configured between both sets of CPE and PE. To provide reliable connectivity to both ISPs, the CPE routers must exchange BGP routes. Therefore, an IBGP session must be configured between the two CPE routers.

Figure 6-45. CPE and PE BGP Connection Example.

[View full size image]



Example 6-18 provides the partial configuration of one of the CPE routers, R3. The EBGP neighbor relationship to router R5 and IBGP neighborship to router R6 is configured. Two network commands are used to advertise two networks. Because network 172.16.0.0 is not directly connected, a static route to null0 interface is configured so that network 172.16.0.0 can be advertised into BGP.

Example 6-18. Configuration of Router R3 in Figure 6-45

```
router bgp 65010
 neighbor 10.7.7.2 remote-as 65010
 neighbor 10.1.1.2 remote-as 65020
 network 10.2.2.0 mask 255.255.255.0
 network 172.16.0.0 mask 255.255.0.0
<output omitted>

ip route 172.16.0.0 255.255.0.0 Null0
```



## Verifying and Troubleshooting BGP

You can verify BGP operation using **show EXEC** commands, including the following:

- **show ip bgp**— Displays entries in the BGP topology database (BGP table). Specify a network number to get more specific information about a particular network.
- **show ip bgp rib-failure**— Displays BGP routes that were not installed in the routing information base (RIB), and the reason that they were not installed.
- **show ip bgp neighbors**— Displays detailed information about the TCP and BGP connections to neighbors.
- **show ip bgp summary**— Displays the status of all BGP connections.

Use the **show ip bgp ?** command on a router to see other BGP show commands.

**debug** commands display events as they happen on the router. For BGP, the **debug ip bgp** privileged EXEC command has many options, including the following:

- **dampening**— BGP dampening
- **events**— BGP events
- **keepalives**— BGP keepalives
- **updates**— BGP updates

### Caution

Use caution when executing **debug** commands because they may consume a lot of router resources and could cause problems in a busy production network. Debugging output takes priority over other network traffic. Too much debug output may severely reduce the performance of the router or even render it unusable in the worst case.

The following sections provide sample output for some of these commands.

### show ip bgp Command Output Example

Use the **show ip bgp** command to display the BGP topology database (the BGP table).

Example 6-19 is a sample output for the **show ip bgp** command. The status codes are shown at the beginning of each line of output, and the origin codes are shown at the end of each line. In this output, most of the rows have an asterisk (\*) in the first column. This means that the next-hop address (in the fifth column) is valid. Remember, for BGP the next-hop address is not always on a router that is directly connected to this router. Other options for the first column are as follows:

- An *s* indicates that the specified routes are suppressed (usually because routes have been summarized and only the summarized route is being sent).
- A *d*, for dampening, indicates that the route is being dampened (penalized) for going up and down too often. Although the route might be up right now, it is not advertised until the penalty has expired.
- An *h*, for history, indicates that the route is unavailable and is probably down. Historic information about the route exists, but a best route does not exist.
- An *r*, for RIB failure, indicates that the route was not installed in the RIB. The reason that the route is not installed can be displayed using the **show ip bgp rib-failure** command, as described in the next section.
- An *S*, for stale, indicates that the route is stale. (This is used in a nonstop forwarding-aware router.)

### Example 6-19. show ip bgp Command Output

Code View: Scroll / Show All

RouterA#**show ip bgp**

BGP table version is 14, local router ID is 172.31.11.1

Status codes: s suppressed, d damped, h history, \* valid, > best, i - internal, r

RIB-failure, S Stale

Origin codes: i - IGP, e - EGP, ? - incomplete

| Network          | Next Hop   | Metric | LocPrf | Weight  | Path    |
|------------------|------------|--------|--------|---------|---------|
| *> 10.1.0.0/24   | 0.0.0.0    | 0      |        | 32768   | i       |
| * i              | 10.1.0.2   | 0      | 100    | 0       | i       |
| *> 10.1.1.0/24   | 0.0.0.0    | 0      |        | 32768   | i       |
| *>i10.1.2.0/24   | 10.1.0.2   | 0      | 100    | 0       | i       |
| *> 10.97.97.0/24 | 172.31.1.3 |        |        | 0 64998 | 64997 i |



```

* 172.31.11.4 0 64999 64997 i
* i 172.31.11.4 0 100 0 64999 64997 i
*> 10.254.0.0/24 172.31.1.3 0 0 64998 i
* 172.31.11.4 0 64999 64998 i
* i 172.31.1.3 0 100 0 64998 i
r> 172.31.1.0/24 172.31.1.3 0 0 64998 i
r 172.31.11.4 0 64999 64998 i
r i 172.31.1.3 0 100 0 64998 i
*> 172.31.2.0/24 172.31.1.3 0 0 64998 i
<output omitted>

```

A greater-than sign (>) in the second column indicates the best path for a route selected by BGP. This route is offered to the IP routing table.

The third column is either blank or has an *i* in it. If it is blank, BGP learned that route from an external peer. If it has an *i*, an IBGP neighbor advertised this route to this router.

The fourth column lists the networks that the router learned.

The fifth column lists all the next-hop addresses for each route. This next-hop address column might contain 0.0.0.0, which signifies that this router originated the route.

The next three columns list three BGP path attributes associated with the path: metric (MED), local preference, and weight.

The column with the "Path" header may contain a sequence of autonomous systems in the path. From left to right, the first autonomous system listed is the adjacent autonomous system from which this network was learned. The last number (the rightmost autonomous system number) is this network's originating autonomous system. The autonomous system numbers between these two represent the exact path that a packet takes back to the originating autonomous system. If the path column is blank, the route is from the current autonomous system.

The last column signifies how this route was entered into BGP on the original router (the origin attribute). If the last column has an *i* in it, the original router probably used a **network** command to introduce this network into BGP. The character *e* signifies that the original router learned this network from EGP, which is the historic predecessor to BGP. A question mark (?) signifies that the original BGP process cannot absolutely verify this network's availability because it is redistributed from an IGP into the BGP process.

show ip bgp rib-failure Command Output Example

Use the show ip bgp rib-failure command to display BGP routes that were not installed in the RIB, and the reason that they were not installed. In Example 6-20 the displayed routes were not installed because a route or routes with a better administrative distance already existed in the RIB.

Example 6-20. show ip bgp rib-failure Command Output

```

Code View: Scroll / Show All
RouterA#show ip bgp rib-failure
Network Next Hop RIB-failure RIB-NH Matches
172.31.1.0/24 172.31.1.3 Higher admin distance n/a
172.31.11.0/24 172.31.11.4 Higher admin distance n/a

```

show ip bgp summary Command Output Example

The show ip bgp summary command is one way to verify the BGP neighbor relationship. Example 6-21 presents sample output from this command. Here are some of the highlights:

- **BGP router identifier**— IP address that all other BGP speakers recognize as representing this router.
- **BGP table version**— Increases in increments when the BGP table changes.

- **Main routing table version**— Last version of BGP database that was injected into the main routing table.
- **Neighbor**— The IP address, used in the **neighbor** statement, with which this router is setting up a relationship.
- **Version (V)**— The version of BGP this router is running with the listed neighbor.
- **AS**— The listed neighbor's autonomous system number.
- **Messages received (MsgRcvd)**— The number of BGP messages received from this neighbor.
- **Messages sent (MsgSent)**— The number of BGP messages sent to this neighbor.
- **TblVer**— The last version of the BGP table that was sent to this neighbor.
- **In queue (InQ)**— The number of messages from this neighbor that are waiting to be processed.
- **Out queue (OutQ)**— The number of messages queued and waiting to be sent to this neighbor. TCP flow control prevents this router from overwhelming a neighbor with a large update.
- **Up/down**— The length of time this neighbor has been in the current BGP state (established, active, or idle).
- **State**— The current state of the BGP session: active, idle, open sent, open confirm, or idle (admin). The admin state is new to Cisco IOS Software Release 12.0; it indicates that the neighbor is administratively shut down. This state is created by using the `neighbor ip-address shutdown` router configuration command. (Neighbor states are discussed in more detail in the "Understanding and Troubleshooting BGP Neighbor States" section, later in this chapter.) Note that if the session is in the established state, a state is not displayed. Instead, a number representing the PfxRcd is displayed, as described next.

#### Note

If the State field of the **show ip bgp summary** command indicates active, the router is attempting to create a TCP connection to this neighbor.

- **Prefix received (PfxRcd)**— When the session is in the established state, this value represents the number of BGP network entries received from this neighbor.

Example 6-21. *show ip bgp summary* Command Output

```
RouterA#show ip bgp summary
BGP router identifier 10.1.1.1, local AS number 65001
BGP table version is 124, main routing table version 124
9 network entries using 1053 bytes of memory
22 path entries using 1144 bytes of memory
12/5 BGP path/bestpath attribute entries using 1488 bytes of memory
6 BGP AS-PATH entries using 144 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 3829 total bytes of memory
BGP activity 58/49 prefixes, 72/50 paths, scan interval 60 secs

Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd
10.1.0.2 4 65001 11 11 124 0 0 00:02:28 8
172.31.1.3 4 64998 21 18 124 0 0 00:01:13 6
172.31.11.4 4 64999 11 10 124 0 0 00:01:11 6
```

#### Note

Example output of the `show ip bgp neighbors` command is provided in the "Understanding and Troubleshooting BGP Neighbor States" section, later in this chapter.

## debug ip bgp updates Command Output Example

Example 6-22 shows partial output from the debug ip bgp updates command on Router A after the clear ip bgp command is issued to clear BGP sessions with its IBGP neighbor 10.1.0.2.

Example 6-22. *debug ip bgp updates* Command Output

Code View: Scroll / Show All

RouterA#**debug ip bgp updates**

Mobile router debugging is on for address family: IPv4 Unicast

RouterA#**clear ip bgp 10.1.0.2**

<output omitted>

\*May 24 11:06:41.309: %BGP-5-ADJCHANGE: neighbor 10.1.0.2 Up

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 send UPDATE (format) 10.1.1.0/24, next 10.1.0.1, metric 0, path Local

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 send UPDATE (prepend, chgflags: 0x0) 10.1.0.0/24, next 10.1.0.1, metric 0, path Local

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 NEXT\_HOP part 1 net 10.97.97.0/24, next 172.31.11.4

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 send UPDATE (format) 10.97.97.0/24, next 172.31.11.4, metric 0, path 64999 64997

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 NEXT\_HOP part 1 net 172.31.22.0/24, next 172.31.11.4

\*May 24 11:06:41.309: BGP(0): 10.1.0.2 send UPDATE (format) 172.31.22.0/24, next 172.31.11.4, metric 0, path 64999

<output omitted>

\*May 24 11:06:41.349: BGP(0): 10.1.0.2 rcvd UPDATE w/ attr: nexthop 10.1.0.2, origin i, localpref 100, metric 0

\*May 24 11:06:41.349: BGP(0): 10.1.0.2 rcvd 10.1.2.0/24

\*May 24 11:06:41.349: BGP(0): 10.1.0.2 rcvd 10.1.0.0/24

After the neighbor adjacency is reestablished, Router A creates and sends updates to 10.1.0.2. The first update highlighted in the example, **10.1.1.0/24, next 10.1.0.1**, is an update about network 10.1.1.0/24, with a next hop of 10.1.0.1, which is Router A's address. The second update highlighted in the example, **10.97.97.0/24, next 172.31.11.4**, is an update about network 10.97.97.0/24, with a next hop of 172.31.11.4, which is the address of one of Router A's EBGP neighbors. The EBGP next-hop address is being carried into IBGP.

Router A later receives updates from 10.1.0.2. The update highlighted in the example contains a path to two networks, 10.1.2.0/24 and 10.1.0.0/24.

### Understanding and Troubleshooting BGP Neighbor States

After the TCP handshake is complete, the BGP application tries to set up a session with the neighbor. BGP is a state machine that takes a router through the following states with its neighbors:

- **Idle**— The router is searching the routing table to see whether a route exists to reach the neighbor.
- **Connect**— The router found a route to the neighbor and has completed the three-way TCP handshake.
- **Open sent**— An open message was sent, with the parameters for the BGP session.
- **Open confirm**— The router received agreement on the parameters for establishing a session.

Alternatively, the router goes into active state if there is no response to the open message.

- **Established**— Peering is established and routing begins.

After you enter the **neighbor** command, BGP starts in the *idle* state, and the BGP process checks that it has a route to the IP address listed. BGP should be in the idle state for only a few seconds. However, if BGP does not find a route to the neighboring IP address, it stays in the idle state. If it finds a route, it goes to the *connect* state when the TCP handshaking synchronize acknowledge (SYN ACK) packet returns (when the TCP three-way handshake is complete). After the TCP connection is set up, the BGP process creates a BGP

open message and sends it to the neighbor. After BGP dispatches this open message, the BGP peering session changes to the *open sent* state. If there is no response for 5 seconds, the state changes to the *active* state. If a response does come back in a timely manner, BGP goes to the *open confirm* state and starts scanning (evaluating) the routing table for the paths to send to the neighbor. When these paths have been found, BGP then goes to the *established* state and begins routing between the neighbors.

The BGP state is shown in the last column of the **show ip bgp summary** command output.

#### Note

You can observe the states that two BGP routers are going through to establish a session using **debug** commands. In Cisco IOS Software Release 12.4, you can use the **debug ip bgp ipv4 unicast** command (or the **debug bgp ipv4 unicast events** command) to see this process. In earlier IOS releases, only the **debug ip bgp events** command was available to provide similar output.

#### Idle State Troubleshooting

The idle state indicates that the router does not know how to reach the IP address listed in the **neighbor** statement. The router is idle for one of the following reasons:

- It is waiting for a static route to that IP address or network to be configured.
- It is waiting for the local routing protocol (IGP) to learn about this network through an advertisement from another router.

The most common reason for the idle state is that the neighbor is not announcing the IP address or network that the **neighbor** statement of the router is pointing to. Check the following two conditions to troubleshoot this problem:

- Ensure that the neighbor announces the route in its local routing protocol (IGP) (for IBGP neighbors).
- Verify that you have not entered an incorrect IP address in the **neighbor** statement.

#### Active State Troubleshooting

If the router is in the active state, this means that it has found the IP address in the **neighbor** statement and has created and sent out a BGP open packet but has not received a response (an open confirm packet) back from the neighbor.

One common cause of this is when the neighbor does not have a return route to the source IP address.

Ensure that the source IP address or network of the packets is advertised into the local routing protocol (IGP) on the neighboring router.

Another common problem associated with the active state is when a BGP router attempts to peer with another BGP router that does not have a **neighbor** statement peering back at the first router, or the other router is peering with the wrong IP address on the first router. Check to ensure that the other router has a **neighbor** statement peering at the correct address of the router that is in the active state.

If the state toggles between idle and active, the autonomous system numbers might be misconfigured. You see the following console message at the router with the wrong autonomous system number configured in the **neighbor** statement:

Code View: Scroll / Show All

```
%BGP-3-NOTIFICATION: sent to neighbor 172.31.1.3 2/2 (peer in wrong AS)
```

```
2 bytes FDE6
```

```
FFFF FFFF FFFF FFFF FFFF FFFF FFFF 002D 0104 FDE6 00B4 AC1F 0203 1002 0601
```

```
0400 0100 0102 0280 0002 0202 00
```

At the remote router, you see the following message:

Code View: Scroll / Show All

```
%BGP-3-NOTIFICATION: received from neighbor 172.31.1.1 2/2 (peer in wrong AS) 2 bytes FDE6
```

#### Established State

The desired state for a neighbor relationship is the established state. This state means that both routers agree to exchange BGP updates with one another and routing has begun. As mentioned, if the state column

in the **show ip bgp summary** command output is blank or has a number in it, BGP is in the established state, and the number shown is the number of routes that have been learned from this neighbor.

Use the **show ip bgp neighbors** command to display information about the BGP connections to neighbors. In Example 6-23, the BGP state is established, which means that the neighbors have established a TCP connection and the two peers have agreed to use BGP to communicate.

Example 6-23. *show ip bgp neighbors* Command Output

Code View: Scroll / Show All

RouterA#**show ip bgp neighbors**

BGP neighbor is 172.31.1.3, remote AS 64998, external link

BGP version 4, remote router ID 172.31.2.3

BGP state = Established, up for 00:19:10

Last read 00:00:10, last write 00:00:10, hold time is 180, keepalive interval is 60 seconds

Neighbor capabilities:

Route refresh: advertised and received(old & new)

Address family IPv4 Unicast: advertised and received

Message statistics:

InQ depth is 0

OutQ depth is 0

|                | Sent | Rcvd |
|----------------|------|------|
| Opens:         | 7    | 7    |
| Notifications: | 0    | 0    |
| Updates:       | 13   | 38   |

<output omitted>

#### Basic BGP Path Manipulation Using Route Maps

Manipulating path-selection criteria can affect the inbound and outbound traffic policies of an autonomous system. This section discusses path manipulation and how to manipulate the BGP local preference, MED, and weight attributes to influence BGP path selection, including the use of route maps. Manipulating path selection by extending the AS-path attribute is also described.

#### Note

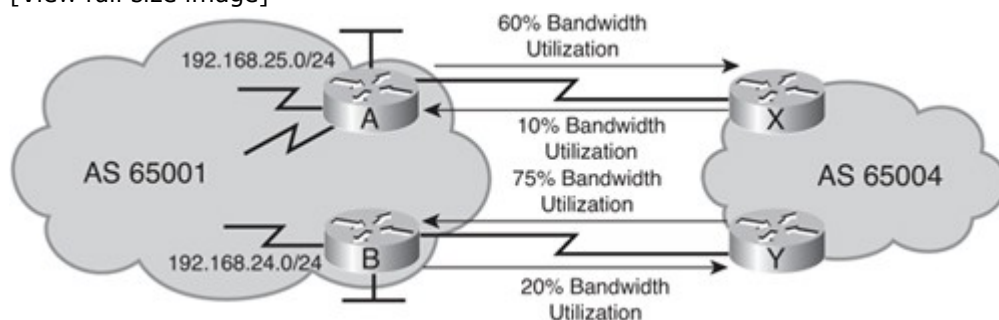
Recall that route maps and route map configuration are described in Chapter 4. This section includes some specifics for route map use with BGP.

## BGP Path Manipulation

Unlike local routing protocols, BGP was never designed to choose the quickest path. Rather, it was designed to manipulate traffic flow to maximize or minimize bandwidth use. Figure 6-46 demonstrates a common situation that can result when using BGP without any policy manipulation.

Figure 6-46. BGP Network Without Policy Manipulation.

[View full size image]



Using default settings for path selection in BGP might cause uneven use of bandwidth. In Figure 6-46, Router A in autonomous system 65001 is using 60 percent of its outbound bandwidth to Router X in 65004, but Router B is using only 20 percent of its outbound bandwidth. If this utilization is acceptable to the administrator, no manipulation is needed. But if the load averages 60 percent and has temporary bursts above 100 percent of the bandwidth, lost packets, higher latency, and higher CPU usage may result because of the number of packets being routed. When another link to the same location is available and is not heavily used, it makes sense to divert some of the traffic to the other path. To change outbound path selection from autonomous system 65001, the local preference attribute must be manipulated.

Recall that a higher local preference is preferred. To determine which path to manipulate, the administrator performs a traffic analysis on Internet-bound traffic by examining the most heavily visited addresses, web pages, or domain names. This information can usually be found by examining network management records or accounting information.

Assume that in Figure 6-46, 35 percent of all traffic from autonomous system 65001 has been going to <http://www.cisco.com>. The administrator can obtain the Cisco IP address or autonomous system number by performing a reverse Domain Name System (DNS) lookup or by going to ARIN ([athttp://www.arin.net](http://www.arin.net)) and looking up the autonomous system number of Cisco Systems or the address space assigned to the company. After this information has been determined, the administrator can use route maps to change the local preference to manipulate path selection for packets destined to Cisco's network.

Using a route map, Router B can announce, to all routers within autonomous system 65001, all routes to networks associated with Cisco's autonomous system with a higher local preference than Router A announces for routes to those networks. Because routers running BGP prefer routes with the highest local preference, other BGP routers in autonomous system 65001 send all traffic destined for Cisco's autonomous system to exit autonomous system 65001 via Router B. The outbound load for Router B increases from its previous load of 20 percent to account for the extra traffic from autonomous system 65001 destined for the Cisco networks. The outbound load for Router A, which was originally 60 percent, should decrease. This change will make the outbound load on both links more balanced. The administrator should monitor the outbound loads and adjust the configuration accordingly as traffic patterns change over time.

Just as there was a loading issue outbound from autonomous system 65001, there can be a similar problem inbound. For example, if the inbound load to Router B has a much higher utilization than the inbound load to Router A, the BGP MED attribute can be used to manipulate how traffic enters autonomous system 65001. Router A in autonomous system 65001 can announce a lower MED for routes to network 192.168.25.0/24 to autonomous system 65004 than Router B announces. This MED recommends to the next autonomous system how to enter autonomous system 65001. However, MED is not considered until later in the BGP path-selection process than local preference. Therefore, if the administrator for autonomous system 65004 prefers that traffic leave the autonomous system via Router Y (to Router B in autonomous system 65001), Router Y should be configured to announce a higher local preference to the BGP routers in autonomous system 65004 for routes to network 192.168.25.0/24 than Router X announces. The local preference that Routers X and Y advertise to other BGP routers in autonomous system 65004 is evaluated before the MED coming from Routers A and B. MED is considered a *recommendation* because the receiving autonomous system can override it by manipulating another variable that is considered before the MED is evaluated.

For example, using Figure 6-46, assume that 55 percent of all inbound traffic is going to the 192.168.25.0/24 subnet (on Router A). The inbound utilization to Router A is averaging only 10 percent, but the inbound

utilization to Router B is averaging 75 percent. If the inbound load for Router B spikes to more than 100 percent, this may cause problems such as the route flapping, resulting in some of the sessions crossing that link being lost. For example, if these sessions were purchases being made on autonomous system 65001 web servers, revenue would be lost, which is something administrators want to avoid. If autonomous system 65001 were set to prefer to have all traffic that is going to 192.168.25.0/24 enter through Router A, the load inbound on Router A should increase, and the load inbound on Router B should decrease.

If load averages less than 50 percent for an outbound or inbound case, path manipulation might not be needed. However, as soon as a link starts to reach its capacity for an extended period of time, either more bandwidth is needed or path manipulation should be considered. The administrator should monitor the inbound loads and adjust the configuration accordingly as traffic patterns change over time.

#### Changing the Weight

Recall that the weight attribute influences only the local router. Routes with a higher weight are preferred.

#### Changing the Weight for All Updates from a Neighbor

The neighbor {ip-address | peer-group-name} weight weight router configuration command is used to assign a weight to updates from a neighbor connection, as described in Table 6-8.

Table 6-8. *neighbor weight* Command Description

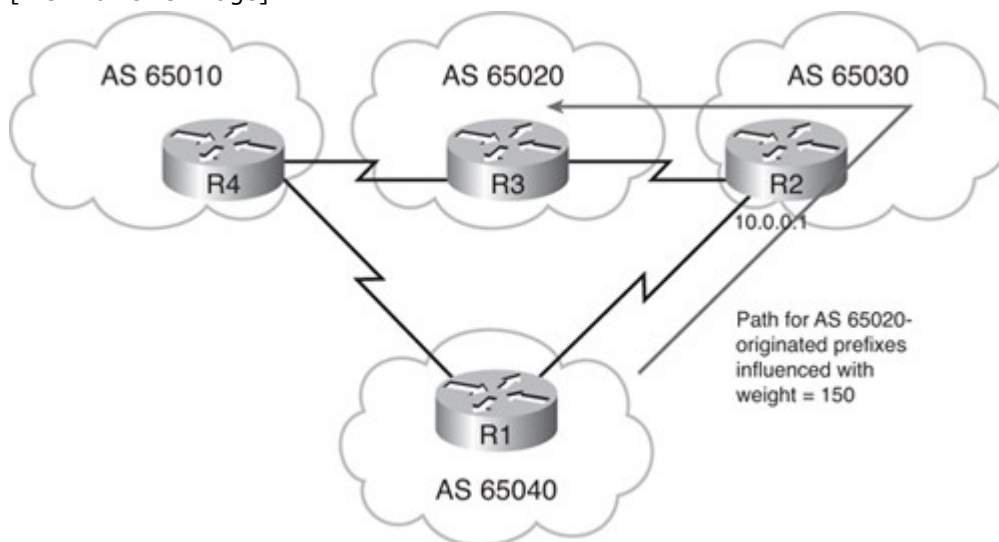
| Parameter       | Description                                                                                                                                                                    |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ip-address      | The BGP neighbor's IP address.                                                                                                                                                 |
| peer-group-name | The name of a BGP peer group.                                                                                                                                                  |
| Weight          | The weight to assign. Acceptable values are 0 to 65535. The default is 32768 for local routes (routes that the router originates). Other routes have a weight of 0 by default. |

#### Changing the Weight Using Route Maps

The network shown in Figure 6-47 is used as an example to demonstrate how to change the weight attribute using route maps. The partial configuration of Router R1 is shown in Example 6-24.

Figure 6-47. Setting Weight with Route Map Example.

[View full size image]



Example 6-24. Configuration of Router R1 in Figure 6-47

```

<output omitted>
!
router bgp 65040

```



```

neighbor 10.0.0.1 route-map set-weight in
<output omitted>
!
route-map set-weight permit 10
 match as-path 10
 set weight 150
!
route-map set-weight permit 20
 set weight 100
!
ip as-path access-list 10 permit _65020$

```

In this example, the routing policy dictates the selection of autonomous system 65030 as the primary way out of autonomous system 65040 for the traffic destined to any network originated by the autonomous system 65020. This is achieved by placing a higher weight (150) on all incoming announcements from autonomous system 65030 (from neighbor 10.0.0.1), which carry the information about the network originated in autonomous system 65020.

The first line of the route map called set-weight is a **permit** statement with a sequence number of 10; it defines the first **route-map** statement. The match condition for this statement checks the AS-path attributes of updates to see which are permitted by autonomous system access list 10.

The match condition is defined by the **match as-path path-list-number** route-map configuration command. The *path-list-number* parameter is the number of the autonomous system path access list, which is 10 in this example.

The autonomous system path access list is defined by the ip as-path access-list acl-number {permit | deny} regexp global configuration command. The parameters of this command are described in Table 6-9.

Table 6-9. *ip as-path access-list* Command Description

| Parameter  | Description                                                                                                                                                                                                                                                                                                                                              |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| acl-number | Number from 1 to 500 that specifies the AS-path access list number.                                                                                                                                                                                                                                                                                      |
| regexp     | Regular expression that defines the AS-path filter. The autonomous system number is expressed in the range from 1 to 65535. See the "Regular Expressions" appendix in the Cisco IOS Terminal Services Configuration Guide, available at <a href="http://www.cisco.com">http://www.cisco.com</a> , for information about configuring regular expressions. |

In this example, autonomous system access list 10 permits updates whose AS-path attribute ends with 65020 (note that the parameter is \_65020\$). These are updates originating in autonomous system 65020. The route map sets these updates to a weight of 150, with the **set weight 150** command.

The second statement in the route map called set-weight is a **permit** statement with a sequence number of 20; it does not have any **match** statements, so all remaining updates are permitted. These remaining updates have their weight set to 100, with the **set weight 100** command. The sequence number 20 (rather than 11) is chosen for the second statement in case other policies have to be implemented later before this statement.

This route map is linked to neighbor 10.0.0.1 as an inbound route map. Therefore, as Router R1 receives updates from 10.0.0.1, it processes them through the set-weight route map and sets the weight accordingly as the routes are placed in Router R1's BGP table.

## Setting Local Preference

Recall that local preference is used only within an autonomous system between IBGP speakers to determine the best path to leave the autonomous system to reach an outside network. The local preference is set to 100 by default; higher values are preferred.

### Note

If for some reason an EBGp neighbor did receive a local preference value (such as because of faulty software), the EBGp neighbor ignores it.

## Changing Local Preference for All Routes

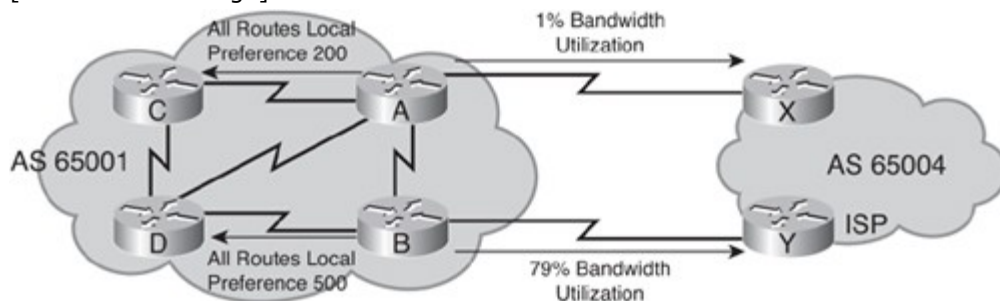
The **bgp default local-preference** value router configuration command changes the default local preference to the value specified. All BGP routes that are advertised include this local preference value. The value can be set to a number between 0 and 4294967295.

Manipulating the default local preference can have an immediate and dramatic effect on traffic flow leaving an autonomous system. Before making any changes to manipulate paths, the network administrator should perform a thorough traffic analysis to understand the effects of the change. For example, the configurations for Routers A and B in Figure 6-48 are shown in Examples 6-25 and 6-26, respectively. In this network, the administrator changed the default local preference for all routes on Router B to 500 and on Router A to 200. All BGP routers in autonomous system 65001 send all traffic destined for the Internet to Router B, causing its outbound utilization to be much higher and the utilization out Router A to be reduced to a minimal amount.

This change is probably not what the network administrator intended. Instead, the network administrator should use route maps to set only routes for specific networks to have a higher local preference through Router B, to decrease some of the original outbound load that was being sent out Router A.

Figure 6-48. Setting a Default Local Preference for All Routes.

[View full size image]



Example 6-25. Configuration for Router A in Figure 6-48

```
router bgp 65001
 bgp default local-preference 200
```

Example 6-26. Configuration for Router B in Figure 6-48

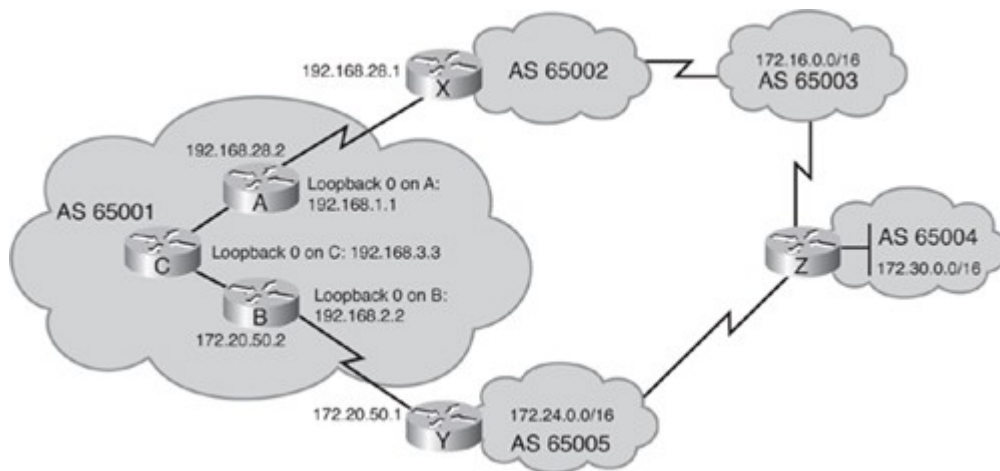
```
router bgp 65001
 bgp default local-preference 500
```

## Local Preference Example

Figure 6-49 illustrates a sample network running BGP that will be used to demonstrate how local preference can be manipulated. This network initially has no commands configured to change the local preference.

Figure 6-49. Network for Local Preference Example.

[View full size image]



Example 6-27 illustrates the BGP forwarding table on Router C in Figure 6-49, showing only the networks of interest to this example:

- 172.16.0.0 in autonomous system 65003
- 172.24.0.0 in autonomous system 65005
- 172.30.0.0 in autonomous system 65004

Example 6-27. BGP Table for Router C in Figure 6-49 Without Path Manipulation

Code View: Scroll / Show All

RouterC#**show ip bgp**

BGP table version is 7, local router ID is 192.168.3.3

Status codes: s suppressed, d damped, h history, \* valid, > best, i - internal, r RIB-failure, S Stale

Origin codes: i - IGP, e - EGP, ? - incomplete

| Network       | Next Hop     | Metric | LocPrf | Weight                  | Path |
|---------------|--------------|--------|--------|-------------------------|------|
| * i172.16.0.0 | 172.20.50.1  | 100    | 0      | 65005 65004 65003       | i    |
| *>i           | 192.168.28.1 | 100    | 0      | 65002 65003             | i    |
| *>i172.24.0.0 | 172.20.50.1  | 100    | 0      | 65005                   | i    |
| * i           | 192.168.28.1 | 100    | 0      | 65002 65003 65004 65005 | i    |
| *>i172.30.0.0 | 172.20.50.1  | 100    | 0      | 65005 65004             | i    |
| * i           | 192.168.28.1 | 100    | 0      | 65002 65003 65004       | i    |

The best path is indicated with a > in the second column of the output.

Each network has two paths that are loop free, have synchronization disabled, and have a valid next-hop address (that can be reached from Router C). All routes have a weight of 0 and a default local preference of 100, so Steps 1 and 2 in the BGP path-selection process do not select the best route.

This router does not originate any of the routes (Step 3), so the process moves to Step 4, and BGP uses the shortest AS-path to select the best routes as follows:

- For network 172.16.0.0, the shortest AS-path of two autonomous systems (65002 65003) is through the next hop of 192.168.28.1.
- For network 172.24.0.0, the shortest AS-path of one autonomous system (65005) is through the next hop of 172.20.50.1.
- For network 172.30.0.0, the shortest AS-path of two autonomous systems (65005 65004) is through the next hop of 172.20.50.1.

Neither Routers A nor B are using the **neighbor next-hop-self** command in this example.

A traffic analysis reveals the following:

- The link going through Router B to 172.20.50.1 is heavily used, and the link through Router A to 192.168.28.1 is hardly used at all.
- The three largest-volume destination networks on the Internet from autonomous system 65001 are 172.30.0.0, 172.24.0.0, and 172.16.0.0.
- Thirty percent of all Internet traffic is going to network 172.24.0.0 (via Router B), 20 percent is going to network 172.30.0.0 (via Router B), and 10 percent is going to network 172.16.0.0 (via Router A). The other 40 percent is going to other destinations. Thus, considering only these three largest-volume destinations, only 10 percent of the traffic is using the link out Router A to 192.168.28.1, and 50 percent of the traffic is using the link out Router B to 172.20.50.1.

The network administrator has decided to divert traffic to network 172.30.0.0 and send it out Router A to the next hop of 192.168.28.1, so that the loading between Routers A and B is more balanced.

#### Changing Local Preference Using Route Maps

A route map is added to Router A in Figure 6-49, as shown in the BGP configuration in Example 6-28. The route map alters the network 172.30.0.0 BGP update from Router X (192.168.28.1) to have a high local preference value of 400 so that it will be more preferred.

Example 6-28. BGP Configuration for Router A in Figure 6-49 with a Route Map

```
router bgp 65001
 neighbor 192.168.2.2 remote-as 65001
 neighbor 192.168.3.3 remote-as 65001
 neighbor 192.168.2.2 remote-as 65001 update-source loopback0
 neighbor 192.168.3.3 remote-as 65001 update-source loopback0
 neighbor 192.168.28.1 remote-as 65002
 neighbor 192.168.28.1 route-map local_pref in
!

route-map local_pref permit 10
 match ip address 65
 set local-preference 400
!

route-map local_pref permit 20
!

access-list 65 permit 172.30.0.0 0.0.255.255
```

The first line of the route map called `local_pref` is a **permit** statement with a sequence number of 10; it defines the first **route-map** statement. The **match** condition for this statement checks all networks to see which are permitted by access list 65. Access list 65 permits all networks that start with the first two octets of 172.30.0.0. The route map sets routes for these networks to a local preference of 400, with the **set local-preference 400** command.

The second statement in the route map called `local_pref` is a **permit** statement with a sequence number of 20, but it does not have any **match** or **set** statements. This statement is similar to a **permit any** statement in an access list. Because there are no **match** conditions for the remaining networks, they are all permitted with their current settings. In this case, the local preference for routes for networks 172.16.0.0 and 172.24.0.0 stays set at the default of 100. The sequence number 20 (rather than 11) is chosen for the second statement in case other policies have to be implemented later before this statement.

This route map is linked to neighbor 192.168.28.1 as an inbound route map. Therefore, as Router A receives updates from 192.168.28.1, it processes them through the `local_pref` route map and sets the local preference accordingly as the routes are placed in Router A's BGP forwarding table.

Example 6-29 illustrates the BGP table on Router C in Figure 6-49, after the route map has been applied on Router A and the BGP sessions have been reset. Router C learns about the new local preference value (400) coming from Router A for network 172.30.0.0. The only difference in this table compared to the original in Example 6-27 is that the best route to network 172.30.0.0 is now through 192.168.28.1 because its local preference of 400 is higher than the local preference of 100 for the next hop of 172.20.50.1. The AS-path through 172.20.50.1 is still shorter than the path through 192.168.28.1, but AS-path length is not evaluated

until Step 4, whereas local preference is examined in Step 2. Therefore, the higher local preference path was chosen as the best path.

Example 6-29. BGP Table for Router C in Figure 6-49 with a Route Map for Local Preference

Code View: Scroll / Show All

RouterC#**show ip bgp**

BGP table version is 7, local router ID is 192.168.3.3

Status codes: s suppressed, d damped, h history, \* valid, > best, i - internal, r RIB-failure, S Stale

Origin codes: i - IGP, e - EGP, ? - incomplete

| Network       | Next Hop     | Metric | LocPrf | Weight                    | Path |
|---------------|--------------|--------|--------|---------------------------|------|
| * i172.16.0.0 | 172.20.50.1  | 100    | 0      | 65005 65004 65003 i       |      |
| *>i           | 192.168.28.1 | 100    | 0      | 65002 65003 i             |      |
| *>i172.24.0.0 | 172.20.50.1  | 100    | 0      | 65005 i                   |      |
| * i           | 192.168.28.1 | 100    | 0      | 65002 65003 65004 65005 i |      |
| * i172.30.0.0 | 172.20.50.1  | 100    | 0      | 65005 65004 i             |      |
| *>i           | 192.168.28.1 | 400    | 0      | 65002 65003 65004i        |      |

#### Setting the AS-Path

It is complicated to influence other autonomous systems to select a particular path for traffic that is returning to a specific autonomous system. Because it is unlikely that the operator of an autonomous system can request changes in router configurations in another autonomous system, it is nearly impossible to influence another autonomous system to select the desired path based on the weight and local preference attributes, because these require configuration changes in the neighboring autonomous system.

#### Note

Communities, described in Appendix C, can be used to set attributes, including weight and local preference, in other autonomous systems. For example, an ISP with multihomed customers could allow those customers to set the local preference using communities.

As we have seen, by default, if no BGP path selection tools are configured to influence traffic flow, BGP uses the shortest autonomous system path, regardless of available bandwidth.

One way that an autonomous system can attempt to influence incoming traffic flow is by sending out EBGp updates with an *extended AS-path attribute* for undesired paths. The autonomous system path is extended with multiple copies of the autonomous system number of the sender. The receiver of this update is less likely to select the path as the best because its AS-path attribute appears to be longer. This feature is called *AS-path prepending*. There is no mechanism to calculate the optimal required prepended AS-path length, because the administrator of an autonomous system has no control over whether other autonomous systems are also doing prepending.

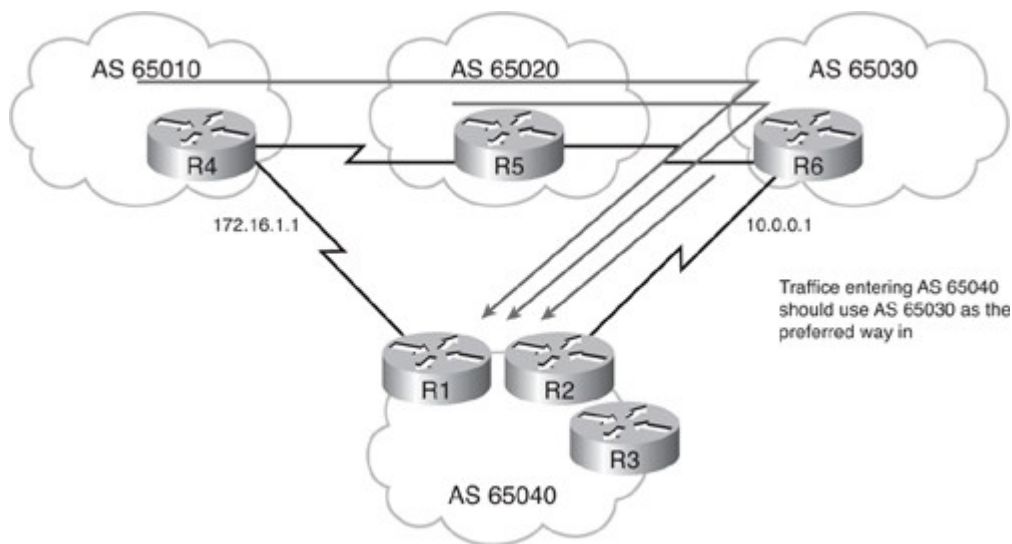
To avoid clashes with BGP loop-prevention mechanisms, no other autonomous system number, except that of the sending autonomous system, should be prepended to the AS-path attribute. If another autonomous system number is prepended in the autonomous system path, the routers in the autonomous system that has been prepended will reject the update because of BGP loop-prevention mechanisms.

You can configure prepending on a router for all routing updates that you send to a neighbor or only on a subset of them.

Figure 6-50 shows an example network, and Example 6-30 is a partial configuration for Router R1.

Figure 6-50. AS-Path Prepending Example.

[View full size image]



Example 6-30. Configuration of Router R1 in Figure 6-50

```
<output omitted>
!
router bgp 65040
 neighbor 172.16.1.1 route-map set-AS-path out
!

route-map set-AS-path permit 10
 set as-path prepend 65040 65040 65040
```

The route map called set-AS-path has only one statement, a permit statement with a sequence number of 10. There is no match condition for this statement, so it matches all updates. The set as-path {tag | prepend as-path-string} route-map configuration command is used to modify the AS-path attribute. The parameters of this command are described in Table 6-10.

Table 6-10. *set as-path* Command Description

| Parameter                            | Description                                                                                                                                                                                                                                                           |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| tag                                  | Converts the tag of a route into an AS-path. Applies only when redistributing routes into BGP.                                                                                                                                                                        |
| <b>prepend</b> <i>as-path-string</i> | Prepends the <i>as-path-string</i> to the AS-path attribute of the route that is matched by the route map. The range of values is any valid autonomous system number from 1 to 65535. Multiple values can be entered. Applies to inbound and outbound BGP route maps. |

This route map is linked to neighbor 172.16.1.1 as an outbound route map. Therefore, as Router R1 sends updates to 172.16.1.1, it processes them through the set-AS-path route map and all updates sent to neighbor 172.16.1.1 are prepended three times with the autonomous system number of the sender (65040), making that path less preferable for the returning traffic.

#### Setting the MED

Recall that MED is used to decide how to enter an autonomous system when multiple paths exist between two autonomous systems and one autonomous system is trying to influence the incoming path from the

other autonomous system. Because MED is evaluated late in the BGP path-selection process (Step 6), it usually has no influence on the process. For example, an autonomous system receiving a MED for a route can change its local preference on how to leave the autonomous system to override what the other autonomous system is advertising with its MED value.

When comparing MED values for the same destination network in the BGP path-selection process, the lowest MED value is preferred.

#### Changing the MED for All Routes

The default MED value for each network an autonomous system owns and advertises to an EBGp neighbor is set to 0. To change this value, use the **default-metric number** router configuration command.

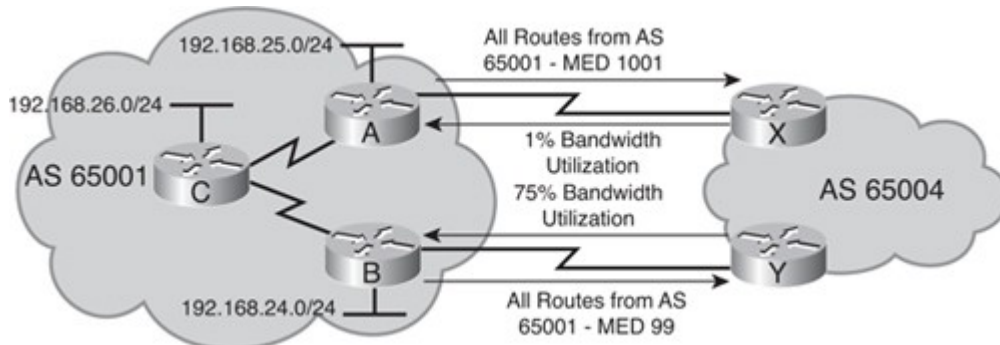
The *number* parameter is the MED value.

Manipulating the default MED value can have an immediate and dramatic effect on traffic flow entering your autonomous system. Before making any changes to manipulate the path, you should perform a thorough traffic analysis to ensure that you understand the effects of the change.

For example, the configurations of Routers A and B in Figure 6-51 are shown in Examples 6-31 and 6-32, respectively. The network administrator in autonomous system 65001 tries to manipulate how autonomous system 65004 chooses its path to reach routes in autonomous system 65001. By changing the default metric under the BGP process on Router A to 1001, Router A advertises a MED of 1001 for all routes to Router X. Router X then informs all the other routers in autonomous system 65004 of the MED through Router X to reach networks originating in autonomous system 65001. A similar event happens on Router B, but Router B advertises a MED of 99 for all routes to Router Y. All routers in autonomous system 65004 see a MED of 1001 through the next hop of Router A and a MED of 99 through the next hop of Router B to reach networks in autonomous system 65001. (The neighbor next-hop self command is not used on either Router X or Router Y.) If autonomous system 65004 has no overriding policy, all routers in autonomous system 65004 choose to exit their autonomous system through Router Y to reach the networks in autonomous system 65001. This traffic goes through Router B. This selection causes Router A's inbound bandwidth utilization to decrease to almost nothing except for BGP routing updates, and it causes the inbound utilization on Router B to increase and be used for all returning packets from autonomous system 65004 to autonomous system 65001.

Figure 6-51. Changing the Default MED for All Routes.

[View full size image]



Example 6-31. BGP Configuration for Router A in Figure 6-51

```
router bgp 65001
 default-metric 1001
```

Example 6-32. BGP Configuration for Router B in Figure 6-51

```
router bgp 65001
 default-metric 99
```

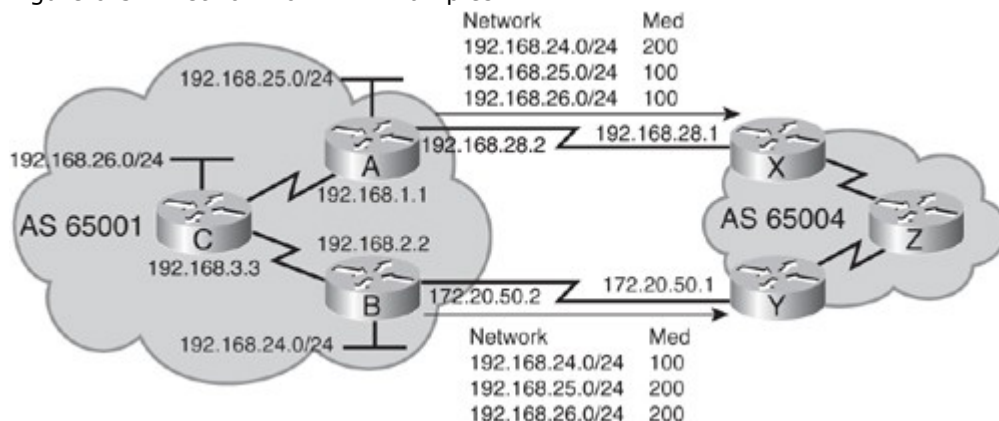
This situation is probably not what the network administrator intended. Instead, to load-share the inbound traffic to autonomous system 65001, the autonomous system 65001 network administrator should configure some networks to have a lower MED through Router B and other networks to have a lower MED through Router A. Route maps should be used to set the appropriate MED values for various networks.



## Changing the MED Using Route Maps

The network shown in Figure 6-52 is used as an example to demonstrate how to manipulate inbound traffic using route maps to change the BGP MED attribute. The intention of these route maps is to designate Router A as the preferred entry point to reach networks 192.168.25.0/24 and 192.168.26.0/24 and Router B as the preferred entry point to reach network 192.168.24.0/24. The other networks should still be reachable through each router in case of a link or router failure.

Figure 6-52. Network for MED Examples.



The MED is set outbound when advertising to an EBGP neighbor. In the configuration for Router A shown in Example 6-33, a route map named med\_65004 is linked to neighbor 192.168.28.1 (Router X) as an outbound route map. When Router A sends an update to neighbor 192.168.28.1, it processes the outbound update through route map med\_65004 and changes any values specified in a set command if the corresponding matchcommand conditions in that section of the route map are met.

Example 6-33. BGP Configuration for Router A in Figure 6-52 with a Route Map

```
router bgp 65001
 neighbor 192.168.2.2 remote-as 65001
 neighbor 192.168.3.3 remote-as 65001
 neighbor 192.168.2.2 update-source loopback0
 neighbor 192.168.3.3 update-source loopback0
 neighbor 192.168.28.1 remote-as 65004
 neighbor 192.168.28.1 route-map med_65004 out
!

route-map med_65004 permit 10
 match ip address 66
 set metric 100

route-map med_65004 permit 100
 set metric 200

!

access-list 66 permit 192.168.25.0.0 0.0.0.255
access-list 66 permit 192.168.26.0.0 0.0.0.255
```

The first line of the route map called med\_65004 is a **permit** statement with a sequence number of 10; it defines the first **route-map** statement. The **match** condition for this statement checks all networks to see which are permitted by access list 66. The first line of access list 66 permits any networks that start with the first three octets of 192.168.25.0, and the second line of access list 66 permits networks that start with the first three octets of 192.168.26.0.

Routes for any networks that are permitted by either of these lines will have the MED set to 100, by the **set metric 100** command. No other networks are permitted by this access list (there is an implicit **deny all** at the end of all access lists), so the MED of the other routes is not changed. These other routes must proceed to the next **route-map** statement in the med\_65004 route map.

The route map's second statement is a **permit** statement with a sequence number of 100. The route map does not have any **match** statements, just a **set metric 200** statement. This statement is similar to a **permit any** statement for route maps. Because the network administrator does not specify a **match** condition for this portion of the route map, all routes being processed through this section of the route map (sequence number 100) are permitted, and the MED of these routes is set to 200. If the network administrator did not set the MED to 200, by default it would have been set to a MED of 0. Because 0 is less than 100, the routes with a MED of 0 would have been the preferred paths to the networks in autonomous system 65001.

Similarly, the configuration for Router B is shown in Example 6-34. A route map named med\_65004 is linked to neighbor 172.20.50.1 as an outbound route map. Before Router B sends an update to neighbor 172.20.50.1, it processes the outbound update through route map med\_65004, and changes any values specified in a set command if the preceding match command conditions in that section of the route map are met.

Example 6-34. BGP Configuration for Router B in Figure 6-52 with a Route Map

```
router bgp 65001
 neighbor 192.168.1.1 remote-as 65001
 neighbor 192.168.3.3 remote-as 65001
 neighbor 192.168.1.1 update-source loopback0
 neighbor 192.168.3.3 update-source loopback0
 neighbor 172.20.50.1 remote-as 65004
 neighbor 172.20.50.1 route-map med_65004 out
!

route-map med_65004 permit 10
 match ip address 66
 set metric 100
route-map med_65004 permit 100
 set metric 200
!

access-list 66 permit 192.168.24.0.0 0.0.0.255
```

The first line of the route map called med\_65004 is a **permit** statement with a sequence number of 10; it defines the first **route-map** statement. The **match** condition for this statement checks all networks to see which are permitted by access list 66. Access list 66 on Router B permits any networks that start with the first three octets of 192.168.24.0.

Routes for any networks that are permitted by this line have the MED set to 100 by the route map. No other networks are permitted by this access list, so the MED of the other routes is unchanged. These other routes must proceed to the next **route-map** statement in the med\_65004 route map.

The second statement of the route map is a **permit** statement with a sequence number of 100, but it does not have any **match** statements, just a **set metric 200** statement. This statement is similar to a **permit any** statement for route maps. Because the network administrator does not specify a **match** condition for this portion of the route map, all routes being processed through this section of the route map are permitted, and the MED of these routes is set to 200. If the network administrator did not set the MED to 200, by default it would have been set to a MED of 0. Because 0 is less than 100, the routes with a MED of 0 would have been the preferred paths to the networks in autonomous system 65001.

Example 6-35 shows the BGP forwarding table on Router Z in autonomous system 65004 indicating the networks learned from autonomous system 65001. (Other networks that do not affect this example have been omitted.) Remember that in this command output, the MED is shown in the column labeled Metric.

Example 6-35. BGP Table for Router Z in Figure 6-52 with a Route Map

Code View: Scroll / Show All

RouterZ#**show ip bgp**

BGP table version is 7, local router ID is 192.168.1.1

Status codes: s suppressed, d damped, h history, \* valid, > best, i - internal,  
r RIB-failure, S Stale

Origin codes: i - IGP, e - EGP, ? - incomplete

| Network         | Next Hop     | Metric | LocPrf | Weight | Path    |
|-----------------|--------------|--------|--------|--------|---------|
| *>i192.168.24.0 | 172.20.50.2  | 100    | 100    | 0      | 65001 i |
| * i             | 192.168.28.2 | 200    | 100    | 0      | 65001 i |
| * i192.168.25.0 | 172.20.50.2  | 200    | 100    | 0      | 65001 i |
| *>i             | 192.168.28.2 | 100    | 100    | 0      | 65001 i |
| * i192.168.26.0 | 172.20.50.2  | 200    | 100    | 0      | 65001 i |
| *>i             | 192.168.28.2 | 100    | 100    | 0      | 65001 i |

Router Z has multiple paths to reach each network. These paths all have valid next-hop addresses, synchronization disabled, and the paths are loop free. All networks have a weight of 0 and a local preference of 100, so Steps 1 and 2 in the route-selection decision process do not determine the best path. None of the routes were originated by this router or any router in autonomous system 65004. All networks came from autonomous system 65001, so Step 3 does not apply. All networks have an AS-path of one autonomous system (65001) and were introduced into BGP with **network** statements (i is the origin code), so Steps 4 and 5 are equal. The route-selection decision process therefore gets to Step 6, which states that BGP chooses the lowest MED if all preceding steps are equal or do not apply.

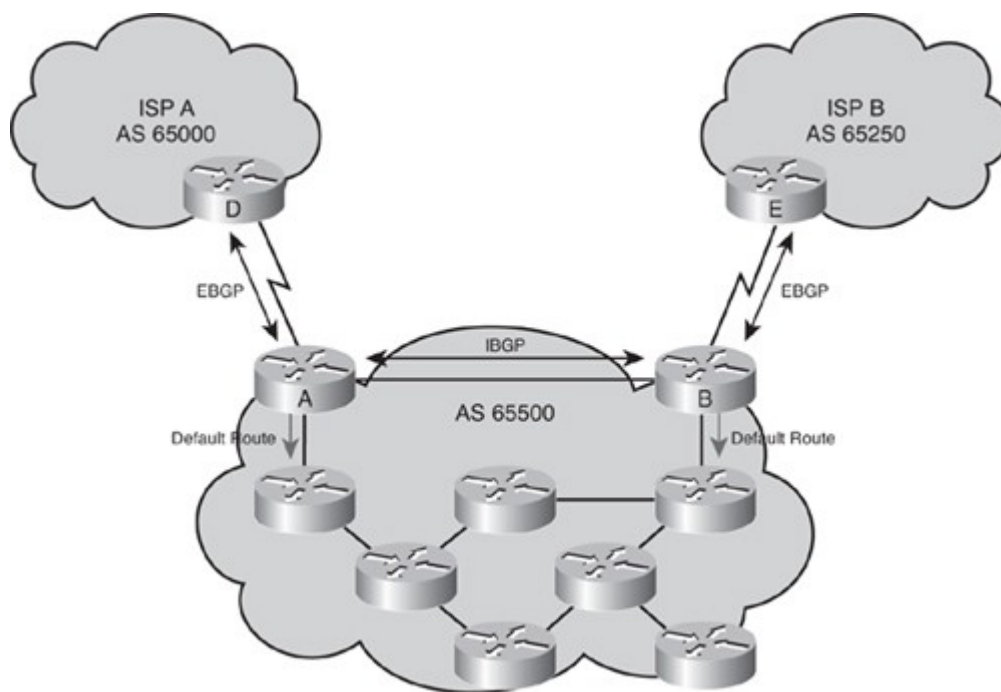
For network 192.168.24.0, the next hop of 172.20.50.2 has a lower MED than the next hop of 192.168.28.2. Therefore, for network 192.168.24.0, the path through 172.20.50.2 is the preferred path. For networks 192.168.25.0 and 192.168.26.0, the next hop of 192.168.28.2 has a lower MED (100) than the next hop of 172.20.50.2 (with a MED of 200). Therefore, 192.168.28.2 is the preferred path for those two networks.

#### Implementing BGP in an Enterprise Network

Figure 6-53 depicts a typical enterprise BGP implementation. The enterprise is multihomed to two ISPs, to increase the reliability and performance of its connection to the Internet. The ISPs might pass only default routes or might also pass other specific routes, or even all routes, to the enterprise. The enterprise routers connected to the ISPs run EBGp with the ISP routers and IBGP between themselves; therefore, all routers in the transit path within the enterprise autonomous system run IBGP. These routers pass default routes to the other routers in the enterprise, rather than redistributing BGP into the interior routing protocol.

Figure 6-53. BGP in an Enterprise.

[View full size image]



BGP attributes can be manipulated, using the methods discussed in this section, by any of the routers running BGP, to affect the path of the traffic to and from the autonomous systems.

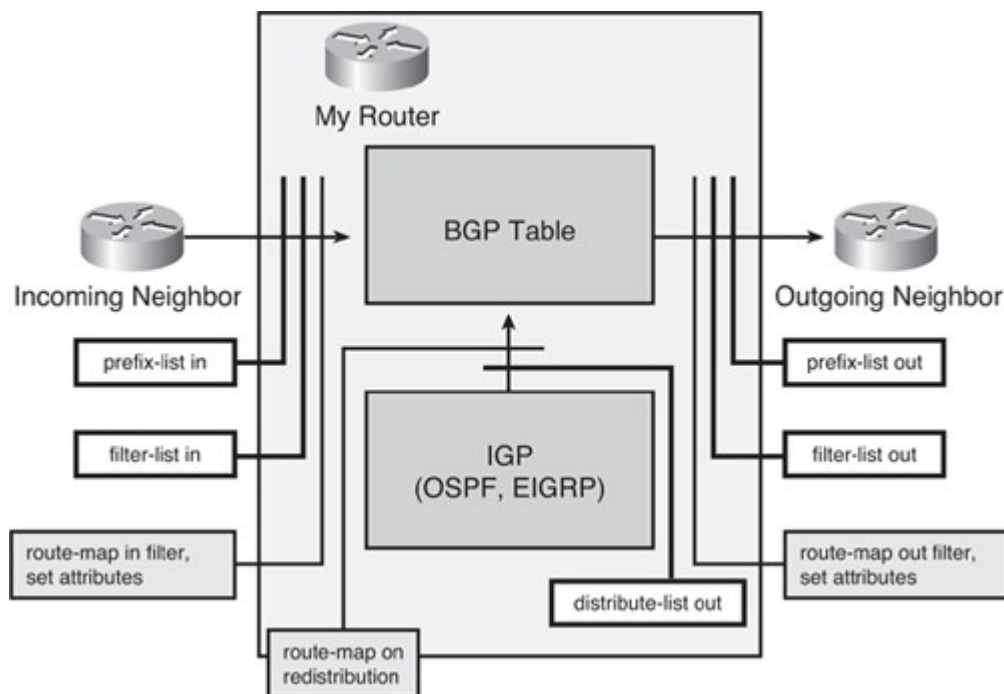
#### Filtering BGP Routing Updates

BGP may receive a high number of routing updates. To optimize BGP configuration, route filtering may be applied.

As illustrated in Figure 6-54, filter lists, prefix lists, and route maps can be applied to either incoming or outgoing BGP information, or in any combination. The incoming prefix list, filter list, and route map must all permit the routes that are received from a neighbor before they will be accepted into the BGP table. Similarly, outgoing routes must pass the outgoing filter list, prefix list, and route map before they will be sent to the neighbor.

Figure 6-54. Filtering BGP Routing Updates.

[View full size image]



If you configure a router to redistribute routing information from an IGP into BGP, the routes must successfully pass any prefix list or route map applied to the redistribution process before the route is injected into the BGP table.

#### BGP Filter Lists

Although BGP filter lists are not explored further in this book, the command is presented here for completeness.

The neighbor {ip-address | peer-group-name} filter-list access-list-number {in | out} router configuration command is used to apply a filter list to routes from or to a neighbor. The parameters of this command are described in Table 6-11.

Table 6-11. *neighbor filter-list* Command Description

| Parameter          | Description                                                                                                    |
|--------------------|----------------------------------------------------------------------------------------------------------------|
| ip-address         | IP address of the BGP neighbor.                                                                                |
| peer-group-name    | Name of a BGP peer group.                                                                                      |
| access-list-number | Number of an AS-path access list (defined with the ip as-path access-list command, as described in Table 6-9). |
| in                 | Access list is applied to incoming routes.                                                                     |
| out                | Access list is applied to outgoing routes.                                                                     |

This section describes the steps needed to configure routing update filtering using prefix lists and route maps.

#### BGP Filtering Using Prefix Lists

Chapter 4 describes prefix lists and how to configure them. This section introduces the use of prefix lists for BGP route filtering.

## Planning BGP Filtering Using Prefix Lists

When planning BGP filter configuration using prefix lists, the following steps should be documented in an implementation plan:

- Define the traffic filtering requirements, including the following:
  - Filtering updates
  - Controlling redistribution
- Configure the **ip prefix-list** statements, including using mask filtering and the **ge** and **le** parameters.
- Apply the prefix list to filter inbound or outbound updates.

### BGP Filtering Using Prefix Lists Example

Recall that the **ip prefix-list** {list-name | list-number} [seq seq-value]

{deny | permit} network/length [ge ge-value] [le le-value] global configuration command is used to create a prefix list.

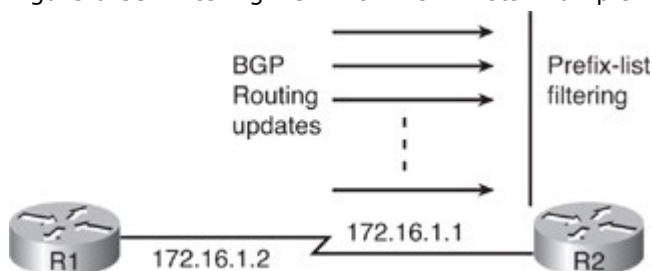
The neighbor {ip-address | peer-group-name} prefix-list prefix-list-name {in | out} router configuration command is used to apply a prefix list to routes from or to a neighbor. The parameters of this command are described in Table 6-12.

Table 6-12. *neighbor prefix-list* Command Description

| Parameter        | Description                                        |
|------------------|----------------------------------------------------|
| ip-address       | IP address of the BGP neighbor.                    |
| peer-group-name  | Name of a BGP peer group.                          |
| prefix-list-name | Name of a prefix list.                             |
| in               | Prefix list is applied to incoming advertisements. |
| out              | Prefix list is applied to outgoing advertisements. |

ip prefix-list statements in which you specify the ge and le option match any prefix within the address range that you specify with the ge and le parameters. For example, consider Figure 6-55 and the configuration on router R2 in Example 6-36. The prefix-list named ANY-8to24-NET is configured to match routes from any networks that have a mask length from 8 to 24 bits. The 0.0.0.0/0 network/length combination does not match a specific network; rather it defines any network. The parameters ge 8 and le 24 specify that any network with the mask length between 8 and 24 matches the prefix list entry.

Figure 6-55. Filtering BGP with Prefix Lists Example.



Example 6-36. Configuration for Router R2 in Figure 6-55

```
router bgp 65001
 neighbor 172.16.1.2 remote-as 65002
 neighbor 172.16.1.2 prefix-list ANY-8to24-NET in
!
ip prefix-list ANY-8to24-NET permit 0.0.0.0/0 ge 8 le 24
```

The prefix list ANY-8to24-NET is applied to the incoming advertisements from the BGP neighbor 172.16.1.2. It permits routes from any network with a mask length from 8 to 24 bits.

#### Note

Cisco IOS documentation for the **neighbor prefix-list** command says this command is used to “prevent distribution” of BGP neighbor information as specified in a prefix list. Other documentation interprets this statement incorrectly and assumes that routes permitted by the prefix list are denied (prevented) from being sent (with the **out** keyword) or received (with the **in** keyword).

Our testing confirmed that the **neighbor prefix-list** command actually behaves as we expected: routes permitted by the prefix list are sent (with the **out** keyword) or received (with the **in** keyword).

Example 6-37 displays the output from the `show ip prefix-list detail` command on router R2. Router R2 has a prefix list called ANY-8to24-NET that has only one entry (sequence number 10). The hit count of 0 means that no routes have matched this entry. Use the `clear ip prefix-list prefix-list-name[network/length]` command to reset the hit count shown on prefix list entries.

Example 6-37. `show ip prefix-list detail` Command Output on Router R2 in Figure 6-55

```
R2#show ip prefix-list detail ANY-8to24-NET
ip prefix-list ANY-8to24-NET:
Description: test-list
count: 1, range entries: 1, sequences: 10 - 10, refcount: 3
seq 10 permit 0.0.0.0/0 ge 8 le 24 (hit count: 0, refcount: 1)
```

#### BGP Filtering Using Route Maps

Route maps, as described in Chapter 4, can also be used to filter BGP updates.

##### Planning BGP Filtering Using Route Maps

When you are planning BGP filter configuration using route maps, the following steps should be documented in an implementation plan:

- Define the route map, including the following:
  - the **match** statements
  - the **set** statements
- Configure route filtering using the route map.

#### BGP Filtering with Route Maps Example

To apply a route map to filter incoming or outgoing BGP routes, use the **neighbor ip address route-map name {in | out}** router configuration command. The routes that are permitted may have their attributes set or changed, using **set** commands in the route map. This is useful when trying to influence route selection.

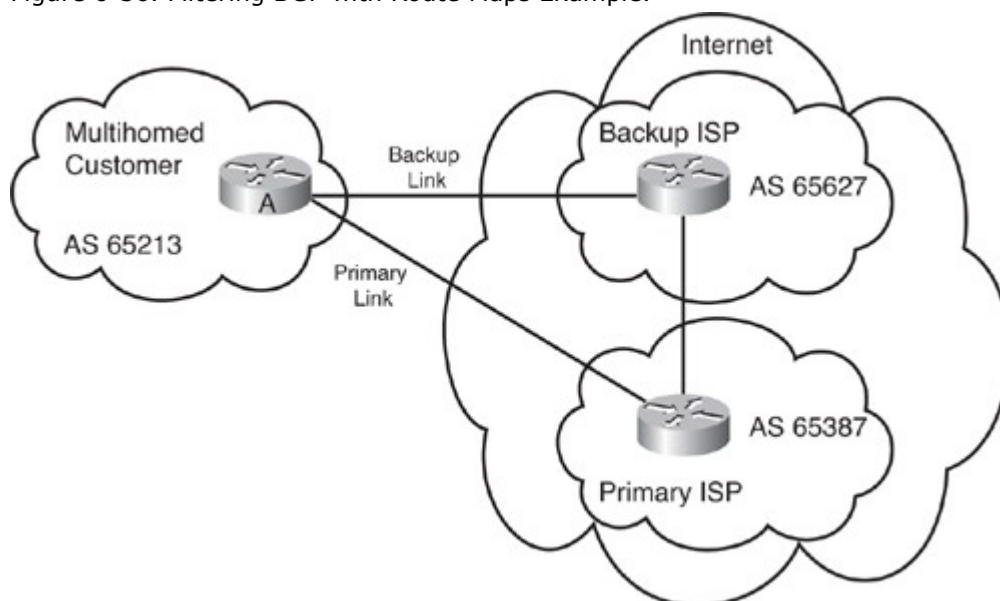
#### Note

Similar route filtering could be performed for the OSPF or EIGRP routing process. In this case, the **distribute-list in** or **distribute-list out** commands are used with route maps, which define which updates are dropped and which updates are accepted.

In the example network in Figure 6-56, the customer accepts only a default route from the two ISPs and uses the link to autonomous system 65387 as their primary link for outbound traffic. The configuration of the customer Router A is shown in Example 6-38.



Figure 6-56. Filtering BGP with Route Maps Example.



Example 6-38. Configuration of Router A in Figure 6-56

```

router bgp 65213
 neighbor 10.2.3.4 remote-as 65627
 neighbor 10.2.3.4 route-map filter in
 neighbor 10.4.5.6 remote-as 65387
 neighbor 10.4.5.6 route-map filter in
!

route-map filter permit 10
 match ip address prefix-list defonly
 match as-path 10
 set weight 150
!

route-map filter permit 20
 match ip address prefix-list defonly
 set weight 100
!

ip as-path access-list 10 permit _65387$
ip prefix-list defonly seq 10 permit 0.0.0.0/0

```

Router A is configured for BGP with two neighbors using **neighbors remote-as** commands. Both neighbors are configured with the **neighbor route-map** command to filter the incoming routing update traffic according to the route-map named *filter*. The route map *filter* allows only a default route into the customer's network, as defined by the prefix list *defonly*. The default route coming from ISP in autonomous system 65387, as defined in the autonomous system path access list 10, is assigned a weight value of 150, and all other default routes (in this case, the one coming from ISP autonomous system 65627) are assigned a weight value of 100. Because a higher weight value is preferred, the link to ISP autonomous system 65387 is preferred.

## Summary

This chapter covered the basics of BGP, the EGP used on the Internet, through discussion of the following topics:

- BGP terminology and concepts, including the following:
  - BGP's use between autonomous systems.
  - The range of private autonomous system numbers: 64512 to 65535.
  - Requirements for an enterprise's connection to an ISP, including public IP address space, link type and bandwidth, routing protocol, and connection redundancy.
  - Using Layer 2 circuit emulation and Layer 3 MPLS VPNs to connect to multiple ISPs.
  - Using static routes to connect to an ISP.
  - The four connection redundancy types:
    - Single-homed: One connection to one ISP
    - Dual-homed: Two connections to one ISP
    - Multihomed: One connection to each of multiple (usually two) ISPs
    - Dual-multihomed: Two connections to each of two ISPs
- BGP neighbor (peer) relationships:
  - IBGP, when BGP runs between routers in the same autonomous system.
  - EBGP, when BGP runs between routers that are in different autonomous systems. EBGP neighbors are typically directly connected.
- Three common multihoming options:
  - Each ISP passes only a default route to the autonomous system.
  - Each ISP passes only a default route and provider-owned specific routes to the autonomous system.
  - Each ISP passes all routes to the autonomous system.
- BGP's loop-free guarantee, because it does not accept a routing update that already includes its autonomous system number in the path list
- When to use BGP: If the autonomous system allows packets to transit through it to reach other autonomous systems, if the autonomous system has multiple connections to other autonomous systems, or if the routing policy and route selection for traffic entering and leaving the autonomous system must be manipulated.
- When not to use BGP: If there is only a single connection to the Internet or another autonomous system, if edge routers have a lack of memory or processing power, or if you have a limited understanding of route filtering and the BGP path-selection process.
- BGP's classification as a path vector protocol and its use of TCP protocol 179.
- The use of full-mesh IBGP on all routers in the transit path within the autonomous system.
- The BGP synchronization rule, which states that a BGP router should not use, or advertise to an external neighbor, a route learned by IBGP, unless that route is local or is learned from the IGP. BGP synchronization is disabled by default in Cisco IOS Software Release 12.2(8)T and later.
- The three tables used by BGP: the BGP table, IP routing table, and BGP neighbor table.
- The four BGP message types: open, keepalive, update, and notification.
- The BGP attributes that can be either well known or optional, mandatory or discretionary, and transitive or nontransitive. An attribute might also be partial. The BGP attributes are the following:
  - AS-path: Well-known mandatory. The list of autonomous system numbers that a route has traversed to reach a destination, with the number of the autonomous system that originated the route at the end of the list.
  - Next-hop: Well-known mandatory. Indicates the next-hop IP address that is to be used to reach a destination. For EBGP, the next hop is the IP address of the neighbor that sent the update; for IBGP, the protocol states that the next hop advertised by EBGP should be carried into IBGP.
  - Origin: Well-known mandatory. Defines the origin of the path information; can be IGP, EGP, or incomplete.
  - Local preference: Well-known discretionary. Indicates to routers in the autonomous system which path is preferred to exit the autonomous system. The path with a *higher* local preference is preferred. Sent only to IBGP neighbors.

- Atomic aggregate: Well-known discretionary. Informs the neighbor autonomous system that the originating router has aggregated the routes.
- Aggregator: Optional transitive. Specifies the BGP router ID and autonomous system number of the router that performed the route aggregation.
- Community: Optional transitive. Allows routers to tag routes with an indicator (the community) and allows other routers to make decisions based on that tag.
- MED: Optional nontransitive. Also called metric. Indicates to external neighbors the preferred path into an autonomous system. A *lower* value is preferred; exchanged between autonomous systems.
- Weight: Cisco defined; provides local routing policy only and is not propagated to any BGP neighbors. Routes with a *higher* weight are preferred.
- The 11-step BGP route-selection decision process:

- Step 1. Prefer the highest weight
- Step 2. Prefer the highest local preference
- Step 3. Prefer the route originated by the local router
- Step 4. Prefer the shortest AS-path
- Step 5. Prefer the lowest origin code
- Step 6. Prefer the lowest MED
- Step 7. Prefer the EBGp path over the IBGP path
- Step 8. Prefer the path through the closest IGP neighbor
- Step 9. Prefer the oldest route for EBGp paths
- Step 10. Prefer the path with the lowest neighbor BGP router ID
- Step 11. Prefer the route with the lowest neighbor IP address

- BGP peer groups, a group of BGP neighbors of the router being configured that all have the same update policies, created with the **neighbor peer-group-name peer-group** router configuration command. The **neighbor ip-address peer-group peer-group-name** router configuration command assigns neighbors as part of the group. The **clear ip bgp peer-group peer-group-name EXEC** command resets the BGP connections for all members of a BGP peer group.
- Basic BGP configuration commands:
  - The **router bgp autonomous-system** global configuration command, which defines the autonomous system that the router is in and enters router configuration mode.
  - The **neighbor {ip-address | peer-group-name} remote-as autonomous-system** router configuration command, which tells BGP *where* to advertise.
  - The **network network-number [mask network-mask] [route-map map-tag]** router configuration command, which tells BGP *what* to advertise. This command permits BGP to advertise a network if it is present in the IP routing table.
- The relationship between the BGP table, the IP routing table and the **network** command: The **network** command allows a BGP router to advertise a network that is in its IP routing table to its neighbors. The neighbor router that receives that network information puts the information in its BGP table and selects its best BGP route for that network. The best route is offered to its IP routing table. BGP neighbors exchange their best BGP routes. Routes learned by IBGP do not have to be in the IP routing table for BGP to use or advertise them (with the default of synchronization disabled).
- Using the **neighbor {ip-address | peer-group-name} shutdown** router configuration command to administratively shut down an existing BGP neighbor or peer group.
- Using the **neighbor {ip-address | peer-group-name} update-source loopback interface-number** router configuration command to cause the router to use the address of the specified loopback interface as the source address for BGP connections to this neighbor.
- Using the **neighbor {ip-address | peer-group-name} ebgp-multihop [ttl]** router configuration command to enable multihop EBGp.
- Using the **neighbor {ip-address | peer-group-name} next-hop-self** router configuration command to force BGP to use the source IP address of the update as the next hop for each network it advertises to the neighbor.

- Using the **neighbor** {*ip-address* | *peer-group-name*} **password** *string* router configuration command to enable MD5 authentication on a TCP connection between two BGP peers.
- Using the **synchronization** router configuration command to enable BGP synchronization so that a router will not advertise routes in BGP until it learns them in an IGP.
- Using the **clear ip bgp** {*\** | *neighbor-address*} privileged EXEC command to cause a hard reset of the BGP neighbors involved. Using the **clear ip bgp** {*\** | *neighbor-address*} [**soft out**] privileged EXEC command to cause BGP to do a soft reset for outbound updates.
- Using the **neighbor** {*ip-address*} **soft-reconfiguration inbound** router configuration command to inform BGP to save all updates that were learned from the neighbor specified. When an inbound policy is changed, use the **clear ip bgp** {*\** | *neighbor-address*} **soft in** privileged EXEC command to cause the router to use the stored unfiltered table to generate new inbound updates. Alternatively, use the **clear ip bgp** {*\** | *neighbor-address*} [**soft in** | **in**] privileged EXEC command for dynamic soft reconfiguration, without using the **neighbor soft-reconfiguration inbound** command, in routers with an IOS that supports this feature.
- Commands for verifying BGP configuration, including the following:
  - show ip bgp neighbors
  - **show ip bgp neighbors** {*address*} **received-routes**
  - **show ip bgp neighbors** {*address*} **routes**
  - **show ip bgp neighbors** {*address*} **advertised-routes**
  - show ip bgp
  - show ip bgp summary
  - show ip bgp rib-failure
  - debug ip bgp {*dampening* | *events* | *keepalives* | *updates*}
- Understanding and troubleshooting the BGP states: idle, connect, active, open sent, open confirm, and established.
- BGP path manipulation, including the following:
  - Using the **neighbor** {*ip-address* | *peer-group-name*} **weight** *weight* router configuration command to assign a weight to updates from a neighbor connection.
  - Using the **match as-path** *path-list-number* route-map configuration command to match AS-path attributes defined in an autonomous system path access-list. The autonomous system path access-list is defined by the **ip as-path access-list** *acl-number* {**permit** | **deny**} *regex* global configuration command.
  - Using the **set weight** *weight* route-map configuration command to change the weight within a route-map.
  - Using the **bgp default local-preference** *value* router configuration command to change the default local preference to the value specified.
  - Using the **set local-preference** *local-preference* route-map configuration command to change the local preference within a route-map.
  - Using the **as-path** {*tag* | **prepend** *as-path-string*} route-map configuration command to modify the AS-path attribute.
  - Using the **default-metric** *number* router configuration command to change the MED value for each network an autonomous system owns and advertises to an EBGp neighbor.
  - Using the **set metric** *metric* route-map configuration command to change the MED value within a route-map.
  - Using the **neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**} router configuration command to apply a filter list to routes from or to a neighbor.
  - Using the **ip prefix-list** {*list-name* | *list-number*} [**seq** *seq-value*] {**deny** | **permit**} *network/length* [**ge** *ge-value*] [**le** *le-value*] global configuration command to create a prefix list. Use the **neighbor** {*ip-address* | *peer-group-name*} **prefix-list** *prefix-list-name* {**in** | **out**} router configuration command to apply a prefix list to routes from or to a neighbor. Use the **show ip prefix-list detail** command to verify prefix lists.
  - Using the **neighbor** *ip address* **route-map** *name* {**in** | **out**} router configuration command to apply a route map to filter incoming or outgoing BGP routes.

## References

For additional information, see these resources:

- “Cisco IOS Software Releases 12.4 Mainline” support page:[http://www.cisco.com/en/US/products/ps6350/tsd\\_products\\_support\\_series\\_home.html](http://www.cisco.com/en/US/products/ps6350/tsd_products_support_series_home.html)
- The Cisco IOS Command Reference at [http://www.cisco.com/en/US/products/ps6350/prod\\_command\\_reference\\_list.html](http://www.cisco.com/en/US/products/ps6350/prod_command_reference_list.html)
- “Load Sharing with BGP in Single and Multihomed Environments—Sample Configurations” at [http://www.cisco.com/en/US/tech/tk365/technologies\\_configuration\\_example09186a00800945bf.shtml](http://www.cisco.com/en/US/tech/tk365/technologies_configuration_example09186a00800945bf.shtml)

## Review Questions

Answer the following questions, and then see Appendix A, “Answers to Review Questions,” for the answers.

1. What is the difference between an IGP and an EGP?
2. What type of routing protocol is BGP?
3. Select the true statements.
  - a. In a Layer 2 MPLS VPN, all the customer edge routers are on the same IP subnet.
  - b. In a Layer 3 MPLS VPN, all the customer edge routers are on the same IP subnet.
  - c. In a Layer 2 MPLS VPN, each customer edge router is on a different IP subnet.
  - d. In a Layer 3 MPLS VPN, each customer edge router is on a different IP subnet.
4. What is the difference between BGP dual homing and dual multihoming?
5. What are three common design options for BGP multihoming?
6. What are some advantages of getting default routes and selected specific routes from your ISPs?
7. What is a disadvantage of having all ISPs pass all BGP routes into your autonomous system?
8. A BGP router knows of three paths to a network and has chosen the best path. Can this BGP router advertise to its peer routers a route to that network other than the best path?
9. When is it appropriate to use BGP to connect to other autonomous systems?
10. When is it appropriate to use static routes rather than BGP to interconnect autonomous systems?
11. What protocol does BGP use as its transport protocol? What port number does BGP use?
12. How does BGP guarantee a loop-free autonomous system path?
13. Any two routers that have formed a BGP connection can be referred to by what two terms?
14. Write a brief definition for each of the following:
  - IBGP
  - EBGP
  - Well-known attribute
  - Transitive attribute
  - BGP synchronization
15. What tables are used by BGP?
16. What are the four BGP message types?
17. How is the BGP router ID selected?
18. What are the BGP states a router can be in with its neighbors?
19. What type of BGP attributes are the following?
  - AS-path
  - Next-hop
  - Origin
  - Local preference
  - Atomic aggregate
  - Aggregator
  - Community
  - Multi-exit-discriminator
20. When IBGP advertises a prefix that was originally learned via EBGP, by default what does it use for the next-hop attribute of the update?

21. Describe the complication that an NBMA network can cause for an update's next-hop attribute.
22. Complete the following table to answer these questions about three BGP attributes:
  - In which order are the attributes preferred (1, 2, or 3)?
  - For the attribute, is the highest or lowest value preferred?
  - Which other routers, if any, is the attribute sent to?

| Attribute        | Order Preferred In | Highest or Lowest Value Preferred? | Sent to Which Other Routers? |
|------------------|--------------------|------------------------------------|------------------------------|
| Local preference |                    |                                    |                              |
| MED              |                    |                                    |                              |
| Weight           |                    |                                    |                              |

23. Under what circumstances should BGP synchronization be disabled?
24. What does the **neighbor 10.1.1.1 ebgp-multihop** command do?
25. Which commands are used to configure Routers A and B if Router A is to run BGP in autonomous system 65000 and establish a neighbor relationship with Router B in autonomous system 65001? The two routers are directly connected but should use their loopback 0 addresses to establish the BGP connection; Router A has loopback 0 address 10.1.1.1/24, and Router B has loopback 0 address 10.2.2.2/24.
26. What command disables BGP synchronization if it is enabled?
27. Which command would Router A in autonomous system 65000 use to activate an IBGP session with Router B, 10.1.1.1, also in autonomous system 65000?
28. What is the difference between the BGP **neighbor** command and the BGP **network** command?
29. Your router's routing table has a route for 172.16.0.0/24. You configure the **network 172.16.0.0 mask 255.255.0.0** command for the BGP process running on the router. What does this command do?
30. What does the **clear ip bgp 10.1.1.1 soft out** command do?
31. Which command is used to display detailed information about BGP connections to neighbors?
32. What does a > in the output of the **show ip bgp** command mean?
33. What column in the **show ip bgp** command output displays the MED?
34. How is the *established* neighbor state represented in the output of the **show ip bgp summary** command?
35. What type of authentication does BGP support?
36. How can BGP path manipulation affect the relative bandwidth used between two connections to the Internet?
37. Describe what the following configuration on Router A does:
 

```
route-map local_pref permit 10
 match ip address 65
 set local-preference 300
route-map local_pref permit 20
router bgp 65001
 neighbor 192.168.5.3 remote-as 65002
 neighbor 192.168.5.3 route-map local_pref in
```
38. Place the BGP route-selection criteria in order from the first step to the last step evaluated by placing a number in the blank provided.
  - \_\_\_\_\_ Prefer the path with the lowest neighbor BGP router ID
  - \_\_\_\_\_ Prefer the lowest MED
  - \_\_\_\_\_ Prefer the shortest AS-path
  - \_\_\_\_\_ Prefer the oldest route for EBGp paths
  - \_\_\_\_\_ Prefer the lowest origin code
  - \_\_\_\_\_ Prefer the highest weight
  - \_\_\_\_\_ Prefer the path through the closest IGP neighbor

- \_\_\_\_\_ Prefer the highest local preference
  - \_\_\_\_\_ Prefer the route originated by the local router
  - \_\_\_\_\_ Prefer the route with the lowest neighbor IP address
  - \_\_\_\_\_ Prefer the EBGp path over the IBGP path
39. What command is used to assign a weight to updates from a BGP neighbor connection?
40. What does the **ip as-path access-list** command do?
41. Describe what AS-path prepending is.
42. What effect would the **ip prefix-list test permit 0.0.0.0/0 ge 8 le 20** command have on the 10.1.0.0/16 route and on the 10.2.0.0/24 route?

## Chapter 7. Implementing Routing Facilities for Branch Offices and Mobile Workers

This chapter discusses the implementation of routing facilities for branch offices and mobile workers. It covers the following topics:

- Planning the Branch Office Implementation
- Planning for Mobile Worker Implementations
- Routing Traffic to the Mobile Worker

This chapter discusses the challenges and unique service requirements of connecting branch sites to central sites. Specifically, the focus is on the routing requirements of branch offices and mobile workers.

### Planning the Branch Office Implementation

This section describes the fundamentals of branch office implementation such as design, security concerns, and scalability.



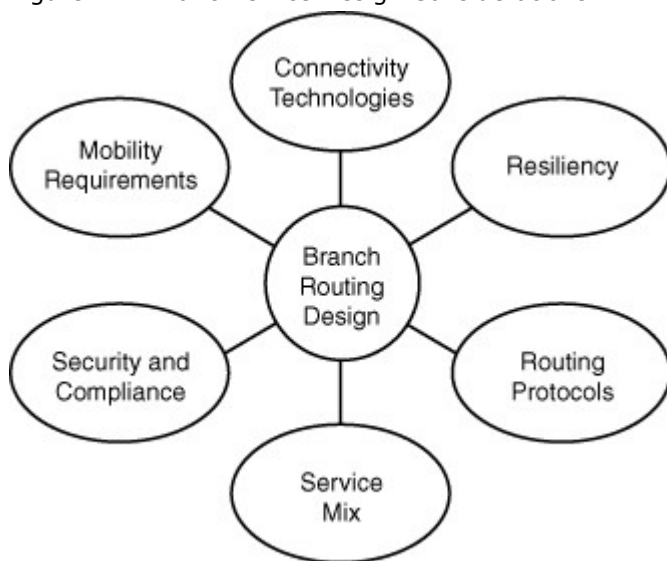
## Branch Office Design

There are common requirements that every branch network design needs to address. Connectivity, security, availability, voice, and application optimization requirements impact the branch office design and must be considered. The challenges when addressing these requirements include the following:

- **Bandwidth and network requirements**— Increasing bandwidth and network requirements are driven by a unified service network, transporting video, voice, and data, and supporting mission critical functions and applications.
- **Consolidated data centers**— Rather than operating local data servers at the branch, information is consolidated at a central local. A primary goal with this consolidation is to gain centralized security and management control, to comply with corporate governance mandates driven in part by government and industry regulations.
- **Mobility**— The dispersion of the staff coupled with the consolidation of the IT resources has resulted in a significant WAN load as distant users compete for access across the WAN. Not addressing the issue of the consolidated data center, coupled with users' mobility, has left legacy branch offices with poor application response time and therefore disgruntled users.
- **Disparate networks**— Branch offices that are built in isolation tend to run aging and separate voice and data networks, which do not benefit from the use of collaborative communications applications hosted in IP call servers. Different circuit-switched private branch exchanges (PBXs) from different vendors might exist at various branch office sites, each with its own feature set, proprietary technology, and special operational requirements.
- **Management costs**— Generally as branch office sites develop, often without much strategic thought given to future requirements, equipment and services are added to solve specific problems. The result then is a patchwork of network devices in which branch offices often have very different equipment and architectures. For this reason, branch offices are often extremely costly to manage and troubleshoot. In addition, rolling out new services across inconsistent branch office infrastructures is extremely difficult.

Meeting the preceding challenges demands a formal branch office strategy and deployment plan. For this reason, network designers must consider a multidimensional approach to branch design. If we look at these scoping and profiling considerations, they all align to a structured approach that should consider specific areas. Presuming that our objective here is to implement routing facilities for the branch, we will consider how these areas affect the routing design. The considerations are shown in Figure 7-1.

Figure 7-1. Branch Office Design Considerations.



Bandwidth and service availability are largely based on the choices of access and connectivity technologies. Technologies such as digital subscriber line (DSL) and cable usually provide dynamic IP addressing, and therefore a default route is created at the moment of connection. Virtual private networks (VPNs) are created using technologies such as IPsec. However, IPsec does not support dynamic routing protocols, and therefore complementary technologies such as generic routing encapsulation (GRE) should be considered.

Resiliency considerations will typically result in alternative paths to the central location and other branches, and this will impact your choice of routing tools. Your choice of routing protocol is important because it will

impact routing convergence, backup routes, load sharing, and so on. In your evaluation of which routing protocol is best suited for your organization, you might consider specific features of Enhanced Interior Gateway Routing Protocol (EIGRP) such as unequal-cost load balancing or stub routing. Otherwise, you might want to control convergence or scalability issues using the multi-area Open Shortest Path First (OSPF) routing protocol.

The traffic and service mix will also have an impact in routing facilities. In deploying branch services, the deployment plan should consider features such as Network Address Translation (NAT), quality of service (QoS), access control lists (ACLs), and WAN optimization.

Other considerations include security and mobility requirements. Now the branch is not only connecting to a central office for service delivery, but also serving as a connectivity hub for mobile users, in their quest for those services. Each user now becomes a new destination that has to be dealt with.

When deploying branch services, one must consider how the following trends and considerations affect the implementation plan:

- Consolidation
- Integration
- High availability
- VPNs as a WAN option
- Operational consistency

Data center and branch consolidation is a transforming trend, which results in the branch being seemingly stripped off multiple services that used to be deployed locally at the branch. The remote node is therefore called a *thin branch*. This approach has no impact on end-user productivity.

However, some of those services are now being deployed in integrated network elements, like Cisco integrated services routers (ISRs). The main result is that implementation plans need to include verification steps to identify services and their impact in the deployment and upgrade of branch routing services.

Following is a list of some services that may be deployed on branch routers:

- Voice
- Application firewall
- Intrusion prevention
- Virtual private network
- WAN optimization
- Wireless
- WAN backup

Therefore, ISRs enable businesses to reduce costs by deploying a single, resilient system for fast, secure delivery of multiple mission-critical business services, including data, voice, security, and wireless.

Note

Cisco Systems has recently introduced a new branch architecture solution called the Cisco Borderless Network Architecture. This new architecture empowers IT to efficiently manage access from multiple locations, from multiple devices, and to applications that can be located anywhere. It provides the framework to unify wired and wireless access, including security, access control and performance management across many different device types and is based on the new generation of Cisco Integrated Services Router routers (ISR G2). For more information, see <http://cisco.com/en/US/netsol/ns1015/>.

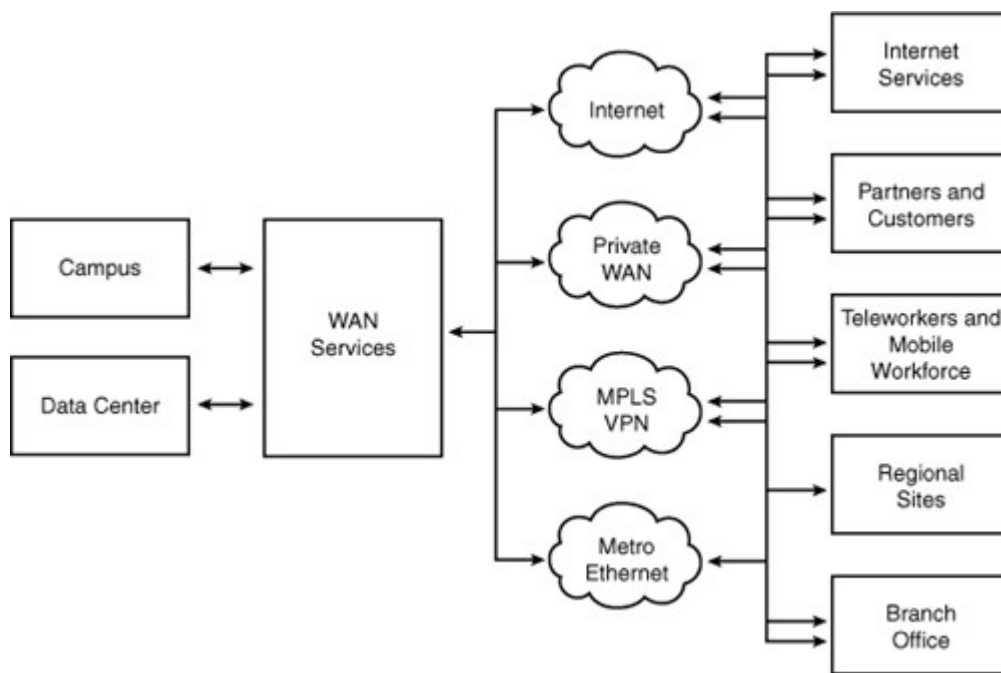
Other transforming trends include the increasing use of technologies that facilitate high availability, scalability and resiliency. Solutions such as Multiprotocol Label Switching (MPLS) VPNs and Internet VPNs, using IPsec, bring with them many benefits but also challenges and complexity in branch routing. You may now have to peer with service providers as part of your interior gateway protocol (IGP) configuration, or perhaps not be able to run an IGP across your VPN at all without configuring additional features.

Ultimately, operational consistency is a critical business requirement. The service- and business-ready branch will require a structured approach to configuration and change management, to provide management scalability and reduced costs.

When we are discussing an enterprise network, it is important to consider that most networks are built from a discreet set of interconnected, architectural elements, seen in Figure 7-2, each of which has its own requirements. A branch office, for example, may not have the same scalability requirements as a data center, but has a greater need for reduced form-factor devices with high-value integrated services.

Figure 7-2. Profiling Locations Based on Size, WAN, and Functional and Service Requirements.

[View full size image]



For the branch office in particular, it is essential to follow an established framework to ensure the best possible user experience in all locations. Because each location type has its specific needs and challenges, using a “place-in-the-network” framework is crucial to providing the best solutions for each location. So, you are more likely to see a deployment scenario where VPN is the main link for a branch office in a small branch scenario. You will also see static routes most likely. In a regional office, you would probably see a primary and backup links, with routing protocols selecting the best path. In a campus network, you will see a lot more redundancy and availability scenario with dual-edge routers and redundancy solutions such as Hot Standby Router Protocol (HSRP).

#### Upgrade Scenario

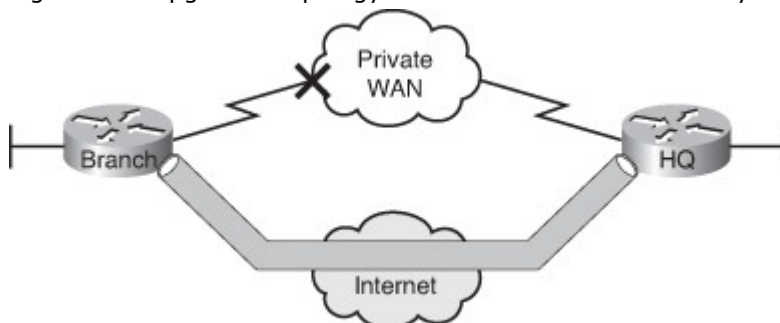
The objective in this chapter is to demonstrate some of the branch office features following a specific scenario. The scenario is that of a hub-and-spoke branch connectivity deployment. Initially, only one hub and basic services are in place. The branches reach the rest of the network via default routes injected via EIGRP through a private WAN using dedicated links. The hub device, our headquarters router, routes to the branches using EIGRP as routing protocol, and there is currently no redundancy to allow for a more resilient branch architecture. The branch site also provides basic services, including Dynamic Host Configuration Protocol (DHCP) and Network Address Translation (NAT). Figure 7-3 illustrates the original topology.

Figure 7-3. Original Topology of Branch Router Upgrade Scenario.



To illustrate the considerations and trends previously introduced, we will upgrade the branch using IPsec VPN connectivity as shown in Figure 7-4. We will also discover how services such as can impact the network design. Finally, we will test how both the WAN link and the IPsec VPN link can use the EIGRP unequal load-balancing feature.

Figure 7-4. Upgraded Topology for Branch Office Connectivity.



#### Implementation Plan

To accomplish the branch office upgrade, we will use a specific implementation plan that will include configurations at both the branch and the headquarters routers, as follows:

- Step 1. Deploy broadband connectivity
- Step 2. Configure static routing
- Step 3. Document and verify other services
- Step 4. Implement and tune the IPsec VPN
- Step 5. Configure GRE tunnels

We will now take a look at a sample configuration of a branch upgrade. Note that this implementation is not exhaustive and other solutions could also be applied. For example, many other connectivity solutions are available such as Frame Relay, Asynchronous Transfer Mode (ATM), MPLS VPNs, and dedicated leased lines and each with their own unique advantages and disadvantages. The following is to serve as a guide and as just one possible solution to routing to a branch site.

#### Note

The specifics of remote-access technologies and of IPsec configuration are beyond the scope of this book. Assume that the Internet team has configured the details of the remote connectivity, including IPsec. Our task, as members of the routing team, is to tune the router for the packets to be translated and routed properly.

#### Deploying Broadband Connectivity

Focusing on our upgrade objective, shown in Figure 7-4, let's now consider the first step of our implementation plan: broadband connectivity.

Branch offices typically use diverse applications (for example, e-mail, web-based applications, mission-critical applications, real-time collaboration, voice, video, and videoconferencing) that require high-bandwidth connections. The choice of access network technology and suitable bandwidth should be the first consideration addressed when connecting branch and small office/home office (SOHO) environments.

Broadband technologies provide always on access which can support enhanced voice and video services. It is often referred to as high-speed access to the Internet because it refers to any connection of 256 Kbps or greater. This can include many different connection options, including the following:

- Satellite broadband is offered by satellite service providers. The computer connects through Ethernet to a satellite modem that transmits radio signals to the nearest point of presence (POP) within the satellite network.
- Broadband cable access is offered by cable television service providers. The Internet signal is carried on the same coaxial cable that delivers cable television. A special cable modem separates the Internet signal from the other signals carried on the cable and provides an Ethernet connection to a host computer or LAN.
- Digital subscriber line (DSL) uses telephone lines, but unlike dialup access, DSL provides a continuous connection to the Internet. DSL uses a special high-speed modem that separates the DSL signal from the telephone signal and provides an Ethernet connection to a host computer or LAN.

#### Note

The choice of broadband connectivity is ultimately affected by what is available locally, the cost of the link, and the data and voice requirements of the business. As well, broadband access technologies may not provide the most secure connections which is why they are often combined with IPsec or SSL VPNs.

The following subsections examine these three broadband technologies.

#### Satellite Broadband Information

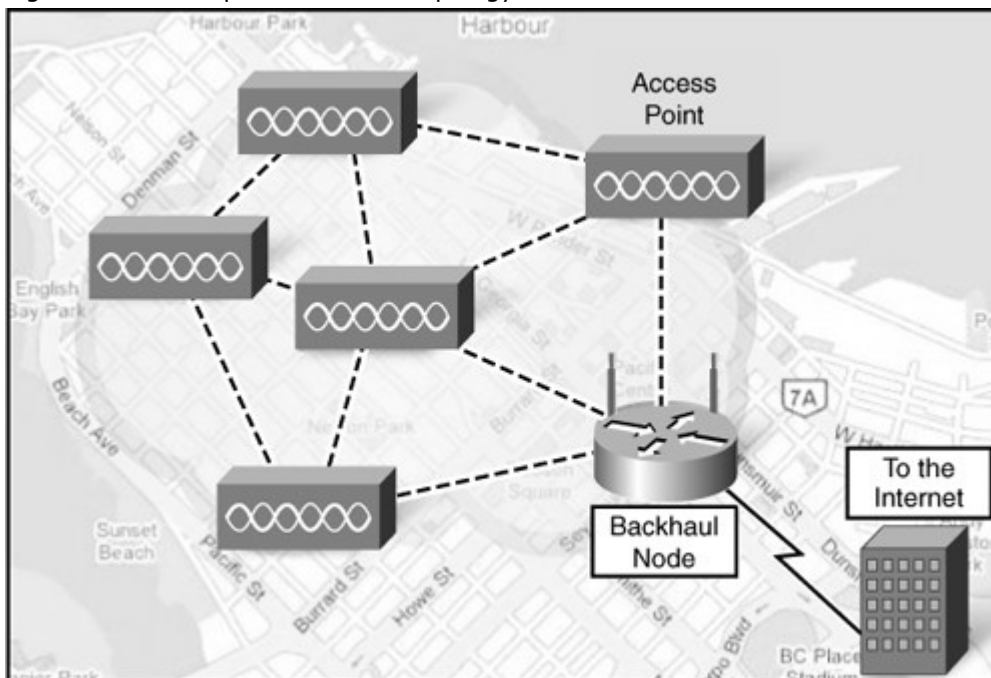
New developments in broadband wireless technology are increasing wireless availability. These new developments include the following:

- Municipal Wi-Fi
- WiMAX
- Satellite Internet

Municipal governments have also joined the Wi-Fi revolution. Often working with service providers, cities are deploying municipal wireless networks. Some of these networks provide high-speed Internet access at no cost or for substantially less than the price of other broadband services. Other cities reserve their Wi-Fi networks for official use, providing police, firefighters, and city workers remote access to the Internet and municipal networks.

Most municipal wireless networks use a mesh topology rather than a hub-and-spoke model. A mesh is a series of access points (radio transmitters), as shown in the Figure 7-5.

Figure 7-5. Municipal Wi-Fi Mesh Topology.



Each access point is in range and can communicate with at least two other access points. The mesh blankets its area with radio signals. Signals travel from access point to access point through this cloud.

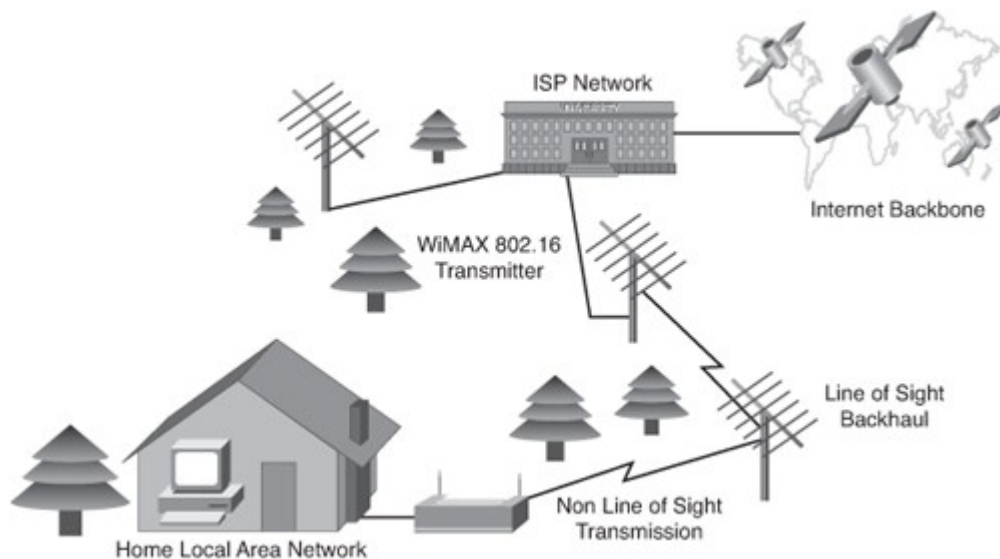
A meshed network has several advantages over single router hotspots. Installation is easier and can be less expensive because there are fewer wires. Deployment over a large urban area is faster. From an operational point of view, it is more reliable. If a node fails, others in the mesh compensate for it.

WiMAX (Worldwide Interoperability for Microwave Access) is telecommunications technology aimed at providing wireless data over long distances in a variety of ways, from point-to-point links to full mobile cellular type access. WiMAX operates at higher speeds, over greater distances, and for a greater number of users than Wi-Fi. Because of its higher speed (bandwidth) and falling component prices, it is predicted that WiMAX will soon supplant municipal mesh networks for wireless deployments.

As shown in Figure 7-6, a WiMAX network consists of two main components:

Figure 7-6. WiMAX Topology.

[View full size image]



- A tower that is similar in concept to a cellular telephone tower. A single WiMAX tower can provide coverage to an area as large as 3000 square miles, or almost 7500 square kilometers.
- A WiMAX receiver that is similar in size and shape to a PCMCIA card, or built in to a laptop or other wireless device.

A WiMAX tower station connects directly to the Internet using a high-bandwidth connection (for example, a T3 line). A tower can also connect to other WiMAX towers using line-of-sight microwave links. WiMAX is thus able to provide coverage to rural areas out of reach of "last mile" cable and DSL technologies.

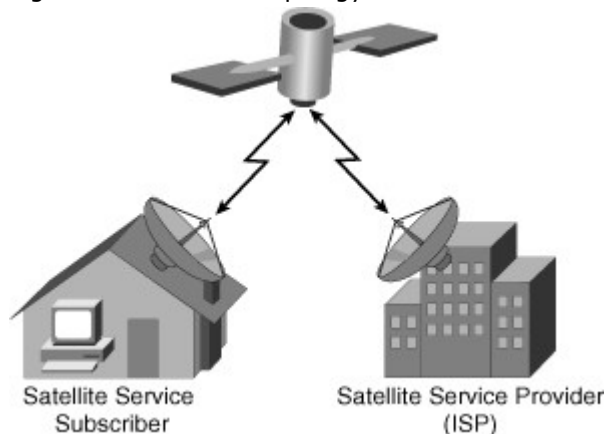
Satellite Internet services are used in locations where land-based Internet access is not available, or for temporary installations that are continually on the move. Internet access using satellites is available worldwide, including for vessels at sea, airplanes in flight, and vehicles moving on land.

There are three ways to connect to the Internet using satellites:

- One-way multicast satellite Internet systems are used for IP multicast-based data, audio, and video distribution. Even though most IP protocols require two-way communication, for Internet content, including web pages, one-way satellite-based Internet services can be information which is "pushed" to end-user sites by satellite Internet. Full interactivity is not possible.
- One-way terrestrial return satellite Internet systems use traditional dialup access to send outbound data through a modem and receive downloads from the satellite.
- Two-way satellite Internet sends data from remote sites via satellite to a hub, which then sends the data to the Internet. The satellite dish at each location needs precise positioning to avoid interference with other satellites.

Figure 7-7 illustrates a two-way satellite Internet system. Upload speeds are about one-tenth of the download speed, which is in the range of 500 kpbs.

Figure 7-7. Satellite Topology.





The key installation requirement is for the antenna to have a clear view toward the equator, where most orbiting satellites are stationed. Trees and heavy rains can affect signal reception.

Two-way satellite Internet uses IP multicasting technology, which allows one satellite to serve up to 5000 communication channels simultaneously. IP multicast sends data from one point to many points at the same time by sending data in a compressed format. Compression reduces the size of the data and the bandwidth required.

#### Cable Background Information

Accessing the Internet through a cable network is a popular option used by teleworkers to access enterprise networks. Although this solution still is not popular for connecting branch sites, it should nonetheless be considered as the technology matures.

The cable system uses a coaxial cable that carries radio frequency (RF) signals across the network. Coaxial cable is the primary medium used to build cable TV systems.

#### History of Cable Technology

Cable television first began in Pennsylvania in 1948. John Walson, the owner of an appliance store in a small mountain town, needed to solve poor over-the-air reception problems experienced by customers trying to receive TV signals from Philadelphia through the mountains. Walson erected an antenna on a utility pole on a local mountaintop that enabled him to demonstrate the televisions in his store, with strong broadcasts coming from the three Philadelphia stations. He connected the antenna to his appliance store via a cable and modified signal boosters. He then connected several of his customers who were located along the cable path. This was the first community antenna television (CATV) system in the United States.

Walson's company grew over the years, and he is recognized as the founder of the cable television industry. He was also the first cable operator to use microwave to import distant television stations, the first to use coaxial cable to improve picture quality, and the first to distribute pay television programming.

Most cable operators use satellite dishes to gather TV signals. Early systems were one way, with cascading amplifiers placed in series along the network to compensate for signal loss. These systems used taps to couple video signals from the main trunks to subscriber homes via drop cables.

Modern cable systems provide two-way communication between subscribers and the cable operator. Cable operators now offer customers advanced telecommunications services, including high-speed Internet access, digital cable television, and residential telephone service. Cable operators typically deploy hybrid fiber-coaxial (HFC) networks to enable high-speed transmission of data to cable modems located in a SOHO.

The cable TV industry uses a portion of the RF electromagnetic spectrum. Within the cable, different frequencies carry TV channels and data. At the subscriber end, equipment such as TVs, VCRs, and high-definition TV set-top boxes tune to certain frequencies that allow the user to view the channel or, using a cable modem, to receive high-speed Internet access.

A cable network is capable of sending signals on the cable in either direction at the same time. The following frequency scope is used:

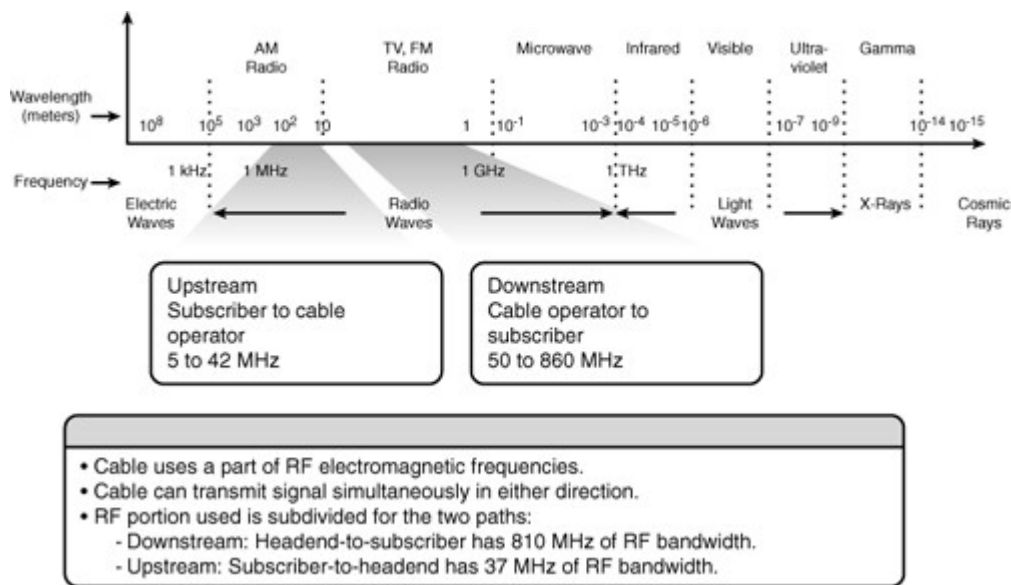
- **Downstream**— The direction of an RF signal transmission (TV channels and data) from the source (headend) to the destination (subscribers). Transmission from source to destination is called the forward path.
- **Upstream**— The direction of the RF signal transmission from subscribers to the headend, or the return or reverse path.

For example, as shown in Figure 7-8, downstream frequencies are in the range of 50 MHz to 860 MHz. Upstream frequencies are in the range of 5 MHz to 42 MHz.

Figure 7-8. Cable Transmission Frequencies.

[View full size image]



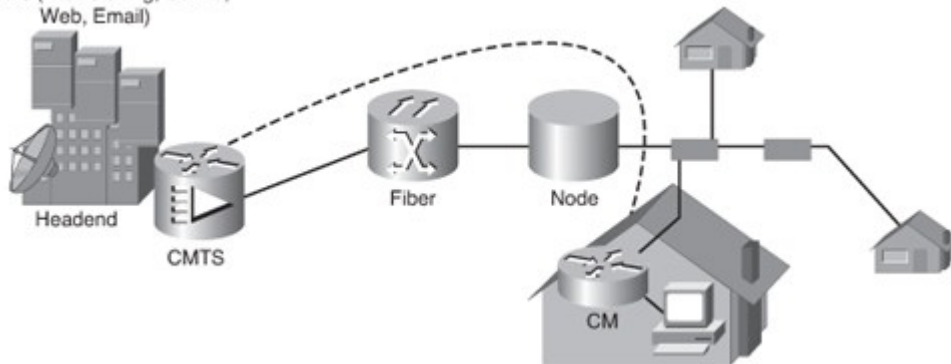


As shown in Figure 7-9, two types of equipment are required to send digital modem signals upstream and downstream on a cable system:

Figure 7-9. Cable Transmission Frequencies.

[View full size image]

Servers (Provisioning, Cache, Web, Email)



- Cable modem termination system (CMTS) at the headend of the cable operator
- Cable modem (CM) on the subscriber end

A headend CMTS communicates with CMs located in subscriber homes. The headend is actually a router with databases for providing Internet services to cable subscribers. The architecture is relatively simple, using a mixed optical-coaxial network in which optical fiber replaces the lower-bandwidth coaxial.

A web of fiber trunk cables connects the headend to the nodes where optical-to-RF signal conversion takes place. The fiber carries the same broadband content for Internet connections, telephone service, and streaming video as the coaxial cable carries. Coaxial feeder cables originate from the node that carries RF signals to the subscribers.

In a modern HFC network, typically 500 to 2000 active data subscribers are connected to a cable network segment, all sharing the upstream and downstream bandwidth. The actual bandwidth for Internet service over a CATV line can be up to 27 Mbps on the download path to the subscriber and about 2.5 Mbps of bandwidth on the upload path. Based on the cable network architecture, cable operator provisioning practices, and traffic load, an individual subscriber can typically get an access speed of between 256 kbps and 6 Mbps.

When high usage causes congestion, the cable operator can add additional bandwidth for data services by allocating an additional TV channel for high-speed data. This addition may effectively double the downstream bandwidth that is available to subscribers. Another option is to reduce the number of subscribers served by

each network segment. To reduce the number of subscribers, the cable operator further subdivides the network by laying the fiber-optic connections closer and deeper into the neighborhoods.

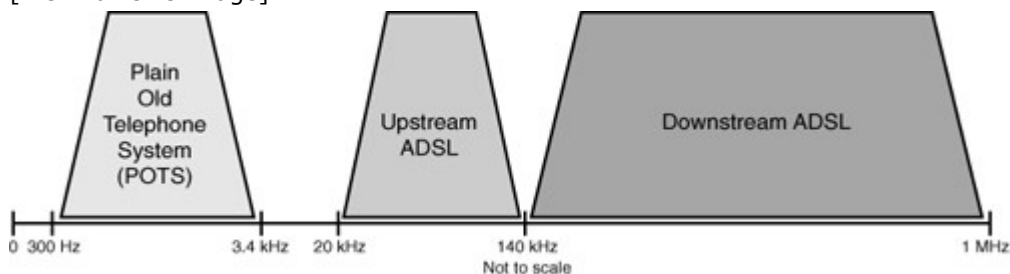
#### DSL Background Information

Although satellite broadband and cable broadband solutions are popular options for telecommuters and SOHOs, they are still not an effective option for corporate Internet access. However, DSL is one of many broadband technologies that have become efficient and effective options for corporate Internet access. For this reason, we will use DSL as our solution for the branch office connection.

Several years ago, research by Bell Labs identified that a typical voice conversation over a plain old telephone service (POTS) local loop only required the use of frequencies in the range of 300 Hz to 3400 Hz. For years, the bandwidth greater than 4 KHz went unused. Advances in technology allowed DSL to use the additional bandwidth from 4 KHz up to 1 MHz to deliver high-speed data services over ordinary copper lines. For example, as shown in Figure 7-10, asymmetric DSL (ADSL) uses a frequency range from approximately 20 KHz to 138 KHz for upstream data transmission and from 142 KHz to 1 MHz for downstream data transmission.

Figure 7-10. DSL Transmission Frequencies.

[View full size image]



What makes ADSL so attractive is that to deliver this high-bandwidth data rates to subscribers, a relatively small change to the existing telephone company infrastructure is required.

There are many variants of DSL, including ADSL, very high bitrate DSL (VDSL), symmetric digital subscriber line (SDSL), high bitrate digital subscriber line (HDSL), and single-pair high-speed digital subscriber line (SHDSL). When discussing these variants, the following properties are compared:

- **Nature**— The nature of DSL is the relation between downstream and upstream speeds. Synchronous DSL has the same speeds in both directions, whereas asynchronous DSL has different downstream and upstream speeds.
- **Maximum data rate**— Defines the maximum speed that can be deployed with a certain type of DSL.
- **Data and voice support**— Depending on the usage of the available frequency spectrum, certain DSL types support data and voice simultaneously, whereas others do not.
- **Line coding technology**— Describes the technique used to represent digital signals to be transported over a copper twisted pair so that the receiver can interpret them accurately.
- **Maximum distance**— Describes the maximum distance that a certain type of DSL connection can span from the customer premises equipment (CPE) to the DSL access multiplexer (DSLAM)

ADSL is designed to deliver more bandwidth downstream than upstream, and supports data and voice simultaneously over existing copper lines. ADSL is oriented toward residential subscribers, where more bandwidth is usually required in the downstream for applications such as downloading music, movies, playing online games, surfing the Internet, and receiving e-mail with large attachments. The downstream rate ranges from 256 kbps to 8 Mbps, while upstream speed can reach upwards of 1 Mbps.

HDSL was the first DSL technology to use a higher frequency spectrum of copper, twisted-pair cables. It could deliver T1 (1.544 Mbps) or E1 (2.048 Mbps) of symmetrical bandwidth over two copper twisted pairs. HDSL was replaced by SDSL.

SDSL is proprietary and nonstandardized technology capable of supporting T1/E1 data rates. It can carry only data and cannot coexist with a conventional voice service on the same pair (because it takes over the entire bandwidth). SDSL was mainly targeted at small and medium-size businesses that did not need the service guarantees of Frame Relay or leased line. SDSL is now considered legacy and new installations typically use SHDSL.

SHDSL is standardized and developed by the International Telecommunication Union (ITU). It is also known by the standard's draft name of G.SHDSL. It offers symmetrical data rates from 192 kbps to 2.3 Mbps and is considered a popular choice to support PBX, VPN, web hosting, and other data services.

VDSL can provide symmetrical or asymmetrical services. The downstream bandwidth ranges from 13 Mbps to 52 Mbps. Like ADSL, VDSL also supports data and voice over a single copper line. VDSL is popular in Japan, South Korea, and Germany.

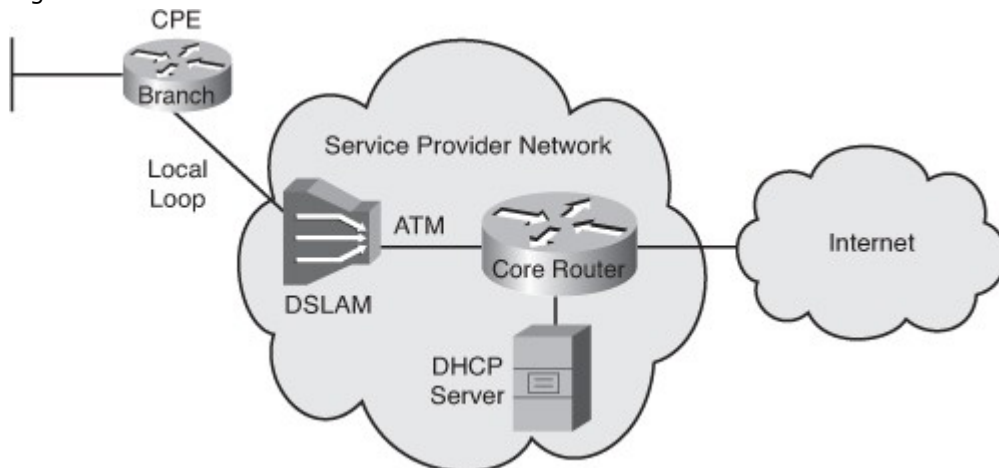
Table 7-1 summarizes the DSL variants and their characteristics.

Table 7-1. DSL Variants

| DSL Technology | Nature                 | Maximum Data Rate (Down / Up) [bps] |
|----------------|------------------------|-------------------------------------|
| ADSL           | Asymmetric             | 8 Mbps / 1 Mbps                     |
| HDSL           | Symmetric              | 2 Mbps / 2 Mbps                     |
| SDSL           | Symmetric              | 2 Mbps / 2 Mbps                     |
| SHDSL          | Symmetric              | 2.3 Mbps / 2.3 Mbps                 |
| VDSL           | Symmetric / asymmetric | 52 Mbps / 16 Mbps                   |

DSL is not a complete end-to-end solution, but rather a physical layer transmission technology similar to dial, cable, or wireless. To carry data services, DSL works in conjunction with other technologies. This results in several deployment modes. They all use a similar infrastructure shown in Figure 7-11.

Figure 7-11. DSL Infrastructures.



DSL connections are deployed in the “last mile” of a local telephone network—the local loop. The connection is set up between a pair of modems on either end of a copper wire extending between the CPE and the DSLAM. A DSLAM is the device located at the central office (CO) of the provider and concentrates connections from multiple DSL subscribers. Users typically use either Point-to-Point Protocol over ATM (PPPoA) or Point-to-Point Protocol over Ethernet (PPPoE) to connect to the providers DSLAM.

In all cases, DSL is a high-speed Layer 1 transmission technology that works over copper wires.

The DSL Layer 1 connection from the CPE is terminated at the DSLAM. The data link layer protocol that is usually used over DSL is ATM. A DSLAM is basically an ATM switch containing DSL interface cards (ATU-Cs). The DSLAM terminates the ADSL connections, and then switches the traffic over an ATM network to the service provider’s core aggregation router. The aggregation router is the Layer 3 device where IP connection from the subscriber terminates.

#### PPPoA

IP packets over an ATM and DSL connection have to be encapsulated in some way, and these three approaches exist:

- RFC 1483/2684 bridged
- PPPoE, Point-to-Point Protocol over Ethernet
- PPPoA, Point-to-Point Protocol over ATM

RFC 1483 bridging has security and scalability issues, making it unpopular as deployment architecture. PPPoE and PPPoA are more scalable and secure, but also more complex for implementation.

With PPPoA deployment option, the PPP connection is established between the CPE and the service provider core router. The CPE device must be configured with a username and password for authentication with core the router, which is responsible for terminating the PPP session of the CPE. The core router authenticates the users using either a local database or an external RADIUS authentication, authorization, accounting (AAA) server.

After the PPP username and password have been authenticated, the PPP Internet Protocol Control Protocol (IPCP) negotiation takes place to assign an IP address to the CPE. The core router will provide an IP address from its DHCP server. The core router typically assigns only one IP address to the CPE, and the CPE can use NAT and Port Address Translation (PAT) to support multiple inside hosts.

After the IP address has been assigned, a host route is established both on the CPE and the aggregation router.

#### Note

The details of PPP negotiations are beyond the scope of this book.

#### Configuring PPPoA

In our scenario, the Internet service provider has provided the branch site with a PPPoA connection to the Internet. The steps to configure PPPoA on the branch router, where components of both the DSL architecture and of basic branch IP services are required, are as follows:

1. Configure an ATM interface.
2. Configure a dialer interface.
3. Configure PAT.
4. Configure the branch router as a local DHCP server.
5. Configure a static default route.

ATM and dialer interfaces will establish the ATM virtual circuits and the PPP sessions. A dialer interface is a virtual interface that is configured as an on-demand component. This dialer interface will be up upon successful DSL subscriber authentication. Meanwhile, PAT, DHCP, and default routes provide the IP addressing and routing infrastructure to allow IP traffic to be carried along the DSL connection.

A final branch router configuration is displayed in Example 7-1.

#### Note

The PPPoA configuration is provided for reference purpose only; the actual configuration of PPPoA is beyond the scope of this chapter. You can find in-depth information about these commands at Cisco.com.

#### Example 7-1. PPPoA Sample Configuration

```
Code View: Scroll / Show All
hostname Branch
!
ip dhcp pool MY-POOL
network 192.168.1.0 255.255.255.0
default-router 192.168.1.1
!
interface ATM0/0
no ip address
dsl operating-mode auto
pvc 8/35
encapsulation aal5mux ppp dialer
dialer pool-member 1
!
interface FastEthernet0/0
ip address 192.168.1.1 255.255.255.0
ip nat inside
!
```

```

interface Dialer0
 ip address negotiated
 encapsulation ppp
 dialer pool 1
 ip nat outside
 ppp authentication chap callin
 ppp chap password mysecret
!
ip nat inside source list 101 interface Dialer0 overload
access-list 101 permit ip 192.168.1.0 0.0.0.255 any
!
ip route 0.0.0.0 0.0.0.0 Dialer0

```

The configuration in Example 7-1 assumes that the branch router has an ATM interface installed on it. Here is a high-level overview of the configuration:

- The branch router provides DHCP services to users connected to the inside LAN interface. Users connecting to the inside LAN interface would be provided with a private address from the 192.168.1.0 pool.
- The configuration specifics of the ATM 0/0 interface and the permanent virtual circuit (PVC) are provided by the DSL service provider. Notice the combination of the ATM interface **dialer pool-member 1** command and the dialer interface **dialer pool 1** commands. These two commands associate the ATM 0/0 interface to the Dialer 0 interface.
- The Dialer 0 interface is a virtual interface that initiates PPP connectivity, including PPP services such as user authentication. Notice that it is also identified as the outside NAT interface.
- NAT is configured to translate traffic initiated at the LAN port into the IP address of the dialer interface, which is obtained via DHCP from the DSL provider. In Example 7-1, you can see the **ip nat** commands, which defined the inside network with ACL 110, the outside network and how inside hosts will be translated when the traffic leaves for the outside network. Notice the **overload** keyword, which enables PAT. This means that inside users exiting through the Internet connection would share the IP address of the dialer interface. Although NAT would be more indicative of a large branch site, PAT is configured here as an example.
- Finally, notice that the static default route points to the dialer interface. The routing of traffic to this default route would trigger the dialer interface to activate.

To summarize the stages of a DSL connection: Inside traffic is routed to the dialer interface (Dialer 0). Dialer 0 being a virtual interface, which has been configured to enlist the help of any physical interfaces member of the dialer pool 1, will turn to interface ATM 0/0. Interface ATM 0/0 has been configured to bring up a DSL connection using PPP encapsulation. When the service provider core router requests a username and a password, the credentials configured under interface Dialer 0 will be presented. The core router then provides an IP address to the ATM 0/0 interface, and the Internet connection will be active. Inside users leaving through the Internet connection are provided with the IP address of the virtual interface.

#### Verifying PPPoA

To check a DSL configuration, the best approach is to use the divide-and-conquer troubleshooting model. At each layer, we will have components that help understand the status of the connection, including DSL line status, ATM PVC establishment, PPP session negotiations, and IP addresses and other components.

To confirm that the branch router has a route pointing to the dialer interface, use the **show ip route** command. Next, to check IP connectivity, use the **ping** and **traceroute** commands from an inside host to confirm proper translation and routing of the packets and that return traffic is coming back.

You can also use **debug ppp authentication** to debug the PPP session authentication. To check ATM connectivity, use **debug atm events** to see the establishment of ATM PVC. Finally, to check Layer 1 connectivity, use **show dsl interface atm** number. This command would assist in discovering the DSL line status.

Note

Again, PPP, ATM, and DSL commands are beyond the scope of this chapter. You can find more information about these commands at Cisco.com.

The list of verification commands presented here is not exhaustive. It represents a small number of useful commands that enable you to verify the configuration.

Now that basic connectivity has been established, we move on to the next step in our implementation plan.

#### Configuring Static Routing

Focusing on our upgrade objective, we will now take a look at the second step of our implementation plan: static routing.

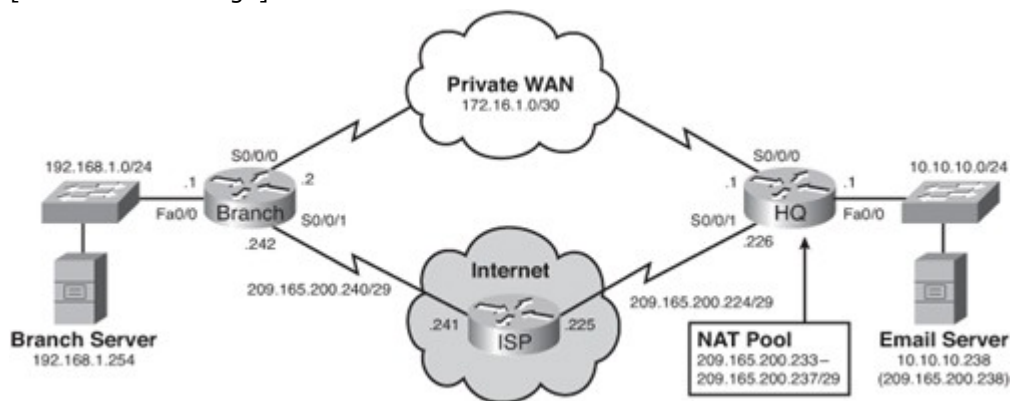
Currently, the branch office users connect to the headquarters site across a private WAN link. This link is used to provide users on the branch LAN access to servers located on the headquarters LAN. EIGRP was implemented as the dynamic routing protocol between the branch router and the HQ router. The HQ router also provides Internet access to branch LAN users by propagating a default route through EIGRP.

Although adding a new Internet connection on the branch router could also provide Internet access to the branch users, the current requirement is just to have it serve as a backup alternative in case the private WAN link fails.

Figure 7-12 will serve as our main topology to guide us through the next implementation step.

Figure 7-12. Static Route Reference Topology.

[View full size image]



The following is a summary of the topology:

- The branch router and HQ router are interconnected over a private WAN using subnet 172.16.1.0 / 30.
- The HQ LAN is on network 10.10.10.0 /24. It also has an e-mail server that the branch users access at IP address 10.10.10.238. Mobile users can also access the e-mail server from the Internet by going to the 209.165.200.238 public address. This address is translated to the internal e-mail server address using static NAT.
- The HQ router also has an Internet connection to the ISP router by exiting out of the Internet-facing interface (Serial 0/0/1). All HQ LAN and Branch LAN traffic is subject to being translated to an address in the NAT pool.
- The branch router LAN is on network 192.168.1.0 /24. It also has a server, which the HQ LAN users access at IP address 192.168.1.254.
- The Branch LAN users access the Internet by using the default route propagated by the HQ router.
- The branch router also has a new Internet connection on subnet 209.165.200.240/29. This connection will serve as a backup route for the private WAN link.

#### Note

To keep it simple, the ATM Internet link from the "Deploy Broadband Connectivity" implementation step has been replaced by the Serial 0/0/1 links on the Branch and HQ routers.



## Routing to the Internet

Our scenario dictates that all corporate traffic, from the HQ LAN to branch office LAN must go through the private WAN. Our next step is to enable the Internet link as a back to the private WAN link. To do so, a floating static route will be configured on the Branch. Let's verify the current status of the network before we configure the floating static route.

### Note

Instead of creating a floating static route, we could have used **backup** interface command to accomplish a quick turn over instead of relying on EIGRP convergence to figure there is an issue with a particular path.

Currently, the main connection to the HQ is via the private WAN network because it is configured for routing with EIGRP. You can verify this with the `show ip protocols` command issued to the branch router, as shown in Example 7-2.

Example 7-2. `show ip protocols` Output on Branch Router

```
Branch#show ip protocols
Routing Protocol is "eigrp 1"
 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 metric weight K1=1, K2=0, K3=1, K4=0, K5=0
 EIGRP maximum hopcount 100
 EIGRP maximum metric variance 1
 Redistributing: eigrp 1
 EIGRP NSF-aware route hold timer is 240s
 Automatic network summarization is not in effect
 Maximum path: 4
 Routing for Networks:
 172.16.1.0/30
 192.168.1.0
 Routing Information Sources:
 Gateway Distance Last Update
 172.16.1.1 90 00:08:19
 Distance: internal 90 external 170

Branch#
```

We see that an EIGRP process is running and that it is advertising the branch office LAN 192.168.1.0/24 toward the HQ router, using the 172.16.1.0/30 segment.

Verify the routing table on the branch router with the `show ip route` command, as shown in Example 7-3.

Example 7-3. `show ip route` Output on Branch

```
Code View: Scroll / Show All
Branch#show ip route
*Mar 26 03:45:38.207: %SYS-5-CONFIG_I: Configured from console by consolee
Codes: 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
```



Gateway of last resort is 172.16.1.1 to network 0.0.0.0

172.16.0.0/30 is subnetted, 1 subnets

C 172.16.1.0 is directly connected, Serial0/0/0

209.165.200.0/29 is subnetted, 1 subnets

C 209.165.200.240 is directly connected, Serial0/0/1

10.0.0.0/24 is subnetted, 1 subnets

D 10.10.10.0 [90/2172416] via 172.16.1.1, 00:00:17, Serial0/0/0

C 192.168.1.0/24 is directly connected, FastEthernet0/0

D\*EX 0.0.0.0/0 [170/2681856] via 172.16.1.1, 00:00:17, Serial0/0/0

Branch#

Notice that we have three directly connected networks. The branch LAN is on network 192.168.1.0/24, the private WAN link is on network 172.16.1.0/30, and the new Internet connection is on network 209.165.200.240/29. There are also two EIGRP routes denoted by the letter *D*, which stands for Diffusing Update Algorithm (DUAL), the EIGRP routing algorithm. The route to the corporate 10.10.10.0 LAN on the HQ router is provided by EIGRP. The other EIGRP route is denoted with an asterisk (\*), which means that it is candidate default route and responsible for providing the gateway of last resort to 172.16.1.1. The EX and the EIGRP administrative distance of 170 signifies that this is a route has been redistributed into EIGRP by another router.

To verify that EIGRP is also running on the HQ router, use the show ip protocols command, as shown in Example 7-4. Notice that it is advertising the 172.16.1.0/30 and 10.10.10.0 subnets.

Example 7-4. *show ip protocols* Output on the HQ Router

HQ#**show ip protocols**

Routing Protocol is "eigrp 1"

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 metric weight K1=1, K2=0, K3=1, K4=0, K5=0

EIGRP maximum hopcount 100

EIGRP maximum metric variance 1

Redistributing: static, eigrp 1

EIGRP NSF-aware route hold timer is 240s

Automatic network summarization is not in effect

Maximum path: 4

Routing for Networks:

10.10.10.0/24

172.16.1.0/30

Routing Information Sources:

| Gateway    | Distance | Last Update |
|------------|----------|-------------|
| 172.16.1.2 | 90       | 00:04:40    |

Distance: internal 90 external 170

HQ#

Notice that the HQ router is advertising a static route. To verify the router, use the show ip route command, as shown in Example 7-5.

Example 7-5. *show ip route* Output on the HQ Router

Code View: Scroll / Show All

HQ#**show ip route**

Codes: 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

Gateway of last resort is 0.0.0.0 to network 0.0.0.0

172.16.0.0/30 is subnetted, 1 subnets

C 172.16.1.0 is directly connected, Serial0/0/0

209.165.200.0/29 is subnetted, 1 subnets

C 209.165.200.224 is directly connected, Serial0/0/1

10.0.0.0/24 is subnetted, 1 subnets

C 10.10.10.0 is directly connected, FastEthernet0/0

D 192.168.1.0/24 [90/2172416] via 172.16.1.2, 00:48:28, Serial0/0/0

S\* 0.0.0.0/0 is directly connected, Serial0/0/1

HQ#

From the branch router, verify connectivity to the HQ e-mail server using the ping and trace commands to the 10.10.10.238 IP address on the HQ router, as shown in Example 7-6.

Example 7-6. Verifying Connectivity to the HQ E-Mail Server

Branch#**ping 10.10.10.238 source 192.168.1.1**

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.10.10.238, timeout is 2 seconds:

Packet sent with a source address of 192.168.1.1

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Branch#

Branch#**trace 10.10.10.238 source 192.168.1.1**

Type escape sequence to abort.

Tracing the route to 10.10.10.238

1 172.16.1.1 0 msec 0 msec \*

Branch#

Notice that both the ping and trace were successful. Also notice the path that the branch router took was the private WAN link to IP address 172.16.1.1. Finally, test connectivity to the Internet. Ping the ISP's website at IP address 209.165.202.111 and source the packets from the inside LAN, as shown in Example 7-7.

#### Example 7-7. Verifying Connectivity to the ISP Website

```
Branch#ping 209.165.202.211 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 209.165.202.211, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms
Branch#
Branch# trace 209.165.202.211 source 192.168.1.1

Type escape sequence to abort.
Tracing the route to 209.165.202.211

 1 172.16.1.1 0 msec 0 msec 0 msec
 2 209.165.200.225 16 msec 16 msec *
```

The pings are successful, and the output of the **trace** command confirms that the branch LAN is reaching the Internet via the private WAN link (172.16.1.1) and then the ISP router (209.165.200.225).

We are now ready to configure the backup connection.

#### Floating Static Route

What happens if the private WAN link fails? Traffic to the HQ e-mail server or to the Internet would not be possible. However, by adding floating default static route to the branch router, we can accomplish resiliency dynamically. Whenever the link through the private WAN link fails, the floating would populate the routing table. When the private WAN reactivates, EIGRP would reroute traffic through the private WAN.

A floating static is simply a static route with an AD greater than that of the existing dynamic routing protocol. To create a floating static route, use the **ip route prefix mask next-hop-ip-address distance** command. The default AD for a static route is 1.

We have already seen in Example 7-3 that the current AD of the default route is 170. That route is learned via EIGRP and uses the private WAN back to the HQ router. Therefore, for a floating static route to work, we need to use a distance greater than 170. Create a floating default static route to the ISP as configured in Example 7-8.

#### Example 7-8. Configuring a Floating Default Static Route on Branch

```
Branch(config)#ip route 0.0.0.0 0.0.0.0 209.165.200.241
171
Branch(config)#exit
```

Notice that the default static route has been configured with an AD of 171, making it a floating static route. We can only assume that the floating static route is waiting for the private WAN link to fail.

To test the backup feature, we will first issue the debug ip routing command on the branch router to observe events and then disable the interface to the private WAN, as shown in Example 7-9.

#### Example 7-9. Observe the Floating Static Route

```
Code View: Scroll / Show All
Branch#debug ip routing
IP routing debugging is on
Branch#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Branch(config)#int s0/0/0
Branch(config-if)#shutdown
Branch(config-if)#
```

```

*Mar 26 06:22:23.759: RT: is_up: Serial0/0/0 0 state: 6 sub state: 1 line: 0 has_route: True
*Mar 26 06:22:23.759: RT: interface Serial0/0/0 removed from routing table
*Mar 26 06:22:23.759: RT: del 172.16.1.0/30 via 0.0.0.0, connected metric [0/0]
*Mar 26 06:22:23.759: RT: delete subnet route to 172.16.1.0/30
*Mar 26 06:22:23.759: RT: NET-RED 172.16.1.0/30
*Mar 26 06:22:23.759: RT: delete network route to 172.16.0.0
*Mar 26 06:22:23.759: RT: NET-RED 172.16.0.0/16
*Mar 26 06:22:23.759: RT: Pruning routes for Serial0/0/0 (3)
*Mar 26 06:22:23.763: RT: delete route to 10.10.10.0 via 172.16.1.1, Serial0/0/0
*Mar 26 06:22:23.763: RT: no routes to 10.10.10.0, flushing
*Mar 26 06:22:23.763: RT: NET-RED 10.10.10.0/24
*Mar 26 06:22:23.767: RT: delete network route to 10.0.0.0
*Mar 26 06:22:23.767: RT: NET-RED 10.0.0.0/8
*Mar 26 06:22:23.767: RT: delete route to 0.0.0.0 via 172.16.1.1, Serial0/0/0
*Mar 26 06:22:23.767: RT: no routes to 0.0.0.0, flushing
*Mar 26 06:22:23.767: RT: NET-RED 0.0.0.0/0
*Mar 26 06:22:23.771: RT: add 0.0.0.0/0 via 209.165.200.241, static metric [171/0]
*Mar 26 06:22:23.771: RT: NET-RED 0.0.0.0/0
*Mar 26 06:22:23.771: RT: default path is now 0.0.0.0 via 209.165.200.241
*Mar 26 06:22:23.771: RT: new default network 0.0.0.0
*Mar 26 06:22:23.771: RT: NET-RED 0.0.0.0/0
*Mar 26 06:22:23.771: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.16.1.1
(Serial0/0/0) is down: interface down
Branch(config-if)#end
Branch#undebug all
All possible debugging has been turned off
Branch#

```

#### Note

**debug** commands produce verbose outputs, which in a production environment could slow down your router. Use with care. Use the **no debug all** or **undebug all** command to turn off debugging on a router.

Notice that the default route to 172.16.1.1 has been removed but the new floating default static route has been added. If we verify the routing table of the branch router, we will see the static route pointing to the HQ LAN, as shown in Example 7-10.

#### Example 7-10. Verify the Routing Table

Code View: Scroll / Show All

Branch#**show ip route**

Codes: 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

Gateway of last resort is 209.165.200.241 to network 0.0.0.0

```
209.165.200.0/29 is subnetted, 1 subnets
C 209.165.200.240 is directly connected, Serial0/0/1
 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, FastEthernet0/0
S* 0.0.0.0/0 [171/0] via 209.165.200.241
Branch#
```

A trace from the branch LAN to the HQ e-mail server global IP address of 209.165.200.238 confirms that the path taken is now through the Internet connection, as shown in Example 7-11.

Example 7-11. Trace from the Branch LAN to the E-Mail Server

```
Branch#ping 209.165.200.238 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 209.165.200.238, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
Branch#
Branch#trace 209.165.200.238 source 192.168.1.1

Type escape sequence to abort.
Tracing the route to 209.165.200.238

 1 209.165.200.241 12 msec 12 msec 16 msec
 2 209.165.200.238 28 msec 28 msec *
Branch#
```

It would appear that all is working as expected. However, the scenario as presented so far would really not be feasible, because the private addresses of the branch LAN would be filtered by the ISP router. Therefore, on the branch router, the internal private IP addresses must be translated via NAT to global public IP addresses.

#### Verifying Branch Services

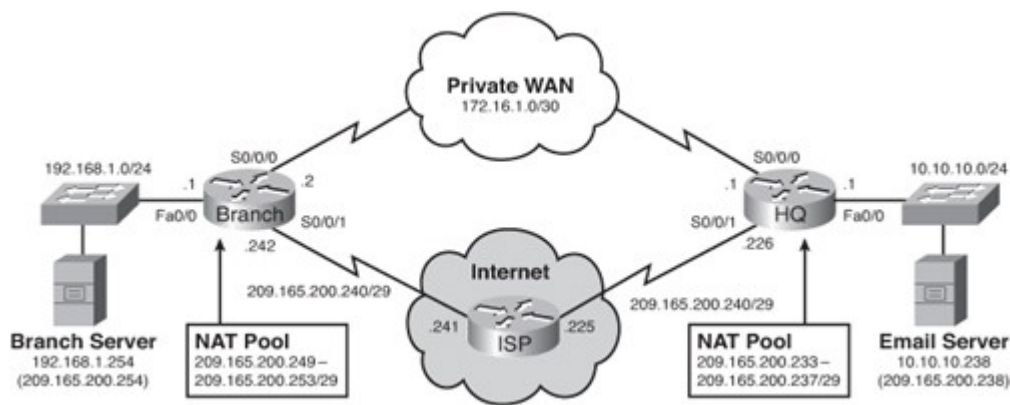
Focusing on our upgrade objective, we will now discuss the third step of our implementation plan, verify branch services, including the following:

- NAT
- DHCP
- Access lists
- Policy-based routing
- HSRP

We will specifically focus on NAT. Figure 7-13 will serve as our main topology to guide us through the next implementation step.

Figure 7-13. NAT Reference Topology.

[View full size image]



The reference topology is the same as in the previous step except that it now displays the NAT pool of global IP addresses available on the branch router. Also notice that the Branch server has a static NAT global address (209.165.200.254).

The branch router must be configured to deploy NAT as described.

#### Configuring NAT

There are three generic steps to configuring NAT. You need to identify

- Which traffic will be translated
- To what address will it be translated
- Which interfaces are involved in the translation selection

The first step in configuring NAT is to create an access control list (ACL) that will declare which traffic will be translated.

It is important to understand that the access list is not used to filter the traffic. Instead, it is used to designate which traffic will be translated by NAT. A **permit** statement in a NAT access list means "translate," and a **deny** statement in the same access list means "do not translate."

After we have declared which traffic will be translated, we need to define to which IP address the traffic will be translated. We will use the `ip nat poolname start-ip end-ip {netmask netmask | prefix-length prefix-length}` command to define a pool of IP addresses for NAT, as shown in Table 7-2.

Table 7-2. *ip nat pool* Global Command

| Parameter                                 | Description                                                                                                                                                                                                |
|-------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| name                                      | IP route prefix for the destination.                                                                                                                                                                       |
| start-ip                                  | Starting IP address that defines the range of addresses in the address pool.                                                                                                                               |
| end-ip                                    | Ending IP address that defines the range of addresses in the address pool.                                                                                                                                 |
| <b>netmask</b> <i>netmask</i>             | Network mask that indicates which address bits belong to the network and subnetwork fields and which bits belong to the host field. Specify the netmask of the network to which the pool addresses belong. |
| <b>prefix-length</b> <i>prefix-length</i> | Number that indicates how many bits of the netmask are 1s (how many bits of the address indicate network). Specify the netmask of the network to which the pool addresses belong.                          |
| type rotary                               | (Optional) Indicates that the range of address in the address pool identify real, inside hosts among which TCP load distribution will occur.                                                               |

We now need to link the source IP address to the pool for translated address. This is accomplished with the `ip nat inside source` command. Table 7-3 shows the details of options for the `ip nat inside source {list {access-`

list-number | access-list-name} | route-map name} {interface type number | pool name} [overload]  
command. If static translation is preferred, the command is ip nat inside source {static {local-ip global-ip}.

Table 7-3. *ip nat inside source* Global Command

| Parameter                             | Description                                                                                                                                                                                                                                                                 |
|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>list</b> <i>access-list-number</i> | Number of a standard IP access list. Packets with source addresses that pass the access list are dynamically translated using global addresses from the named pool.                                                                                                         |
| <b>list</b> <i>access-list-name</i>   | Name of a standard IP access list. Packets with source addresses that pass the access list are dynamically translated using global addresses from the named pool.                                                                                                           |
| <b>route-map</b> <i>name</i>          | Specifies the named route map.                                                                                                                                                                                                                                              |
| <b>interface</b> <i>type number</i>   | Specifies the interface type and number for the global address.                                                                                                                                                                                                             |
| <b>pool</b> <i>name</i>               | Name of the pool from which global IP addresses are allocated dynamically.                                                                                                                                                                                                  |
| overload                              | (Optional) Enables the router to use one global address for many local addresses. When overloading is configured, the TCP or User Datagram Protocol (UDP) port number of each inside host distinguishes between the multiple conversations using the same local IP address. |
| <b>static</b> <i>local-ip</i>         | Sets up a single static translation. The <i>local-ip</i> argument establishes the local IP address assigned to a host on the inside network.                                                                                                                                |
| local-port                            | Sets the local TCP/UDP port in a range from 1 to 65535.                                                                                                                                                                                                                     |
| <b>static</b> <i>global-ip</i>        | Sets up a single static translation. The <i>local-ip</i> argument establishes the globally unique IP address of an inside host as it appears to the outside world.                                                                                                          |
| global-port                           | Sets the global TCP/UDP port in a range from 1 to 65535.                                                                                                                                                                                                                    |
| extendable                            | (Optional) Extends the translation.                                                                                                                                                                                                                                         |
| no-alias                              | (Optional) Prohibits an alias from being created for the global address.                                                                                                                                                                                                    |
| no-payload                            | (Optional) Prohibits the translation of an embedded address or port in the payload.                                                                                                                                                                                         |
| <b>redundancy</b> <i>group-name</i>   | (Optional) Establishes NAT redundancy.                                                                                                                                                                                                                                      |
| <b>esp</b> <i>ip-local</i>            | Establishes IPsec-ESP (tunnel mode) support.                                                                                                                                                                                                                                |

The final step is to designate that traffic originating from or destined for a specific interface is subject to be translated. To do so, use the ip nat interface configuration command. Some details of the ip nat command are shown in Table 7-4.

Table 7-4. *ip nat* Interface Command

| Parameter | Description                                                                                               |
|-----------|-----------------------------------------------------------------------------------------------------------|
| inside    | Indicates that the interface is connected to the inside network (the network subject to NAT translation). |
| outside   | Indicates that the interface is connected to the outside network.                                         |



#### Note

Not all NAT command parameters were included or discussed. For more information, see Cisco.com.

#### Example of NAT Configuration

Our first step is to configure NAT on the branch router depicted in Figure 7-13. To do so, we will define an access list identifying which traffic is allowed to use NAT.

We are going to translate addresses coming from the branch LAN, regardless of destination. This is accomplished with the access list depicted in Example 7-12. Traffic with source IP address 192.168.1.0/24 is targeted for translation by the permit statement. The unseen implicit deny statement will not translate any other addresses.

#### Example 7-12. Access List Used for NAT

```
Branch(config)#ip access-list extended BRANCH-NAT-ACL
Branch(config-ext-nacl)#permit ip 192.168.1.0 0.0.0.255 any
Branch(config-ext-nacl)#exit
Branch(config)#
```

Next, the NAT pool of public IP address is defined using the ip nat pool command. In Example 7-13, the NAT pool is named BRANCH-NAT-POOL and identifies a range of valid and available Internet IP address. In this case, the public IP addresses are in the range from 209.165.201.1 to 209.165.201.29 /27.

#### Example 7-13. NAT Pool Is Created

```
Code View: Scroll / Show All
Branch(config)#ip nat pool BRANCH-NAT-POOL 209.165.200.249 209.165.200.253
netmask
255.255.255.248
Branch(config)#
Branch(config)#! Or use the prefix-length keyword
Branch(config)#
Branch(config)#ip nat pool BRANCH-NAT-POOL 209.165.200.249 209.165.200.253
prefix-
length 29
Branch(config)#
```

Notice that the subnet mask of the public IP addresses can be specified using either the **prefix-length** or **netmask** keywords.

The BRANCH-NAT-ACL and BRANCH-NAT-POOL will now be bound together with the ip nat inside source command shown in Example 7-14.

#### Example 7-14. ip nat inside source Command for Dynamic NAT on the Branch Router

```
Code View: Scroll / Show All
Branch(config)#ip nat inside source list BRANCH-NAT-ACL pool BRANCH-
NAT-POOL
Branch(config)#
```

The **ip nat inside** command specifies that the router will translate all inside IP addresses identified by the **source list BRANCH-NAT-ACL** keywords to the publicly available IP addresses identified by the **pool BRANCH-NAT-POOL** keywords.

#### Note

Optionally, PAT could have been configured by appending the keyword **overload** to the **ip nat inside** command. PAT enables a router to use only one global IP address to be shared by many inside addresses. PAT is commonly deployed in smaller branch and SOHO sites and is discussed in the “Mobile Worker” section of this chapter.

The branch site also has a web server that requires a static global IP address assigned to its internal IP address. Specifically, the internal IP address of the web server is 192.168.1.254, and it should always be translated to the global IP address 209.165.200.254.

Note

In a normal environment, the web server would not be located on the inside network, as shown in Figure 7-8. That particular server-host would be located on a separate segment, usually called the demilitarized zone (DMZ).

Example 7-15 creates a static translation entry in the router, where the inside local address 192.168.1.254 is always translated to the global 209.165.200.254 on the outside.

Example 7-15. *ip nat inside source static* Command for Static NAT on the Branch Router

```
Branch(config)#ip nat inside source static 192.168.1.254
209.165.200.254
```

```
Branch(config)#
```

The final step is to configure the interfaces involved in this particular NAT translation, as explained in Table 7-4. We therefore proceed on the branch router to issue the *ip nat outside* command on interface Serial 0/0/1 and the *ip nat inside* command on the interface Fast Ethernet 0/0, as shown in Example 7-16.

Example 7-16. *ip nat inside* and *ip nat outside* Commands Issued on the Branch Router

```
Branch(config)#interface serial 0/0/1
Branch(config-if)#ip nat outside
Branch(config-if)#
Branch(config-if)#interface fastethernet 0/0
Branch(config-if)#ip nat inside
Branch(config-if)#
```

Verifying NAT

To display active NAT translations, use the *show ip nat translations* command in EXEC mode. The details of command are shown in Table 7-5.

Table 7-5. *show ip nat translation* Command

| Parameter                  | Description                                                                                                                        |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| esp                        | (Optional) Displays Encapsulating Security Payload (ESP) entries                                                                   |
| icmp                       | (Optional) Displays Internet Control Message Protocol (ICMP) entries                                                               |
| pptp                       | (Optional) Displays Point-to-Point Tunneling Protocol (PPTP) entries                                                               |
| tcp                        | (Optional) Displays TCP protocol entries                                                                                           |
| udp                        | (Optional) Displays UDP entries                                                                                                    |
| verbose                    | (Optional) Displays additional information for each translation table entry, including how long ago the entry was created and used |
| <b>vrf</b> <i>vrf-name</i> | (Optional) Displays VPN routing and forwarding (VRF) traffic-related information                                                   |

You can also use the **show ip nat statistics** to display NAT statistics. This command has no arguments or keywords.

The clear ip nat translation can be used to clear dynamic NAT translations from the translation table. The details for the clear ip nat translation { \* | [inside global-ip global-port local-ip local-port] | [outside local-ip global-ip] [esp | tcp | udp] } are shown in Table 7-6.

Table 7-6. *clear ip nat translation* Command

| Parameter   | Description                                                                                                       |
|-------------|-------------------------------------------------------------------------------------------------------------------|
| *           | Clears all dynamic translations                                                                                   |
| inside      | (Optional) Clears the inside translations containing the specified <i>global-ip</i> and <i>local-ip</i> addresses |
| global-ip   | (Optional) Global IP address                                                                                      |
| global-port | (Optional) Global port                                                                                            |
| local-ip    | (Optional) Local IP address                                                                                       |
| local-port  | (Optional) Local port                                                                                             |
| outside     | (Optional) Clears the outside translations containing the specified <i>global</i> and <i>local</i> addresses.     |
| esp         | (Optional) Clears ESP entries from the translation table                                                          |
| tcp         | (Optional) Clears the TCP entries from the translation table                                                      |
| udp         | (Optional) Clears the UDP entries from the translation table                                                      |

#### Note

Even though the **show** command has the word *translations* in plural, as in **show ip nat translations**, the **clear** command uses the singular form in **clear ip nat translation**.

You can also use the **clear ip nat statistics** to reset the statistic counters to zero.

#### Example of NAT Verification

The show ip nat translations command displays the current NAT translations, as shown in Example 7-17.

Example 7-17. *show ip nat translations* Command

```
Branch#show ip nat translations
Pro Inside global Inside local Outside local Outside global
-- 209.165.200.254 192.168.1.254 -- --
Branch#
```

Other than the static translation to the inside web server, there are no dynamic translations listed in the NAT cache.

Another good verification command is show ip nat statistics, as shown in Example 7-18.

Example 7-18. *show ip nat statistics* Command on the Branch Router

```
Branch#show ip nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
Peak translations: 1, occurred 00:31:21 ago
Outside interfaces:
 Serial0/0/1
Inside interfaces:
 FastEthernet0/0
Hits: 0 Misses: 0
```

```
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
— Inside Source
[Id: 1] access-list BRANCH-NAT-ACL pool BRANCH-NAT-POOL refcount 0
pool BRANCH-NAT-POOL: netmask 255.255.255.224
Appl doors: 0
Normal doors: 0
Queued Packets: 0
Branch#
```

This command displays the number of active translations, which in this case is one static and zero dynamic translation. It also lists the interfaces involved in the NAT translations and the specifics of the BRANCH-NAT-POOL in use, including the BRANCH-NAT-ACL access list used for the traffic to be translated.

To test the new configuration, we will attempt to telnet to the public IP address of the HQ router, while sourcing it from the inside network address 192.168.1.1. However, before doing so, we will clear the NAT statistics, and enable the debug ip nat command to observe NAT translations, as shown in Example 7-19.

Example 7-19. Generating NAT Traffic

```
Code View: Scroll / Show All
Branch#debug ip nat
IP NAT debugging is on
Branch#clear ip nat statistics
Branch#clear ip nat translation *
Branch#telnet 209.165.200.226 /source-interface fa0/0
Trying 209.165.200.226 ... Open

Password required, but none set
*Mar 26 14:20:10.563: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10933]
Mar 26 14:20:10.591: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60321]
*Mar 26 14:20:10.595: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10934]
*Mar 26 14:20:10.595: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10935]
*Mar 26 14:20:10.595: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10936]
Mar 26 14:20:10.627: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60322]
*Mar 26 14:20:10.627: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10937]
*Mar 26 14:20:10.627: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10938]
*Mar 26 14:20:10.631: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10939]
Mar 26 14:20:10.639: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60323]
Mar 26 14:20:10.827: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60324]
*Mar 26 14:20:10.839: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10940]
[Connection to 209.165.200.226 closed by foreign host]
```

Branch#

```
Mar 26 14:20:12.723: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60325]
```

```
*Mar 26 14:20:12.723: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10941]
```

```
*Mar 26 14:20:12.727: NAT: s=192.168.1.1->209.165.200.249, d=209.165.200.226
[10942]
```

```
Mar 26 14:20:12.759: NAT: s=209.165.200.226, d=209.165.200.249->192.168.1.1
[60326]
```

Branch#

Although the Telnet to the HQ router failed, it did manage to generate some **debug** output. The first translation is showing that the internal IP address 192.168.1.1 was translated to 209.165.200.249 and sent to the IP address of the HQ router. The next line, with **NAT\***, indicates the return TCP traffic.

Example 7-20 displays the output of the show ip nat translations and show ip nat statistics commands.

Example 7-20. Verifying the NAT Translations

Code View: Scroll / Show All

Branch#**show ip nat translations**

| Pro | Inside global         | Inside local      | Outside local      | Outside global     |
|-----|-----------------------|-------------------|--------------------|--------------------|
| tcp | 209.165.200.249:55041 | 192.168.1.1:55041 | 209.165.200.226:23 | 209.165.200.226:23 |
| --  | 209.165.200.249       | 192.168.1.1       | --                 | --                 |

|    |                 |               |    |    |
|----|-----------------|---------------|----|----|
| -- | 209.165.200.254 | 192.168.1.254 | -- | -- |
|----|-----------------|---------------|----|----|

Branch#

Branch#**show ip nat statistics**

Total active translations: 3 (1 static, 2 dynamic; 1 extended)

Peak translations: 3, occurred 00:13:14 ago

Outside interfaces:

Serial0/0/1

Inside interfaces:

FastEthernet0/0

Hits: 32 Misses: 0

CEF Translated packets: 12, CEF Punted packets: 2

Expired translations: 1

Dynamic mappings:

— Inside Source

[Id: 1] access-list BRANCH-NAT-ACL pool BRANCH-NAT-POOL refcount 2

pool BRANCH-NAT-POOL: netmask 255.255.255.248

Appl doors: 0

Normal doors: 0

Queued Packets: 0

Branch#

The commands are indicating the results of the dynamic translations. The first highlighted entry also includes the source and destination TCP port numbers.

The next step is to test the static translation. To do so, issue a ping command from the HQ router to the static NAT address of 209.165.200.254, as shown in Example 7-21.

Example 7-21. Verifying the Static NAT Translation

```
HQ#ping 209.165.200.254
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 209.165.200.254, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
HQ#
```

Immediately on the branch router, the debug command again generates NAT output, as shown in Example 7-22.

Example 7-22. Verifying the NAT Translations

```
Code View: Scroll / Show All
Branch#
Mar 26 14:46:49.423: NAT: s=209.165.200.226, d=209.165.200.254->192.168.1.254
[10]
*Mar 26 14:46:49.427: NAT: s=192.168.1.254->209.165.200.254, d=209.165.200.226
[10]
Mar 26 14:46:49.483: NAT: s=209.165.200.226, d=209.165.200.254->192.168.1.254
[11]
*Mar 26 14:46:49.483: NAT: s=192.168.1.254->209.165.200.254, d=209.165.200.226
[11]
Mar 26 14:46:49.539: NAT: s=209.165.200.226, d=209.165.200.254->192.168.1.254
[12]
*Mar 26 14:46:49.539: NAT: s=192.168.1.254->209.165.200.254, d=209.165.200.226
[12]
Mar 26 14:46:49.599: NAT: s=209.165.200.226, d=209.165.200.254->192.168.1.254
[13]
*Mar 26 14:46:49.599: NAT: s=192.168.1.254->209.165.200.254, d=209.165.200.226
[13]
Branch#
Mar 26 14:46:49.655: NAT: s=209.165.200.226, d=209.165.200.254->192.168.1.254
[14]
*Mar 26 14:46:49.655: NAT: s=192.168.1.254->209.165.200.254, d=209.165.200.226
[14]
Branch#
```

The source address is the HQ router IP address going to the branch server global IP address, which is then being translated to its internal IP address. Verifying the NAT translations is displayed in Example 7-23.

Example 7-23. Verifying the NAT Translations

```
Code View: Scroll / Show All
Branch#show ip nat translations
Pro Inside global Inside local Outside local Outside global
-- 209.165.200.249 192.168.1.1 -- --
icmp 209.165.200.254:2 192.168.1.254:2 209.165.200.226:2 209.165.200.226:2
-- 209.165.200.254 192.168.1.254 -- --
```

Branch#

Notice we also have an extended ICMP entry. Entries in the NAT cache eventually expire, and the entry is removed. Notice that a minute later, another debug message informs us that the last ICMP entry has expired, as shown in Example 7-24.

Example 7-24. Verifying the NAT Translations

Code View: Scroll / Show All

Branch#

\*Mar 26 14:47:49.787: NAT: expiring 209.165.200.254 (192.168.1.254) icmp 2 (2)

Branch#

The final NAT configuration configured in this section is displayed in Example 7-25.

Example 7-25. Final NAT Configuration

Code View: Scroll / Show All

Branch#**conf t**

Enter configuration commands, one per line. End with CNTL/Z.

Branch(config)#**ip nat pool BRANCH-NAT-POOL 209.165.200.249 209.165.200.253**  
**prefix-**

**length 29**

Branch(config)#**ip nat inside source list BRANCH-NAT-ACL pool BRANCH-NAT-POOL**

Branch(config)#**!**

Branch(config)#**!**

Branch(config)#**ip nat inside source static 192.168.1.254 209.165.200.254**

Branch(config)#**!**

Branch(config)#**!**

Branch(config)#**ip access-list extended BRANCH-NAT-ACL**

Branch(config-ext-nacl)#**deny ip 192.168.1.0 0.0.0.255 10.10.10.0 0.0.0.255**

Branch(config-ext-nacl)#**permit ip 192.168.1.0 0.0.0.255 any**

Branch(config-ext-nacl)#**exit**

Branch(config)#**interface FastEthernet0/0**

Branch(config-if)#**ip nat inside**

Branch(config-if)#**exit**

Branch(config)#**interface Serial0/0/1**

Branch(config-if)#**ip nat outside**

Branch(config-if)#**end**

Branch#

Verifying Other Services

As mentioned earlier, we also have to verify other services and how they can impact the branch site design.

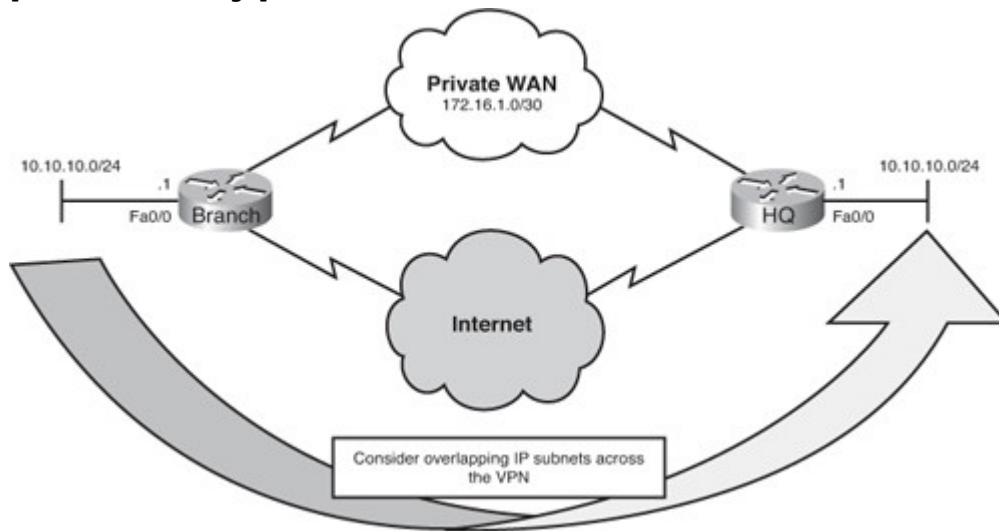
DHCP is a common service used as an initial source of information to see which IP address should be reachable across our VPN connection. Because of the structure of the VPN, the branch office LAN will be considered part of the private corporate network. The headquarters LAN will also be considered part of the private corporate network. Therefore, we must check whether we have overlapping IP addresses between the



two LANs. This could happen when, for example, two organizations merge and they both have IP subnet 10.10.10.0 /24 for their LAN, as shown in Figure 7-14.

Figure 7-14. Overlapping LAN Subnet Address.

[View full size image]



If that were the case, we would have connectivity problem. Each router would think that a packet addressed to 10.10.10.254 is addressed to its local LAN, when in fact it might be for a host in the other LAN.

Additional verification will include checking the access lists currently configured on the routers. A branch router will require an inbound access list applied to its Internet-facing interface. However, there might also be additional security controls on the branch router that block important protocols necessary for the VPN establishment.

For example, edge routers must be capable of forwarding protocols required to support IPsec VPNs, such as the following:

- Encapsulation Security Payload (ESP) protocol, which provides confidentiality through encryption. ESP, located at Layer 4 of the OSI model, uses protocol 50.
- Authentication Header (AH), which is similar to ESP but provides only integrity. AH uses protocol 51.
- Internet Security Association and Key Management Protocol (ISAKMP), which is required during the first stage of IPsec tunnels coming up, when peer negotiations and credentials are exchanged using a protocol. ISAKMP uses the UDP port 500.

#### Note

Securing the edge router interface using access lists and firewalls is beyond the scope of this chapter.

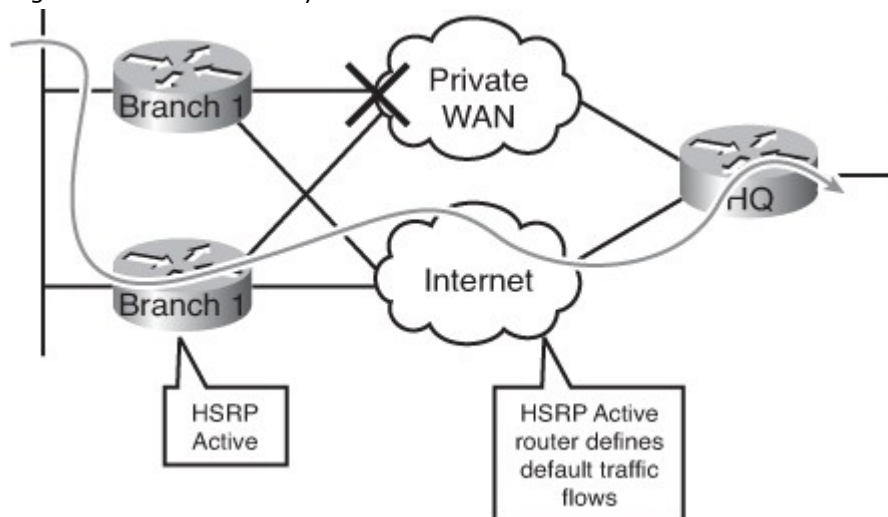
Other services that will require your attention are those that might alter the path of the traffic, especially in the presence of dual connections. Policy-based routing (PBR) can be implemented to redirect traffic.

#### Note

PBR for IPv4 is covered in Chapter 5, "Implementing Path Control," and PBR for IPv6 is covered in Chapter 8, "Implementing IPv6 in an Enterprise Network."

Finally, another service that would have an impact is the Hot Standby Route Protocol (HSRP). This would not be the case with the current topology at our branch office, but other branch offices might have redundancy at the edge routers, as shown in Figure 7-15. HSRP would decide to switch to another active router upon failure and would define the traffic flow.

Figure 7-15. Hot Standby Router Protocol.



#### Note

This topic is discussed further in the SWITCH course and related Cisco Press material.

#### Verifying and Tuning IPsec VPNs

Now that broadband connectivity has been established, and our floating static route works, and NAT is performing properly, we will move on to securing our LAN-to-LAN Internet links using IPsec VPN tunnels. Specifically, we want to allow the branch to use the Internet as a backup connectivity option.

The intent of this section is not to provide detailed coverage of IPsec VPNs. It is about understanding the impact on routing services and addressing schemes when deploying IPsec VPNs at branch office routers.

#### Note

For more information about cryptography and IPsec configuration for IOS routers, see the Cisco Press book *Implementing Cisco IOS Network Security (IINS): (CCNA Security Exam 640-553) (Authorized Self-Study Guide)* by Catherine Paquet, where Chapter 4 deals exclusively with fundamentals of cryptography and Chapter 5 with a site-to-site VPN between a head office and a branch office. In addition, you can find in-depth information about IPsec in the Cisco Networking Academy CCNA Security course.

Using public networks to provide connectivity is a clear trend when designing branch office architectures. The main reasons public networks are connected to branch offices include ubiquitous availability and low cost, as compared to private WANs (the costs of which are often prohibitive for small businesses). However, there are many issues with providing connectivity through the Internet for the private traffic that might be generated within an organization, including the following:

- **Security**— By default, all the traffic leaving on the public network is in clear text and could be read by anyone with the technical know-how and means.
- **Transparency and complexity**— While using a public network to reach the head office, users of branch offices need to have the illusion that they are connected to the corporate network, and as such are using a private IP address compatible with the addressing scheme used by the organization.

IPsec seeks to resolve both issues.

#### IPsec Technologies

IPsec provides two significant benefits:

- **Encryption**— Using a cryptographic algorithm, IPsec encrypts the data exchanged by the corporate offices that are using the public Internet.
- **Encapsulation**— Using tunneling technology, it encapsulate the data as it leaves a corporate site, thus protecting its original IP address and providing the illusion to the recipient that the sender is located within the organization.

IPsec encryption provides three major services:

- **Confidentiality**— Confidentiality provides encryption during the exchange of the data. Only the recipient in possession of the valid key can decrypt the packets. Confidentiality is provided by

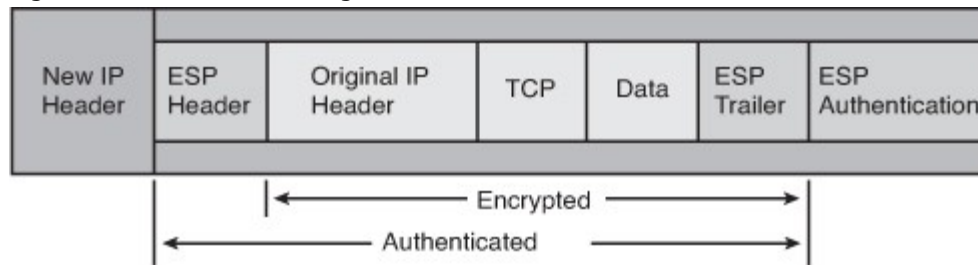
cryptographic algorithms, such as Data Encryption Standard (DES), Triple DES (3DES), and Advanced Encryption Standard (AES).

- **Integrity**— Integrity provides a check to confirm that the data was not altered during the transmission. Integrity checks are provided by hashing algorithms such as message digest algorithm 5 (MD5) and Secure Hash (SHA).
- **Authentication**— Authentication provides assurance that the data is exchanged with the rightful party and not with a hacker who is using the identity of a valid peer. Authentication is provided by signing the results of hashing algorithms.

#### Encapsulation Process

As mentioned, one of the benefits of IPsec is its capability to tunnel packets using an additional encapsulation. For example, in Figure 7-16 the original packet is encapsulated inside a new IP packet before it leaves the branch office.

Figure 7-16. IPsec Tunneling.



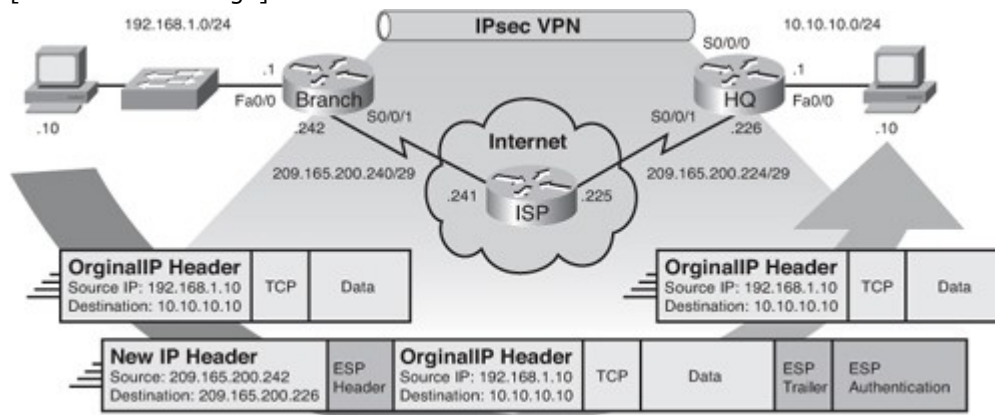
The VPN router responsible for forwarding this packet performs this encapsulation task. At the receiving end, the destination VPN router performs the decapsulation functions, removing the newer IP packet and forwarding the original IP packet to the internal host as it was constructed by the source host.

The IPsec encapsulation process does not just add an additional IP header to the original packet. It also performs security functions. The VPN router responsible for releasing this packet can also encrypt most of the payload of the new packet, providing confidentiality to the transmission.

Figure 7-17 shows a sample scenario.

Figure 7-17. IPsec Tunneling in Action.

[View full size image]



Assume the host on the branch site at IP address 192.168.1.10 wants to contact another host using its internal private IP address (10.10.10.10) and that the link is secured using a site-to-site IPsec VPN.

When the packet from the branch host leaves the branch router, this traffic will be flagged as being interesting and that an IPsec VPN should be established between the branch router and the HQ router. These two routers will then negotiate and secure a tunnel that encapsulates the original IP header into another, secure new IP header. The packet will then be forwarded to the HQ site. Once it arrives at the HQ site, the HQ router will decrypt the packet with the correct preshared key, extracting the IP packet, and forwarding it to the HQ host.

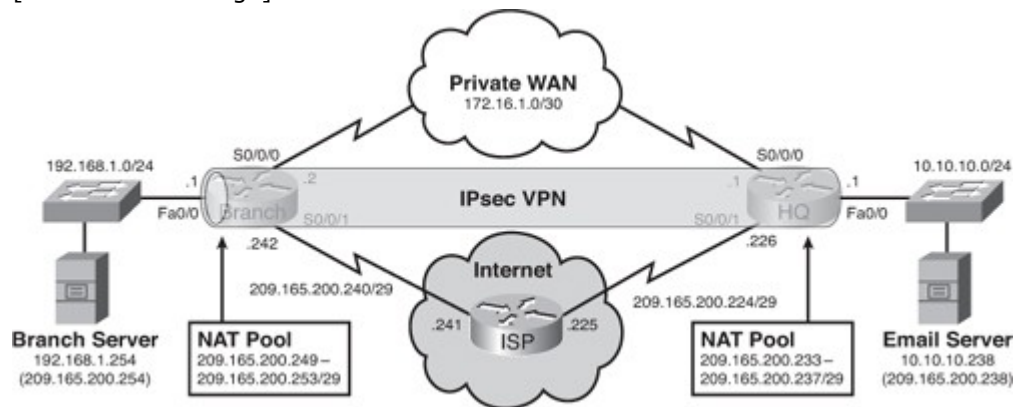
In our scenario, we will assume that our Internet security team has just finished configuring our branch router to support an IPsec VPN with the HQ site. This IPsec VPN is enabled whenever users on the branch LAN connect to another host on the HQ LAN over the Internet connection. Otherwise, the VPN does not

activate. All we have to do is verify its proper configuration and operation. However, all the details of cryptographic services such as confidentiality, integrity, and VPN end-point authentication will be transparent to us.

To guide our discussion, let's take a look at Figure 7-18. Assume that the purpose of the IPsec VPN link is to serve as a backup link in case our private WAN link fails. The long-term goal is to decommission the WAN link completely and use only the VPN connection to communicate between the branch office and the headquarters.

Figure 7-18. Updated Topology with IPsec Tunnel.

[View full size image]



### IPsec Site-to-Site VPN Configuration

To better understand how to verify an IPsec VPN, we must ensure that certain concepts are understood. The steps to configure an IPsec VPN are as follows:

1. Configure the initial key (ISAKMP) details.
2. Configure the IPsec details.
3. Configure the crypto ACL.
4. Configure the VPN tunnel details.
5. Apply the crypto map

The final branch router IPsec VPN configuration is displayed in Example 7-26.

#### Note

The details of cryptology and IPsec VPNs are beyond the scope of this chapter. However, you can find more information about these at Cisco.com.

#### Example 7-26. Sample IPsec VPN Configuration

Code View: Scroll / Show All

Branch#**conf t**

Enter configuration commands, one per line. End with CNTL/Z.

Branch(config)#**crypto isakmp policy 1**

Branch(config-isakmp)#**encryption aes**

Branch(config-isakmp)#**authentication pre-share**

Branch(config-isakmp)#**group 2**

Branch(config-isakmp)#**exit**

Branch(config)#**crypto isakmp key cisco123 address 209.165.200.226**

Branch(config)#

Branch(config)#**crypto ipsec transform-set HQ-VPN esp-sha-hmac esp-3des**

Branch(cfg-crypto-trans)#**exit**

```

Branch(config)#
Branch(config)#crypto map HQ-MAP 10 ipsec-isakmp
% NOTE: This new crypto map will remain disabled until a peer
Branch(config-crypto-map)#set transform-set HQ-VPN
Branch(config-crypto-map)#set peer 209.165.200.226
Branch(config-crypto-map)#match address 110
Branch(config-crypto-map)#exit
Branch(config)#access-list 110 permit ip 192.168.1.0 0.0.0.255 10.10.10.0 0.0.0.255
Branch(config)#
Branch(config)#int s0/0/1
Branch(config-if)#crypto map HQ-MAP
*Mar 24 06:27:18.091: %CRYPTO-6-ISA_KMP_ON_OFF: ISAKMP is ON
Branch(config-if)# ^Z
Branch#

```

Here is a high-level overview of the configuration:

- The ISAKMP policy identifies the specifics for the initial key and secure parameters exchange.
- The IPsec details define how the IP packet will be encapsulated and how it will be identified by the named HQ VPN.
- The VPN tunnel information is identified in the crypto map named HQ-MAP, which combines the ISAKMP policies, IPsec packet detail, the peer address, and ACL 110.
- ACL 110 is the crypto access control list that identifies interesting traffic that will trigger the VPN to activate.
- Finally, the crypto map is applied to the interface.

#### ISAKMP Policy

Whenever an IPsec VPN is initiated, the first stage is to negotiate and exchange credentials with a peer. This exchange uses the protocol called ISAKMP on UDP port 500. The ISAKMP parameters are configured using the **crypto isakmp policy** command, which opens the crypto subconfiguration mode. This command enables you to specify the following:

- Which authenticated method will be used
- Which encryption method to use
- Which hashing method to use
- How long of a random number to use when creating unique key strings between peers
- How long before these parameters have to be exchanged

#### Note

Other optional features that may be configured include Dead Peer Detection (DPD), using the crypto isakmp keepalive command. However, this is beyond the scope of this book. For more information on these topics, see Cisco.com.

#### IPsec Details

IPsec is the framework that enables a VPN tunnel to be created. Specifically, the **crypto ipsec transform-set** command identifies how the packets will be encapsulated by identifying an acceptable combination of security protocols, algorithms, and other settings to apply to IPsec-protected traffic. During the IPsec security association (SA) negotiation, the peers agree to use a particular transform set when protecting a particular data flow.

#### VPN Tunnel Information

Once the ISAKMP and IPsec parameters are identified, the actual VPN tunnel specifics must be entered. The **crypto map** command enters a subconfiguration mode where you can create or edit a named entry that specifies the VPN settings to apply them to an interface.

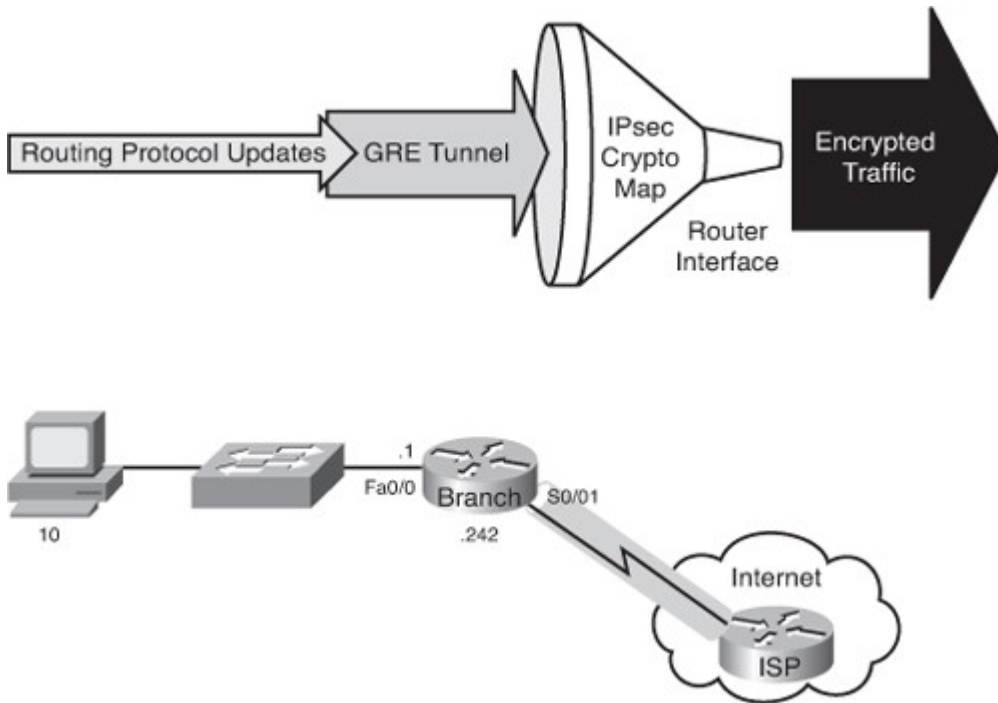
The crypto map is where you specify the following:

- The predefined ISAKMP settings to use
- Which IPsec transform set to use
- Which peer router to establish an IPsec VPN tunnel with
- Which ACL will be used to identify interesting traffic
- How long the security association should be kept before it is renegotiated

Conceptually, a crypto map is similar to a funnel. The funnel, shown in Figure 7-19, channels all configuration components to the appropriate interface, serving as initiation and termination point of IPsec tunnels. You configure the IPsec settings, group them together in a crypto map, and then apply the crypto map to the interface. When traffic meets the criteria, it passes through the funnel, and its policies are enforced. Traffic that does not meet criteria configured in the crypto maps leaves the Internet-facing interface unencrypted.

Figure 7-19. A Crypto Map.

[View full size image]



This crypto map-to-interface association will indicate where initiation and termination of the tunnel occur, thus identifying the interface requiring additional configuration for VPN to work, such as allowing dynamic routing and tuning address translations. In Figure 7-19, crypto maps are applied to the VPN router interface facing the public.

#### VPN ACL

Typically, traffic exiting a branch site is destined for the corporate LAN and thus should be protected. Otherwise, traffic is destined for the Internet and an IPsec VPN is not required to protect it.

The crypto ACL is an extended IP ACL that is used to identify the traffic that should be protected.

A **permit** statement in a crypto ACL will result in the traffic being encrypted, and a **deny** statement will result in the traffic being sent out by the Internet-facing interface in clear text.

Both VPN peers must have reciprocating ACLs. For instance, the branch router requires an extended ACL to identify traffic going from its LAN to the HQ LAN, whereas the HQ router requires an ACL to identify traffic going from its LAN to the branch LAN.

#### Apply the Crypto Map

Finally, the named crypto map must be applied to the Internet-facing interface that the peering router will connect to. The map entry is identified using the **crypto map** interface configuration command. Once configured, all traffic exiting the interface is subject to filtering by the crypto map ACL. If the traffic matches the ACL, the router will begin the process to encrypt and tunnel traffic across to the VPN peer.



## Verifying an IPsec VPN

Various commands can be used to verify whether a VPN is functioning properly. Which you choose will depend on the information you want. For example, to display the specifics contained in a crypto map configuration, use the show crypto map command, as explained in Table 7-7.

Table 7-7. *show crypto map* Command

| Parameter                         | Description                                                                    |
|-----------------------------------|--------------------------------------------------------------------------------|
| <b>interface</b> <i>interface</i> | (Optional) Displays only the crypto map set applied to the specified interface |
| <b>tag</b> <i>map-name</i>        | (Optional) Displays only the crypto map set with the specified map-name        |

To display status information for active crypto sessions, use the show crypto session command. The details are shown in Table 7-8.

Table 7-8. *show crypto session* Command

| Parameter                      | Description                                                                                                                                                                                                                                                                                                                         |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| detail                         | (Optional) Provides more detailed information about the session, such as the capability of the Internet Key Exchange (IKE) SA, connection ID, remaining lifetime of the IKE SA, inbound or outbound encrypted or decrypted packet number of the IPsec flow, dropped packet number, and kilobyte per second lifetime of the IPsec SA |
| local ip-address               | (Optional) Displays status information about crypto sessions of a local crypto endpoint                                                                                                                                                                                                                                             |
| <b>port</b> <i>local-port</i>  | (Optional) Port of the local crypto endpoint                                                                                                                                                                                                                                                                                        |
| remote ip-address              | (Optional) Displays status information about crypto sessions of a remote session                                                                                                                                                                                                                                                    |
| <b>port</b> <i>remote-port</i> | (Optional) Displays status information about crypto sessions of a remote crypto endpoint                                                                                                                                                                                                                                            |
| active                         | (Optional) Displays all crypto sessions in the active state                                                                                                                                                                                                                                                                         |
| standby                        | (Optional) Displays all crypto sessions that are in the standby state                                                                                                                                                                                                                                                               |

Use the show crypto ipsec sa command to display the settings used by current SAs. The details of the command are in Table 7-9.

Table 7-9. *show crypto ipsec sa* Command

| Parameter                         | Description                                                                                                                                                                       |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>map</b> <i>map-name</i>        | (Optional) Any existing SAs that were created for the crypto map set named <i>map-name</i> are displayed.                                                                         |
| address                           | (Optional) All existing SAs are displayed, sorted by the destination address (either the local address or the address of the IPsec remote peer) and then by protocol (AH or ESP). |
| identity                          | (Optional) Only the flow information is displayed. It does not show the SA information.                                                                                           |
| <b>interface</b> <i>interface</i> | (Optional) All existing SAs created for an interface that is named <i>interface</i> are displayed.                                                                                |



To view real time IPsec events, use the **debug crypto ipsec** command. This command has no arguments or keywords.

#### Example of IPsec VPN Verification

Continuing with our scenario, recall that the security team of the organization has configured the Branch and HQ routers to support VPN connectivity. Your role is to confirm that the IPsec tunnel is up and to add the necessary configuration to ensure that routing is successful. Therefore, only the IPsec commands related to confirming that IPsec is working are covered in this book.

In Example 7-26, interesting traffic was identified by ACL 110, and therefore we will initiate interesting traffic and then display the VPN tunnel information using the show crypto session command. However, to observe the IPsec events as they happen, we will initially enable the debug crypto ipsec command, as shown in Example 7-27.

#### Example 7-27. Initiating an IPsec VPN Tunnel

```
Branch#debug crypto ipsec
Crypto IPSEC debugging is on
Branch#ping 10.10.10.1 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
Branch#
Branch#show crypto session
Crypto session current status

Interface: Serial0/0/1
Session status: DOWN
Peer: 209.165.200.226 port 500
IPSEC FLOW: permit ip 192.168.1.0/255.255.255.0 10.10.10.0/255.255.255.0
Active SAs: 0, origin: crypto map

Branch#
```

Although the ping was successful, it appears that the tunnel is down. Recall that in the last implementation step, we implemented NAT. Perhaps this is causing some problems with the IPsec tunnel being created.

To test this, we will enable the debug ip nat command and reissue the extended ping, as shown in Example 7-28.

#### Example 7-28. Debugging NAT in an IPsec VPN Tunnel

```
Code View: Scroll / Show All
Branch#debug ip nat
IP NAT debugging is on
Branch#ping 10.10.10.1 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
Branch#
*Mar 26 16:35:21.251: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [35]
```

```

Mar 26 16:35:21.307: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[35]
*Mar 26 16:35:21.307: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [36]
Mar 26 16:35:21.367: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[36]
*Mar 26 16:35:21.367: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [37]
Mar 26 16:35:21.423: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[37]
*Mar 26 16:35:21.423: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [38]
Mar 26 16:35:21.479: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[38]
Branch#
*Mar 26 16:35:21.483: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [39]
Mar 26 16:35:21.539: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[39]
Branch#

```

Again, the pings are successful. Notice, however, that the internal IP address is being translated to a global NAT IP address, making the source traffic uninteresting.

We started this section with the intention to understand the impact on routing services and addressing schemes when deploying IPsec VPNs at branch office routers. Corporate LAN-to-LAN IPsec traffic does not need to be translated. It should remain private in its path, because it is encapsulated inside another IP packet. However, NAT can interfere with this process.

Because the NAT process takes place before the encryption process, by the time the traffic arrives at the crypto map ACL, it looks like it is from 209.165.200.248 /29 going to 10.10.10.0. In Example 7-29, we used the command `show ip access-lists` to display the content of the configured ACLs.

Example 7-29. `show ip access-lists` 110 Command

```

Branch#show access-lists
Extended IP access list 110
 10 permit ip 192.168.1.0 0.0.0.255 10.10.10.0 0.0.0.255
Extended IP access list BRANCH-NAT-ACL
 10 permit ip 192.168.1.0 0.0.0.255 any (1 match)
Branch#

```

ACL 110 identifies interesting VPN traffic, and the BRANCH-NAT-ACL identifies traffic to be translated. Notice that the crypto map ACL 110 is configured to encrypt traffic between 192.168.1.0/24 to 10.10.10.0/24, but the traffic arrives at the crypto process with a 209.165.200.248 address. So, the crypto map does not encrypt it. We can therefore deduce that our current NAT configuration is creating the problem and will need to be tuned.

The solution to this NAT problem is to create a NAT exemption. The NAT access list that defines traffic that will be translated can also identify when traffic should not be translated. For the NAT process, a **permit** line in an access list means "translate," and a **deny** line means "do not translate."

On our branch router, we need to alter the NAT ACL so that it excludes the VPN traffic from being translated, as shown in Example 7-30.

Example 7-30. Alter the NAT ACL

```

Branch#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Branch(config)#no ip access-list extended BRANCH-NAT-ACL
Branch(config)#ip access-list extended BRANCH-NAT-ACL
Branch(config-ext-nacl)#deny ip 192.168.1.0 0.0.0.255 10.10.10.0

```

**0.0.0.255**

```
Branch(config-ext-nacl)#permit ip 192.168.1.0 0.0.0.255 any
Branch(config-ext-nacl)#^Z
Branch#
```

Now test our link again using an extended ping, as shown in Example 7-31.

Example 7-31. Test the VPN Link

```
Code View: Scroll / Show All
Branch#ping 10.10.10.1 source 192.168.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
Branch#
*Mar 26 17:47:15.842: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [70]
Mar 26 17:47:15.898: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[70]
*Mar 26 17:47:15.902: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [71]
Mar 26 17:47:15.958: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[71]
*Mar 26 17:47:15.958: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [72]
Mar 26 17:47:16.014: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[72]
*Mar 26 17:47:16.018: NAT: s=192.168.1.1->209.165.200.249, d=10.10.10.1 [73]
Mar 26 17:47:16.074: NAT: s=209.165.200.238, d=209.165.200.249->192.168.1.1
[73]
Branch#
```

Again, the ping is successful, but it appears that NAT still translated the inside LAN address. Verify the NAT translation as shown in Example 7-32.

Example 7-32. Display the NAT Translations

```
Code View: Scroll / Show All
Branch#show ip nat translations
Pro Inside global Inside local Outside local Outside global
icmp 209.165.200.249:18 192.168.1.1:18 209.165.200.238:18
209.165.200.238:18
-- 209.165.200.249 192.168.1.1 -- --
-- 209.165.200.254 192.168.1.254 -- --
Branch#
```

Notice that the 192.168.1.1 address is still in the NAT cache. This is the cause of our current problem. The NAT translations should be cleared, and only then will the branch router enforce the new BRANCH-NAT-ACL entries.

Clear the NAT translations and reissue the extended ping, as shown in Example 7-33.

### Example 7-33. Clear the NAT Translation and Test the VPN Link

Code View: Scroll / Show All

```
Branch#clear ip nat translation *
```

```
Branch#clear crypto isakmp
```

```
Branch#clear crypto sa
```

```
Branch#ping 10.10.10.1 source 192.168.1.1
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:

Packet sent with a source address of 192.168.1.1

```
*Mar 26 18:28:45.166: IPSEC(sa_request): ,
(key eng. msg.) OUTBOUND local= 209.165.200.242, remote= 209.165.200.226,
local_proxy= 192.168.1.0/255.255.255.0/0/0 (type=4),
remote_proxy= 10.10.10.0/255.255.255.0/0/0 (type=4),
protocol= ESP, transform= esp-3des esp-sha-hmac (Tunnel),
lifedur= 3600s and 4608000kb,
spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x0
*Mar 26 18:28:45.730: IPSEC(validate_proposal_request): proposal part #1
*Mar 26 18:28:45.730: IPSEC(validate_proposal_request): proposal part #1,
(key eng. msg.) INBOUND local= 209.165.200.242, remote= 209.165.200.226,
local_proxy= 192.168.1.0/255.255.255.0/0/0 (type=4),
remote_proxy= 10.10.10.0/255.255.255.0/0/0 (type=4),
protocol= ESP, transform= NONE (Tunnel),
lifedur= 0s and 0kb,
spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x0
*Mar 26 18:28:45.730: Crypto mapdb : proxy_match

*Mar 26 18:28:45.734: IPSEC(key_engine): got a queue event with 1 KMI message(s)
*Mar 26 18:28:45.734: Crypto mapdb : proxy_match

*Mar 26 18:28:45.734: IPSEC(crypto_ipsec_sa_find_ident_head): reconnecting with
the same proxies
and peer 209.165.200.226
*Mar 26 18:28:45.734: IPSEC(policy_db_add_ident): src 192.168.1.0, dest
10.10.10.0, dest_port 0

*Mar 26 18:28:45.734: IPSEC(create_sa): sa created,
(sa) sa_dest= 209.165.200.242, sa_proto= 50,
sa_spi= 0xADDF016B(2917073259),
sa_trans= esp-3des esp-sha-hmac , sa_conn_id= 2009
*Mar 26 18:28:45.738: IPSEC(create_sa): sa created,
(sa) sa_dest= 209.165.200.226, sa_proto= 50,
sa_spi= 0x1C838B72(478382962),
sa_trans= esp-3des esp-sha-hmac , sa_conn_id= 2010
*Mar 26 18:28:45.738: IPSEC(update_current_outbound_sa): updated peer
209.165.200.226 current
outbound sa to SPI 1C838B72!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 88/89/92 ms
```

Branch#

It appears our VPN link has been activated, as evidenced by the generated output of the **debug crypto ipsec** command. Notice, too, that four out of the five pings were successful. Typically, the initial traffic that initiates the VPN tunnel may time out, which in this case made the first ping fail.

Example 7-34 provides a quick look at the show crypto session command.

Example 7-34. Display Crypto Session Information

Branch#**show crypto session**

Crypto session current status

Interface: Serial0/0/1

Session status: UP-ACTIVE

Peer: 209.165.200.226 port 500

IKE SA: local 209.165.200.242/500 remote 209.165.200.226/500 Active

IPSEC FLOW: permit ip 192.168.1.0/255.255.255.0 10.10.10.0/255.255.255.0

Active SAs: 2, origin: crypto map

Branch#

As you can see, the session is active with the HQ peer. As an alternative, we could have added the detail keyword as shown in Example 7-35.

Example 7-35. Display the Crypto Session with Detail Information

Branch#**show crypto session detail**

Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection

K - Keepalives, N - NAT-traversal, T - cTCP encapsulation

X - IKE Extended Authentication, F - IKE Fragmentation

Interface: Serial0/0/1

Uptime: 00:28:17

Session status: UP-ACTIVE

Peer: 209.165.200.226 port 500 fvrf: (none) ivrf: (none)

Phase1\_id: 209.165.200.226

Desc: (none)

IKE SA: local 209.165.200.242/500 remote 209.165.200.226/500 Active

Capabilities:(none) connid:1005 lifetime:23:31:42

IPSEC FLOW: permit ip 192.168.1.0/255.255.255.0 10.10.10.0/255.255.255.0

Active SAs: 2, origin: crypto map

Inbound: #pkts dec'ed 4 drop 0 life (BPSKBPSec) 4444197/1902

Outbound: #pkts enc'ed 4 drop 1 life (BPSKBPSec) 4444197/1902

Branch#

Now we will issue the show crypto ipsec sa command to display the specifics about the current SA, as shown in Example 7-36.

Example 7-36. Display the Crypto IPsec SA Information

Code View: Scroll / Show All

Branch#**show crypto ipsec sa**

interface: Serial0/0/1

Crypto map tag: HQ-MAP, local addr 209.165.200.242

protected vrf: (none)

local ident (addr/mask/prot/port): (192.168.1.0/255.255.255.0/0/0)

remote ident (addr/mask/prot/port): (10.10.10.0/255.255.255.0/0/0)

current\_peer 209.165.200.226 port 500

PERMIT, flags={origin\_is\_acl,}

#pkts encaps: 4, #pkts encrypt: 4, #pkts digest: 4

#pkts decaps: 4, #pkts decrypt: 4, #pkts verify: 4

#pkts compressed: 0, #pkts decompressed: 0

#pkts not compressed: 0, #pkts compr. failed: 0

#pkts not decompressed: 0, #pkts decompress failed: 0

#send errors 1, #recv errors 0

local crypto endpt.: 209.165.200.242, remote crypto endpt.: 209.165.200.226

path mtu 1500, ip mtu 1500, ip mtu idb Serial0/0/1

current outbound spi: 0x1C838B72(478382962)

inbound esp sas:

spi: 0xADD016B(2917073259)

transform: esp-3des esp-sha-hmac ,

in use settings = {Tunnel, }

conn id: 2009, flow\_id: NETGX:9, crypto map: HQ-MAP

sa timing: remaining key lifetime (k/sec): (4444197/3572)

IV size: 8 bytes

replay detection support: Y

Status: ACTIVE

inbound ah sas:

inbound pcg sas:

outbound esp sas:

spi: 0x1C838B72(478382962)

transform: esp-3des esp-sha-hmac ,

in use settings = {Tunnel, }

conn id: 2010, flow\_id: NETGX:10, crypto map: HQ-MAP

sa timing: remaining key lifetime (k/sec): (4444197/3572)

IV size: 8 bytes

replay detection support: Y

Status: ACTIVE

outbound ah sas:

outbound pcg sas:

Branch#

The command confirms that the branch router and HQ router have an established VPN. Also notice that the four successful pings were encapsulated and the return pings were decapsulated.

We have just demonstrated the link and dependencies between VPN and common router services such as NAT.

#### Impact on Routing

One of the significant drawbacks of an IPsec VPN is that it cannot route multicast and broadcast packets. Therefore, interior gateway protocols (IGPs) such as EIGRP and OSPF that use multicast packets cannot send routing advertisements through an IPsec VPN. However, IPsec can be combined with generic routing encapsulation (GRE) to create a tunnel to circumvent the issue with IGP routing within VPN tunnels.

#### Configuring GRE Tunnels

Even though you might not be responsible for configuring VPN tunnels (after all, it is often the responsibility of the security group staff to do so), you might have to build routing around them. Therefore, if you want the Internet connection to use the IPsec VPN and become the main link back to headquarters, additional configuration will be needed to support dynamic routing protocols.

There are four options to route dynamic routing protocols through an IPsec tunnel:

- Point-to-point generic routing encapsulation (P2P GRE)
- Virtual tunnel interface (VTI)
- Dynamic multipoint VPN (DMVPN)
- Group encrypted transport VPN (GET VPN)

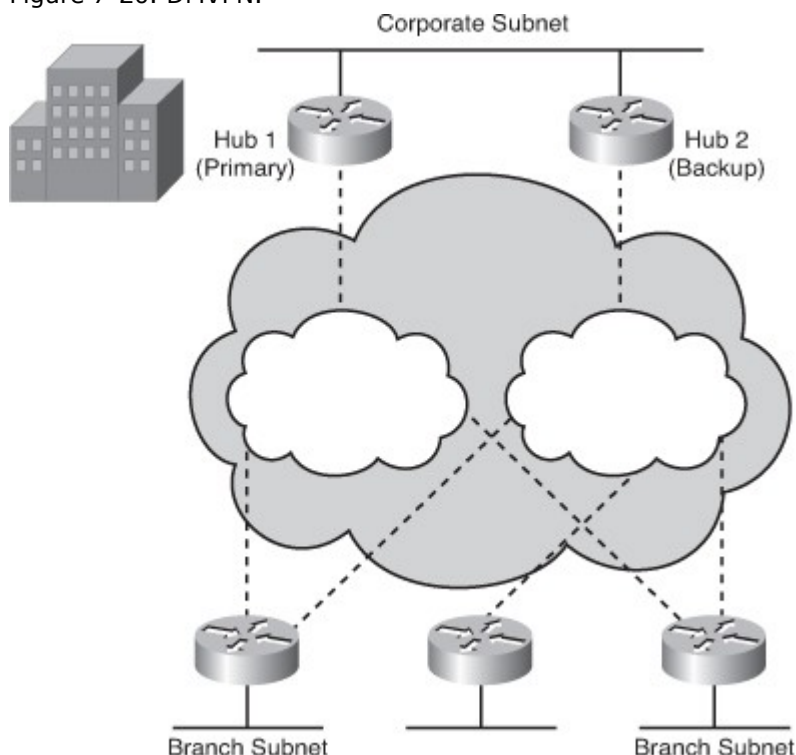
In this section, we focus on P2P GRE. However, the other options are briefly discussed.

A major benefit of VTI is that the configuration does not require a static mapping of an IPsec session to a physical interface. The IPsec endpoint is associated with an actual virtual interface. This is then a routable interface at the tunnel endpoint, and many common interface capabilities can be applied to the IPsec tunnel. They include the association of routing protocols and therefore routing across the VPN tunnel. In the end, VTI is a good alternative to IPsec over GRE tunnels.

Looking ahead, it is common for dual headend routers to be used for redundancy in bigger scenarios. While looking at Figure 7-20, think of the implications of a full-meshed GRE tunnel where hub-to-spoke routing and spoke-to-spoke routing are required. Nowadays, with VoIP, branch offices expect to call other branch offices directly, without going through headquarters, which would only contribute latency. Configuring GRE tunnels is the way we will do it in the next section; for point-to-point connectivity between every spoke, does not scale well.



Figure 7-20. DMVPN.



Scalable alternatives to GRE when full-mesh connectivity is necessary between the branches are DMVPN and GET VPN. These technologies combine multipoint GRE tunnels, dynamic resolutions of endpoints, and crypto profiles that overwrite the requirement for defining crypto maps.

So, in preparation for more complex scenarios, you may want to read up on backup interfaces, VTI, DMVPN, and GET VPN.

#### Note

VTI, DMVPN, and GET VPN are beyond the scope of this book. For more information on these topics, see Cisco.com.

#### Generic Routing Encapsulation

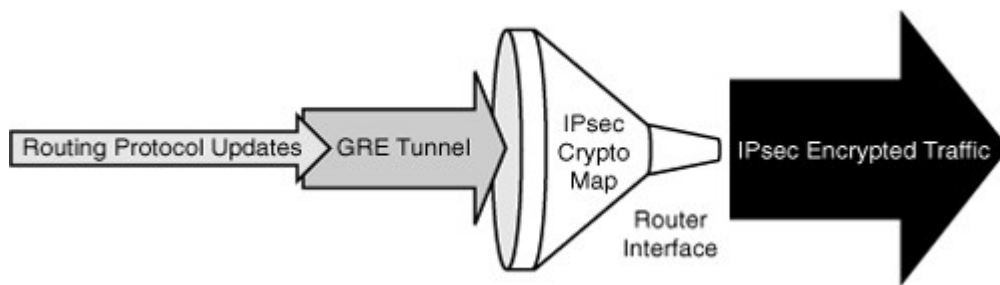
GRE is a tunneling protocol developed by Cisco. It is capable of encapsulating a wide variety of network layer protocols packets inside IP tunnels. This creates virtual point-to-point links. It is a common option to use GRE to pass dynamic routing protocol traffic across an IPsec tunnel.

It is worth mentioning, though, that GRE tunnels do not provide encryption services. GRE is just an encapsulation protocol. It does not offer other services such as encryption. By default, the traffic leaves in clear text. In our implementation, however, GRE packets will be encrypted by IPsec.

Point-to-point GRE encapsulates routing protocols in GRE first, and then the GRE packets are encapsulated in IPsec and encrypted. Figure 7-21 shows the encapsulation process. The routing protocols will be associated with tunnel interfaces, which will use the physical interface of the router to send GRE traffic that will then have to match the parameters of the crypto map and therefore be encrypted by IPsec.

Figure 7-21. GRE and IPsec Encapsulation Process.

[View full size image]



With GRE, multicasts and broadcasts will be carried inside unicast packets, which IPsec can forward.

Working with the example we have used in the previous section of this chapter, the payload of GRE packets will be EIGRP routing updates and LAN-to-LAN corporate traffic. The GRE packet will then be encapsulated inside an IPsec packet. This means that the payload of the IPsec packet is a GRE packet, which itself has as payload of an EIGRP packet or user traffic. Therefore, IPsec is the "transport protocol," and GRE is the "carrier protocol" used to carry other "passenger protocols," such as IP broadcast or IP multicast, and non-IP protocols, as shown in Figure 7-22.

Figure 7-22. Transport, Carrier, and Passenger Protocols.

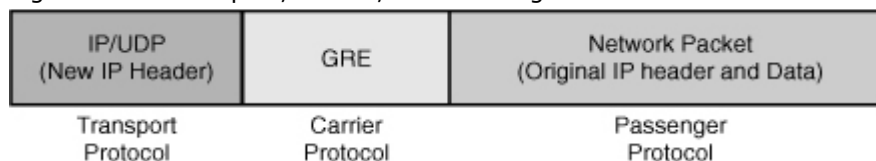
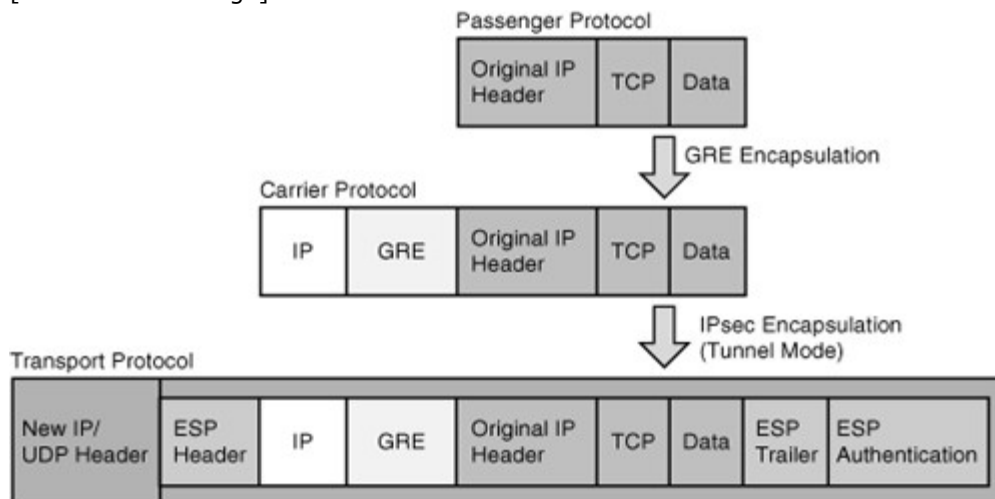


Figure 7-23 shows the extra overhead created by these additional encapsulations (necessary overhead if we want to pass routing protocols with the current implementation).

Figure 7-23. GRE and IPsec: Tunnel Within a Tunnel.

[View full size image]



### Configuring GRE

These following three configuration steps will help us accomplish our goal:

1. Create tunnel interfaces for GRE.
2. Change the crypto ACL to encrypt GRE traffic.
3. Configure routing protocols to route through the GRE tunnel.

We will first configure the tunnel interfaces with GRE encapsulation. We will make sure that the tunnel is up and running. Next, we will make a change to the IPsec configuration to include GRE traffic to the crypto ACL. This will cause GRE traffic, and with it the routing updates, to be channeled across the IPsec VPN tunnel.

Finally, we will configure our routing protocol to use the tunnel interface as a routable interface and therefore advertise on it.

#### Note

This section covers the basics of configuring GRE over IPv4. For more information, see the Point-to-Point GRE over IPsec Design Guide located atCisco.com ([http://www.cisco.com/en/US/customer/docs/solutions/Enterprise/WAN\\_and\\_MAN/P2P\\_GRE\\_IP\\_Sec/P2P\\_GRE\\_IPSec.html](http://www.cisco.com/en/US/customer/docs/solutions/Enterprise/WAN_and_MAN/P2P_GRE_IP_Sec/P2P_GRE_IPSec.html)).

A GRE tunnel is created using the interface tunnel 0 command. A tunnel interface is a virtual interface that is assigned an IP address to identify the end of the tunnel. After the tunnel interface has been created and has been configured with an IP address, the actual source of the tunnel is identified using the tunnel source {ip-address | interface-type interface-number} command shown in Table 7-10.

Table 7-10. *tunnel source* Command

| Parameter        | Description                                                                                                                                                                                            |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ip-address       | IP address to use as the source address for packets in the tunnel.                                                                                                                                     |
| interface-type   | Interface type, such as loopback interface.                                                                                                                                                            |
| interface-number | Port, connector, or interface card number. The numbers are assigned at the factory at the time of installation or when added to a system and can be displayed with the <b>show interfaces</b> command. |

#### Note

A common best practice is to create a loopback interface, which is then used as the tunnel source IP address.

We also identify the other endpoint of the tunnel with the tunnel destination command shown in Table 7-11.

Table 7-11. *tunnel destination* Command

| Parameter  | Description                                                             |
|------------|-------------------------------------------------------------------------|
| type       | Type of interface to be configured                                      |
| host-name  | Name of the host destination                                            |
| ip-address | IP address of the host destination expressed in dotted-decimal notation |

A GRE tunnel can be configured to operate within an IPv4, IPv6, or multipoint configuration using the **tunnel mode gre** command. However, by default the encapsulation of GRE is within an IPv4 packet, and therefore the **tunnel mode gre ip** command is not required.

#### Note

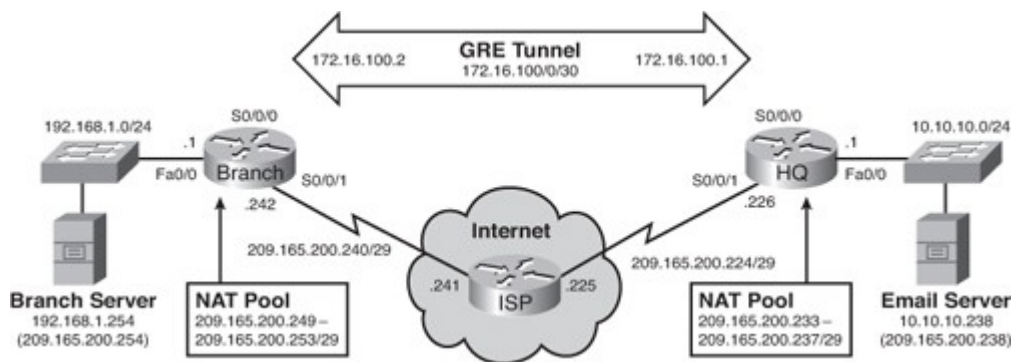
The **tunnel mode gre ipv6** command is required when configuring GRE over IPv6. This command will be used in the next chapter.

#### Example of GRE Configuration

Recall that the long-term goal is to decommission to WAN link completely and use only the VPN connection to communicate between the branch office and the headquarters. We are now ready to so. We'll use the configuration in Figure 7-24 as our guide as we progress.

Figure 7-24. Topology of the Example Scenario for GRE Configuration.

[View full size image]



Although we now have a working IPsec VPN solution, we need to add routing functionality between sites using a GRE tunnel. To do so, we need to do the following:

- Create a tunnel interface and configure the specifics such as the tunnel IP address, tunnel source, and tunnel destination.
- Change the crypto ACL to encrypt GRE traffic.
- Configure routing protocols to route through the GRE tunnel.

To avoid errant EIGRP neighbor messages from appearing, we will first begin by removing the dynamic routing protocol, as shown in Example 7-37.

Example 7-37. Disable EIGRP

```
Branch(config)#no router eigrp 1
Branch(config)#
```

Now we configure the GRE Tunnel 0 interface, as shown in Example 7-38.

Example 7-38. Creating a Tunnel Interface on the Branch Router

```
Code View: Scroll / Show All
Branch(config)#interface tunnel 0
Branch(config-if)#ip address 172.16.100.2 255.255.255.252
Branch(config-if)#tunnel source 209.165.200.242
Branch(config-if)#tunnel destination 209.165.200.226
Branch(config-if)#
*Mar 27 15:45:05.647: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0,
changed state to
up

Branch(config-if)#
```

The tunnel IP address is 172.16.100.2 /30, which will serve as the tunnel destination in the HQ router tunnel configuration. The tunnel source is the reachable IP address of the Internet-facing interface on the branch router. The **tunnel source** command is used to specify either the source interface or the source IP address. We have chosen to specify the IP address. The destination of the tunnel interface will be the reachable global IP address of the HQ router.

Notice the console message informing us that the tunnel interface is active.

Note

GRE over IP is the default encapsulation of GRE tunnel, and therefore the tunnel interface command **tunnel mode gre ip** is not required.

Next, we repeat the preceding configurations on the HQ router, as shown in Example 7-39.

Example 7-39. Creating a loopback and Tunnel Interfaces on the HQ Router

```
Code View: Scroll / Show All
HQ(config-if)#no router eigrp 1
HQ(config)#interface Tunnel0
HQ(config-if)#ip address 172.16.100.1 255.255.255.252
HQ(config-if)#tunnel source 209.165.200.226
HQ(config-if)#tunnel destination 209.165.200.242
HQ(config-if)#
*Mar 27 10:50:59.151: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0,
changed state to
up

HQ(config)#
```

Again, we receive a console message informing us that the tunnel interface is active.

Note

Optionally, the **keepalive** command could also be configured to provide a trigger mechanism to cause the line protocol to be changed from an up/up to an up/down state during a failure event.

To verify the current tunnel interface configuration, we can use the show ip interfaces command, as shown in Example 7-40.

Example 7-40. show interface tunnel 0 on the Branch Router

```
Code View: Scroll / Show All
Branch#show interfaces tunnel 0
Tunnel0 is up, line protocol is up
 Hardware is Tunnel
 Internet address is 172.16.100.2/30
 MTU 17916 bytes, BW 100 Kbit/sec, DLY 50000 usec,
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation TUNNEL, loopback not set
 Keepalive not set
 Tunnel source 209.165.200.242, destination 209.165.200.226
 Tunnel protocol/transport GRE/IP
 Key disabled, sequencing disabled
 Checksumming of packets disabled
 Tunnel TTL 255
 Fast tunneling enabled
 Tunnel transport MTU 1476 bytes
 Tunnel transmit bandwidth 8000 (kbps)
 Tunnel receive bandwidth 8000 (kbps)
 Last input never, output never, 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: 0/0 (size/max)
 5 minute input rate 0 bits/sec, 0 packets/sec
 5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
```

```
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
0 packets output, 0 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 unknown protocol drops
0 output buffer failures, 0 output buffers swapped out
Branch#
```

Notice that the tunnel is reported to be up/up state and that the encapsulation used is tunnel. We can also confirm the source and destination tunnel IP addresses and that the tunnel protocol is GRE over IP. Remember that GRE over IP is the default for tunnel interfaces, and therefore we did not need to configure that tunnel mode.

No traffic is currently using these tunnel interfaces because our dynamic routing process is not yet aware that it has to use them to communicate.

For this reason, we must now change the crypto ACL to make the GRE traffic interesting and enable the IPsec tunnel. To do so, we will remove the current crypto ACL and replace it, as shown in Example 7-41.

Example 7-41. Replacing the Crypto ACL on the Branch Router

```
Code View: Scroll / Show All
Branch(config)#no access-list 110
Branch(config)#access-list 110 permit gre host 209.165.200.242 host
209.165.200.226
Branch(config)#
```

The new crypto map ACL specifies that whenever the public IP address of the branch router attempts to send a GRE update to the public IP address of the HQ router an IPsec VPN should be enabled. The reciprocating crypto map is configured on the HQ router, as shown in Example 7-42.

Example 7-42. Replacing the Crypto ACL on the HQ Router

```
Code View: Scroll / Show All
HQ(config)#no access-list 110
HQ(config)#$ access-list 110 permit gre host 209.165.200.226 host
209.165.200.242
HQ(config)#
```

We should now have basic GRE over IPv4 connectivity. Ping the tunnel IP address as shown in Example 7-43.

Example 7-43. Activating the GRE Tunnel

```
Branch#ping 172.16.100.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.100.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 96/99/100 ms
Branch#
```

The pings are 80 percent successful, indicating that perhaps the first ping timed out because of the IPsec VPN being activated. Verify the VPN tunnel information using the show crypto session command, as shown in Example 7-44.

Example 7-44. Verifying the IPsec VPN Tunnel

```
Branch#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation

Interface: Serial0/0/1
Uptime: 00:01:38
Session status: UP-ACTIVE
Peer: 209.165.200.226 port 500 fvrf: (none) ivrf: (none)
 Phase1_id: 209.165.200.226
 Desc: (none)
 IKE SA: local 209.165.200.242/500 remote 209.165.200.226/500 Active
 Capabilities:(none) connid:1002 lifetime:23:58:21
 IPSEC FLOW: permit 47 host 209.165.200.242 host 209.165.200.226
 Active SAs: 2, origin: crypto map
 Inbound: #pkts dec'd 4 drop 0 life (BPSKBPSec) 4495373/3501
 Outbound: #pkts enc'd 4 drop 1 life (BPSKBPSec) 4495373/3501
Branch#
```

#### Note

The IPSEC FLOW is permitting IP protocol 47, which is the protocol number for GRE.

Now we must verify connectivity from the branch LAN to the HQ LAN, as shown in Example 7-45.

Example 7-45. Verifying LAN to LAN Connectivity

```
Branch#ping 10.10.10.1 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
.....
Success rate is 0 percent (0/5)
Branch#
```

Now the LANs can no longer reach each other. Verify the routing table of the branch router, as shown in Example 7-46.

Example 7-46. Routing Table of the Branch Router

```
Code View: Scroll / Show All
Branch#show ip route
Codes: 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

Gateway of last resort is 209.165.200.241 to network 0.0.0.0

172.16.0.0/30 is subnetted, 1 subnets

C 172.16.100.0 is directly connected, Tunnel0

209.165.200.0/29 is subnetted, 1 subnets

C 209.165.200.240 is directly connected, Serial0/0/1

C 192.168.1.0/24 is directly connected, FastEthernet0/0

S\* 0.0.0.0/0 [171/0] via 209.165.200.241

Branch#

Notice that we have the 172.16.100.0 network connected to the Tunnel 0 interface. In addition, we still have the default static route we configured earlier. However, the branch LAN does not know about the HQ LAN located on 10.10.10.0 /24.

We will now configure EIGRP to propagate the LAN and the tunnel routing information between the sites, as shown in Example 7-47.

Example 7-47. Enabling EIGRP on the Branch Router

```
Branch(config)#router eigrp 1
Branch(config-router)#network 192.168.1.0
0.0.0.255
Branch(config-router)#network 172.16.100.0 0.0.0.3
Branch(config-router)#no auto-summary
Branch(config-router)#
```

Next, we configure the HQ router to propagate its LAN and the tunnel routing information, as shown in Example 7-48.

Example 7-48. Enabling EIGRP on the HQ Router

```
Code View: Scroll / Show All
HQ(config)#router eigrp 1
HQ(config-router)#network 10.10.10.0 0.0.0.255
HQ(config-router)#network 172.16.100.0 0.0.0.3
HQ(config-router)#no auto-summary
HQ(config-router)#
*Mar 27 12:02:52.483: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor
172.16.100.2
(Tunnel0) is up:
new adjacency

HQ(config-router)#
```

Notice that right away EIGRP formed an adjacency with the branch router using the tunnel interface IP address 172.16.100.2.

From the branch router, verify the EIGRP neighbor, as shown in Example 7-49.

#### Example 7-49. Verifying the EIGRP Neighbor

```
Branch#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H Address Interface Hold Uptime SRTT RTO Q Seq
 (sec) (ms) Cnt Num
0 172.16.100.1 Tu0 14 00:00:27 92 2151 0 3
Branch#
```

As you can see, we have the HQ router as an EIGRP neighbor through the Tunnel 0 interface. Now verify the routing table for only specific EIGRP information, as shown in Example 7-50.

#### Example 7-50. Verifying the EIGRP Routing Table

```
Branch#show ip route eigrp 1
10.0.0.0/24 is subnetted, 1 subnets
D 10.10.10.0 [90/26882560] via 172.16.100.1, 00:04:40, Tunnel0
Branch#
```

Notice that we see the HQ LAN listed in the routing table.

Now that we have configured EIGRP to work over the IPsec tunnel, using GRE, we will do some tests. Verify connectivity to it using an extended ping and trace commands, as shown in Example 7-51.

#### Example 7-51. Verifying the EIGRP Routing Table

```
Branch#ping 10.10.10.1 source 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/100 ms
Branch#
Branch#trace 10.10.10.1 source 192.168.1.1

Type escape sequence to abort.
Tracing the route to 10.10.10.1

 1 172.16.100.1 68 msec 68 msec *
Branch#
```

Both are successful. Notice how it appears that the trace has traveled to only one other router. This confirms that packets are indeed traversing the IPsec VPN. Otherwise, the trace would have displayed the actual router IP addresses between the branches and HQ routers.

Verify the IPsec counters using the show crypto session detail command, as shown Example 7-52.

#### Example 7-52. Verifying the IPsec VPN Information

```
Branch#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation

Interface: Serial0/0/1
```

```
Uptime: 00:35:47
Session status: UP-ACTIVE
Peer: 209.165.200.226 port 500 fvrf: (none) ivrf: (none)
 Phase1_id: 209.165.200.226
 Desc: (none)
IKE SA: local 209.165.200.242/500 remote 209.165.200.226/500 Active
 Capabilities:(none) connid:1002 lifetime:23:24:11
IPSEC FLOW: permit 47 host 209.165.200.242 host 209.165.200.226
 Active SAs: 2, origin: crypto map
 Inbound: #pkts dec'ed 142 drop 0 life (BPSKBPSec) 4495354/1452
 Outbound: #pkts enc'ed 211 drop 1 life (BPSKBPSec) 4495345/1452

Branch#
```

As confirmed in the output, many packets are being decrypted and encrypted on this link. However, not all packets have been generated by the **ping** and **trace** commands. Remember that EIGRP is also being transmitted across the IPsec VPN.

Now we test to see which path is taken when we do an extended trace from the branch LAN to the HQ Internet address, as shown in Example 7-53.

Example 7-53. Verifying Internet Connectivity

```
Branch#trace 209.165.200.226 source
192.168.1.1

Type escape sequence to abort.
Tracing the route to 209.165.200.226

 1 209.165.200.241 12 msec 12 msec 16 msec
 2 209.165.200.226 28 msec 28 msec *
```

As you can see, regular traffic does not take the GRE over IPsec VPN tunnel.

Example 7-54 shows the final GRE over IPsec configuration in this section.

Example 7-54. Final GRE over IPsec Configuration on the Branch Router

```
Code View: Scroll / Show All
Branch#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Branch(config)#interface tunnel 0
Branch(config-if)#ip address 172.16.100.2 255.255.255.252
Branch(config-if)#tunnel source 209.165.200.242
Branch(config-if)#tunnel destination 209.165.200.226
Branch(config-if)#
*Mar 27 21:58:46.631: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up
Branch(config-if)#exit
Branch(config)#
Branch(config)#
Branch(config)#no access-list 110
Branch(config)#access-list 110 permit gre host 209.165.200.242 host 209.165.200.226
Branch(config)#
Branch(config)#router eigrp 1
```

```
Branch(config-router)#network 192.168.1.0 0.0.0.255
Branch(config-router)#network 172.16.100.0 0.0.0.3
Branch(config-router)#no auto-summary
Branch(config-router)#end
Branch#
```

This concludes our implementation plan to upgrade a branch office. Specifically, we followed these steps:

- Step 1. Deployed broadband connectivity
- Step 2. Configured static routing
- Step 3. Verified other services
- Step 4. Implemented and tuned the IPsec VPN
- Step 5. Configured GRE tunnels

We have now completed the branch office update. It is time for us to explore routing solutions for the mobile worker.

#### Planning for Mobile Worker Implementations

This section examines the challenges of connecting an increasingly mobile workforce. Mobile workers have become power users who may not even need a full-time office but require a professional environment and support services on-demand. From collaboration to presence services, remote-access solutions are an extension of the converged network and present similar requirements in terms of security, quality of service (QoS), and management transparency.

##### Connecting a Mobile Worker

The enterprise mobile worker solution provides an always-on, secure, centrally managed connection from multiple global locations to the corporate network. It enables businesses to meet several requirements, including the following:

- Provide continuity of operations in case of loss of access to the workplace by inclement weather, commuter issues, man-made and natural disasters, and so on
- Increase responsiveness across geographical, functional, business, and decision-making boundaries
- Provide secure, reliable, manageable employee access to critical network assets and confidential information
- Cost-effectively extend data, voice, video, and real-time applications over a common network connection
- Increase employee productivity, satisfaction, and retention

The first consideration that needs to be addressed when connecting the mobile worker is the choice of a suitable network access technology.

The mobile worker typically uses diverse applications, such as e-mail, web-based applications, mission-critical applications, real-time collaboration, voice, video, and videoconferencing, many of which require a high-bandwidth connection. Therefore, the first factors to consider in a remote connectivity solution are the access network technology and the bandwidth availability.

Possible options to provide high bandwidth include residential cable, DSL, and wireless solutions. Further considerations involve infrastructure services options, such as the following:

- **IPsec and Secure Sockets Layer (SSL) VPNs**— Establish a secure tunnel over existing broadband connections between the mobile worker's remote sites, and from laptops and desktops, and the central site. Site-to-site VPNs are used to achieve an always-on transparent VPN connection for home users. Remote-access VPNs are used to provide an on-demand secured connection from multiple locations. The choice of IPsec or SSL VPNs depends on multiple factors such as choice of applications, end-point management issues, and others.
- **Security**— You want security to safeguard the corporate network and prevent unguarded back doors. The security measures are achieved by deploying firewall, intrusion prevention, and URL filtering

services. Depending on the enterprise corporate security policy, split tunneling may be used to share the broadband connection between secured corporate access and unsecured Internet access at the same time. Split tunneling allows the remote worker to access public Internet sites while connected to the private network, and might result in security concerns.

- **Authentication**— Authentication defines who gets access to resources and is achieved by deploying identity-based network services with authentication using AAA servers, 802.1X port-based access control, Cisco security, and trust agents.
- **QoS**— Quality of service addresses application availability and behavior. The QoS mechanisms have to be used to prioritize the traffic, optimize the use of WAN bandwidth, address the difference in uplink and downlink speed of a broadband connection, and achieve adequate performance for applications sensitive to delay and jitter, such as voice and video.
- **Management**— Management addresses the complexity of support and the loss of corporate control. IT centrally manages and supports the mobile worker connection and equipment, and transparently configures and pushes security and other policies to the remote devices. Tools can be used to implement performance and fault management and to monitor service level agreements (SLAs).

#### Components for Mobile Workers

The mobile worker solution is made up of three major components: components located at the mobile worker's remote site; corporate components, and optional IP telephony and other services; which are sometimes embedded into the worker's laptop via soft phones and other applications.

The required home office components are broadband access (cable, satellite, or DSL), remote VPN router with QoS functionality, and laptop or desktop. The optional components include IP phone, wireless LAN (WLAN) access point, and Cisco video telephony (VT) camera. However, mobile workers may want to connect from multiple locations, and in some of those locations a laptop is all they have. The rest of the infrastructure is facilitated by an increasingly welcoming business and public infrastructure that provides broadband access and basic wireless connectivity.

Corporate components are a VPN headend router or a multifunction security appliance such as the Cisco Adaptive Security Appliance (ASA), authentication, and central management devices for resilient aggregation and termination of the IPsec and SSL VPN tunnels.

The optional IP telephony components are Cisco Unified CallManager for call processing, signaling, and device control; voice gateway for interconnection of traditional phone networks with VoIP environments; IP phones for voice and added-value services; and a voice messaging platform for diverse message consolidation. And, do not forget, Cisco Contact Center for advanced call treatment.

The traditional mobile worker solution is based on a software VPN client on the remote user laptop or desktop PC. This setup is characterized by these drawbacks:

- Lower level of accessibility (for example, the inability to deploy and support advanced applications, such as voice, video, and videoconferencing)
- No QoS for efficient delivery and prioritization of traffic
- Inadequate security, with security relying on the end user (with IT maintaining no control)
- Absence of controlled configuration, management, and support by IT (and end users having to perform these functions)

The business-ready mobile worker solution overcomes all the weak points of the traditional mobile worker solution. Business-Ready model replaces the traditional model for mobile worker connectivity.

#### Business-Ready Mobile Worker and VPN Options

The business-ready mobile worker solution is driven by the explosion in broadband access, especially wireless connectivity in recent years. From Wi-Fi to WiMAX to 3G and 4G efforts on the cellular wireless spectrum, more and more mobile workers can connect to corporate resources. As workers roam across wireless networks, their connectivity might be affected because of automatic IP address changes, performed by DHCP or different IP address pools. Roaming transparency is required to make transitions smooth and allow for uninterrupted connectivity across wireless cells, even inside the organization campus.

With the advent of SSL VPNs, the mobile worker obtains access from unmanaged devices and from a multitude of locations. This results in a variety of security concerns. The likelihood of connecting from vulnerable or even already infected machine makes integrated endpoint security one of the most important features of any mobile worker solution. Cisco's NAC (Network Admission Control) overcomes this issue.

Central management of the mobile worker and his components is also crucial. The challenge becomes providing support to potentially thousands of users connecting from all over the world 24x7. Cisco efforts such as Easy VPN and the Cisco Security Manager (CSM) allow for central policy management and automated deployment of those policies. A reliable architecture complements the solution, typically using multiple VPN concentrators acting as headends and providing redundancy and resiliency.

Ultimately, mobility solutions must contemplate all the resources and conditions the mobile worker finds at the physical office. The tools of the trade for this virtual office must include the security, management, flexibility, and transparency features we have discussed.

When we consider remote-access and mobile worker solutions, IPsec and SSL VPNs become clear options. Table 7-12 presents a condensed view of the characteristics of both IPsec VPN and SSL VPN solutions.

Table 7-12. Characteristics of IPsec and SSL VPN Solutions

| IPsec VPN                                                                     | SSL VPN                                                                                       |
|-------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Is a client-based technology                                                  | Is a clientless technology—reduces operations costs associated with managing client software  |
| Provides full tunnel access, with all applications supported                  | Provides optional full tunnel access, but may suffer from application interoperability issues |
| Uses IPsec—a well-proven technology                                           | Uses SSL—a well-proven technology                                                             |
| Operates well for extending access to employees using company-managed devices | Simplifies business partner access                                                            |
| —                                                                             | Provides “anywhere” access from unmanaged devices                                             |
| —                                                                             | Enables customized access portals                                                             |

The transparency of IPsec VPNs in terms of application support is offset by the need for client software installed at the connecting machine. SSL VPNs provide clientless technology, if we consider the web browser application to be ubiquitous and already installed in the most popular laptop and desktop operating systems. This comes at the expense of application support, as some nonweb applications may require additional components to the solution. When looked at from the security perspective, both technologies allow access into mission critical assets, and both require integrated endpoint security controls. SSL VPNs may need to enforce more stringent security policies, because they are typically used to connect from nonmanaged devices.

Ultimately, both technologies will integrate into your remote-access solution, to provide effective connectivity for both managed and unmanaged devices.

### Routing Traffic to the Mobile Worker

Now that you have learned about the components and technologies used by mobile workers, this section describes the preparation and configuration necessary to provide them access to the corporate network. This section shows you, using a simple scenario, the tasks for the implementation plan, focusing on the configuration of network elements such as routers and switches.

In a real-world scenario, deploying remote-access networks for mobile worker connectivity requires careful planning. If we look at the overall, end-to-end solution, we must consider several steps and service areas in building such solution. Perhaps our first decision is the choice of VPN solutions. For business-ready teleworkers, this will probably be an IPsec site-to-site VPN, from their home office router into the headend VPN router. For mobile, traveling users, SSL VPNs are gaining ground as the solution of choice.

Any remote-access solution must consider security and access control. VPNs provide confidentiality, integrity, and authentication, but the private network will require access control and threat mitigation. So from identity management via AAA services, to firewall deployment, to intrusion prevention technologies, our mobile worker solution must consider these services as part of the overall security policy.

The remote teleworker will require local services that facilitate the VPN connection and enable productivity. Mobility through wireless, along with the support services encompassing wireless solutions, is key. This includes wireless security through authentication and encryption, such as using 802.1X.

In most cases, the remote user will enhance productivity by using data, voice, presence, and multimedia services. QoS becomes crucial, and must be carefully planned. It is also important to provision all of these

services to be automated, centralized, and policy based. The Cisco Easy VPN solution centralizes policy in the headend device, such as a Cisco VPN router or the ASA firewall, and pushes those policies to connecting clients at the moment of connection.

The following components are required to provide remote access to mobile workers:

- VPN router (for example, Cisco Easy VPN server)
- Mobile worker device (for example, Cisco Easy VPN client)
- IPsec VPN tunnel
- Internet connectivity

Now let's take a look at how we can configure the VPN router and the mobile worker device and the necessary IPsec VPN tunnel, starting with the VPN router configuration.

#### VPN Headend Configuration

The headend VPN router is also known as the Easy VPN server in Easy VPN terminology. It concentrates the bulk of the remote-end configuration, which "pushes" the policies to the client at the moment of connection. The remote end, the device used by the mobile worker, is known in Easy VPN terminology as the Easy VPN remote or Easy VPN client. The Easy VPN remote device starts an IPsec VPN tunnel to connect to the Easy VPN server across the public network.

The following steps are required to configure a router as an Easy VPN server:

- Step 1. Allow IPsec traffic.
- Step 2. Define an address pool for connecting clients.
- Step 3. Provide routing services for VPN subnets.
- Step 4. Tune NAT for VPN traffic flows.
- Step 5. Verify IPsec VPN configuration.

We first need to allow IPsec traffic through the Internet-facing device, such as a perimeter router or a firewall. It is also possible that our headend VPN router is the Internet-facing device.

To provide VPN services, we will define a pool of addresses that will provide routing and addressing for remote users. We will define address pools for connecting clients in the private network. The address pools defined in this step will be used as part of the VPN configuration.

Next, we will confirm that the address pools configured in Step 2 are known across the network, and therefore proper routing of remote clients' return traffic is assured. This is sometimes a hidden and obscure part of the configuration, because it is often assumed that the VPN architecture will provide routing to its subnets.

Then, we will tune NAT so that VPN traffic bypasses that function and it is not translated. As mentioned earlier, one of the functions of IPsec is to tunnel the traffic. The traffic is tunneled into a new IP header expressly so the original header is not changed.

The last step is to verify VPN configuration and operations, using verification commands that will help us identify the main components and proper operation of the VPN service.

The following subsection cover, in detail, each VPN router configuration step. Remember that our network security team has already configured the routers for VPN access. It is our responsibility as the routing team to check that it works. This verification will be done with **show** commands, and with tools such as ping and trace. When required, we will amend the routing configuration of the router, but will not touch its security configuration.

#### Allowing IPsec Traffic

Our first step is to make sure we are allowing IPsec traffic in our VPN router. This router is probably an Internet edge network element. It is in an ideal position to provide access control services through firewalling. Most likely, this router is running some sort of firewall service, or at least ACLs to implement antispoofing mechanisms and other security controls.

To check whether ACLs are filtering traffic on the router, use the **show ip interface** command and the **show access-lists** commands.

IPsec traffic needs to be allowed through those security controls on the Internet-facing interface so that the router can, in fact, terminate IPsec VPN tunnels. There are different types of Cisco IOS firewalls:

- A classic firewall is based on ACLs, and is the more traditional approach to firewalling in a Cisco IOS router. This classical approach is referred to context-based access control (CBAC).



- A zone-based firewall (ZBF) is based on security zones and access, and it is the more recent approach to implementing the service in routers. It is important to identify which type of firewall we have, to determine the component we need to change to allow IPsec traffic through the firewall.

The `show ip inspect` command gives you the details on the classic firewall. A subset of the `show ip inspect` command is explained in Table 7-13.

Table 7-13. *show ip inspect* Command

| Parameter                          | Description                                                                                    |
|------------------------------------|------------------------------------------------------------------------------------------------|
| <b>name</b> <i>inspection-name</i> | Displays the configured inspection rule with the name <i>inspection-name</i>                   |
| <b>interfaces</b>                  | Displays the interface configuration with respect to applied inspection rules and access lists |

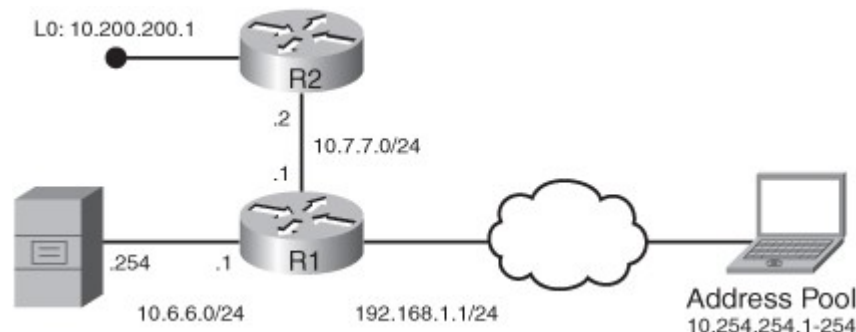
Use the `show zone-pair security` command, shown in Table 7-14, to display information about the zone-based firewall.

Table 7-14. *show zone-pair security* Command

| Parameter                                       | Description                             |
|-------------------------------------------------|-----------------------------------------|
| <b>source</b> <i>source-zone-name</i>           | (Optional) Name of the source zone      |
| <b>destination</b> <i>destination-zone-name</i> | (Optional) Name of the destination zone |

Figure 7-25 illustrates the commands discussed in Table 7-13 and Table 7-14.

Figure 7-25. VPN Router Topology Example.



Example 7-55 shows the result of the `show ip interface fa0/1` command, that there is an inbound access list called FIREWALL-INBOUND applied to interface Fa0/1.

Example 7-55. *show ip interface* Command on R1

```
R1#show ip interface fa0/1
<output omitted>
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: 224.0.0.10
Outgoing access list is not set
Inbound access list is FIREWALL-INBOUND
Proxy ARP is enabled
Local Proxy ARP is disabled
```

```
Security level is default
Split horizon is enabled
ICMP redirects are always sent
R1#
```

Now, using the `show access-lists` command, shown in Example 7-56, we will find the details of the FIREWALL-INBOUND access list applied on interface fa0/1 of R1.

Example 7-56. *show access-lists* Command on R1

```
R1#show access-lists
Extended IP access list FIREWALL-INBOUND
 10 permit eigrp any any (1452 matches)
 20 permit tcp any any eq telnet
 30 permit icmp any any (20 matches)
 40 permit tcp any host 192.168.1.10 eq www
 50 permit tcp any host 192.168.1.10 eq ftp
 60 permit udp any any eq domain
Extended IP access list NAT-ACL
 10 permit ip 10.6.6.0 0.0.0.255 any (2 matches)
Extended IP access list SPLIT
 10 permit ip 10.6.6.0 0.0.0.255 172.31.7.0 0.0.0.255
 20 deny ip any any
R1#
```

The access list called FIREWALL-INBOUND, currently configured in R1, could be part of a bigger firewalling strategy, and therefore we need to investigate further whether our IOS router is configured to act as a firewall.

Using the `show ip inspect interfaces` command, shown in Example 7-57, we find out that we have a classic firewall configured inbound on R1. We can also see which access lists are involved in the access control process, so we can quickly make a note and proceed to change the ACLs to allow IPsec traffic. In Example 7-57, the access list is conveniently called FIREWALL-INBOUND, which we looked at earlier.

Example 7-57. *show ip inspect interfaces* Command on R1

```
R1#show ip inspect interfaces
Interface Configuration
Interface FastEthernet0/0
Inbound inspection rule is INSPECTION
 tcp alert is on audit-trail is off timeout 3600
 udp alert is on audit-trail is off timeout 30
 icmp alert is on audit-trail is off timeout 10
Outgoing inspection rule is not set
Inbound access list is ALL
Outgoing access list is not set
Interface FastEthernet0/1
Inbound inspection rule is INSPECTION
 tcp alert is on audit-trail is off timeout 3600
 udp alert is on audit-trail is off timeout 30
 icmp alert is on audit-trail is off timeout 10
Outgoing inspection rule is not set
Inbound access list is FIREWALL-INBOUND
Outgoing access list is not set
R1#
```

If we attempt to do the `show zone-pair security` command on R1, we will see that zone-based firewall has not been configured, as shown in Example 7-58.

Example 7-58. `show zone-pair security` Command on R1

```
R1#show zone-pair
security
R1#
```

#### Note

It is possible to have both ZBF and CBAC configured on the same router but on a different interface. An interface cannot be configured with CBAC and ZBF at the same time.

Now that we know what kind of firewall we are dealing with for R1, let's add the IPsec support to the ACL. IPsec uses ESP to provide confidentiality through encryption. ESP, found at Layer 4 of the OSI model, uses protocol 50. IPsec can also AH if only integrity is required. AH uses protocol 51.

#### Note

Many publications refer to ESP as using port 50 and AH as using port 51. This is wrong, because at Layer 4 we are referring to protocols not ports, such as in TCP being protocol 6 and UDP being protocol 17. See <http://www.iana.org/assignments/protocol-numbers/> for confirmation.

During the first stage of IPsec, peer negotiations and credentials are exchanged using a protocol called ISAKMP, UDP port 500 (sometimes erroneously called IKE, or Internet Key Exchange). ISAKMP is one of three components that make up IKE. Finally, UDP 4500 will need to be opened for NAT Traversal (NAT-T), another IPsec service. To recapitulate, here are the additional configurations to be added to the ACLs to prepare for VPN tunnels:

- Protocol 50 (ESP)
- Protocol 51 (AH)
- UDP port 500 (ISAKMP)
- UDP port 4500 (NAT-T)

Example 7-59 shows how to add those lines to our access list.

Example 7-59. Modifying the ACL on R1 to Support IPsec Protocols

```
R1(config)#ip access-list extended FIREWALL-
INBOUND
R1(config-ext-nacl)#4 permit 50 any any
R1(config-ext-nacl)#5 permit 51 any any
R1(config-ext-nacl)#6 permit udp any any eq 500
R1(config-ext-nacl)#7 permit udp any any eq 4500
```

#### Defining Address Pools

The second step is to define IP address pools for the remote connecting clients. One might wonder why we need address pools for these VPN users. Of course, they already have an IP address to start with, which allows them to connect to the IP network, but we need to provide for the fact that we are using IPsec VPNs. With IPsec tunnels, IPsec VPNs encapsulate original traffic within an additional packet, to allow that private traffic to be routed across a public network.

But this is a virtual *private* network, so ultimately traffic needs to go between a private host and a private resource. The private host happens to be virtual in a VPN; it is really located outside of the private network. The encapsulation process will use private addressing in the original (encapsulated) packet and public addressing for the "outer" (encapsulation) packet. The private address of the connecting host is typically assigned dynamically for flexibility reasons.

The address pool should be planned carefully. It should be unused in the network and follow best practices in address assignment. For instance, you should define it to be a summarizable address block, so that it can later be easily and efficiently identified in access lists, or properly summarized by routing protocols.

The address pool is created locally at the VPN router, and later assigned to the VPN configuration.

The local pool at the router level is commonly used as a configuration alternative. Other options include DHCP servers sitting behind the router and the VPN client requesting the private address via DHCP. The router, at that point, acts as a DHCP relay agent. AAA servers, such as the Cisco Access Control Server (ACS), can also be used to provide a per-user or per-group address pool.

Use the `ip local pool poolname first-address—last-address global` command, described in Table 7-15, to create address pools.

Table 7-15. *ip local pool* Command

| Parameter                                       | Description                             |
|-------------------------------------------------|-----------------------------------------|
| <b>source</b> <i>source-zone-name</i>           | (Optional) Name of the source zone      |
| <b>destination</b> <i>destination-zone-name</i> | (Optional) Name of the destination zone |

Example 7-60 shows how to configure the address pool in R1. The global command used is `ip local pool`, as explained in Table 7-15. The pool must be given a name that will be assigned later to the IPsec VPN configuration. Notice that under the Easy VPN framework you could have a separate local pool for different user profiles using the IPsec VPN. This configuration granularity allows for even more flexibility in differentiating traffic from each user profile. In addition, you can apply different path, routing, and access policies to each profile.

Example 7-60. *ip local pool* Command on R1

```
R1#config t
Enter configuration commands, one per line. End with CNTL/Z
R1(config)#ip local pool EZVPN 10.254.254.1
10.254.254.254
R1(config)#
```

In Example 7-60, the address pool receives the name EZVPN. The range of addresses for the pool is 10.254.254.1 to 10.254.254.254. In this example, we allocated numbers that are easy to identify so that administrators can quickly know that this is the remote-access users when they see source addresses within this range when looking at various screen outputs such as routing tables, access lists, or in parts of the network configuration.

#### Providing Routing Services for VPN Subnets

The third step is to provide effective routing services so that traffic coming from VPN clients can reach internal resources and the return traffic can find its way back to those remote users.

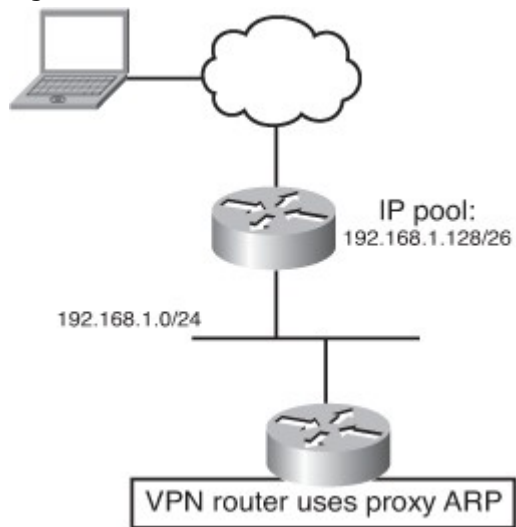
It is worth saying that the VPN subnets, defined by the IP address pools allocated for remote-access clients, are ephemeral. They appear and disappear as VPN clients connect and disconnect. They must be known, though, to all routers in the internal network, or at least along the path of traffic flows that VPN clients will use.

Several methods, including the following, can be used to make those address pools known to routers in the internal network:

- Proxy ARP
- Reverse route injection
- Static routes with redistribution

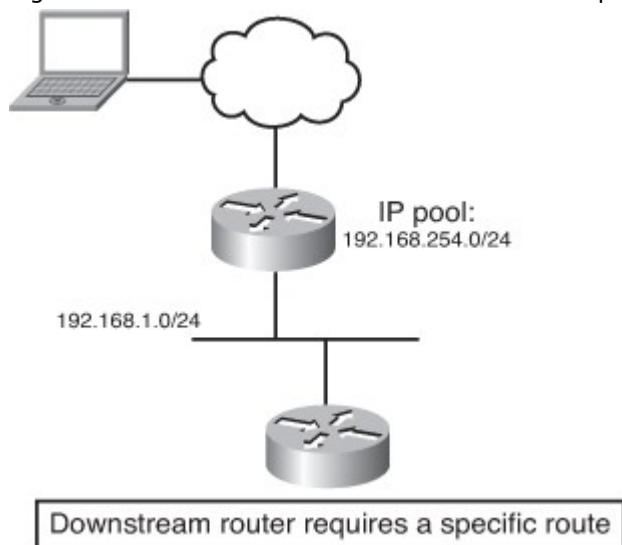
Proxy ARP is one of the simplest methods. It involves selecting the address pool as a subnet of an existing physical segment. An example of this method, shown in Figure 7-26, would consist of using for remote users the subnet 192.168.1.128/26 when the internal network is using subnet number 192.168.1.0/24.

Figure 7-26. Address Pool of Remote Users Is a Subnet of the Internal Network.



One advantage of the method shown in Figure 7-26 is that you do not have to use additional subnets. Another advantage is that no changes are required to routers or routing protocols, because they already know this network. This reduces the complexity of the configuration. However, you must enable Proxy ARP on the VPN router's interface connected to the internal segment containing the borrowed subnet. Other methods are more dynamic but are also more complex. They will designate a separate, nonoverlapping subnet to be the address pool for remote-access VPN clients, as shown in Figure 7-27. This would require advertising that subnet using routing protocols.

Figure 7-27. Address Pool of Remote Users Is Separate from the Internal Subnet Address.



A commonly used dynamic approach is to use a method known as reverse route injection (RRI). RRI is the ability for static routes to be automatically inserted into the routing process for those networks and hosts that are protected by a remote tunnel endpoint. This method injects the client's host address in the routing table with the next hop to this network being the remote tunnel endpoint. After the static route is created on the VPN router, this information is propagated to upstream devices, allowing them to determine the appropriate VPN router to which to send returning traffic to maintain IPsec state flows. This "dynamic" injection happens only when a client is connected. If the client disconnects, the entry is removed from the routing table. RRI is an IPsec feature, configured within crypto map statements. The configuration of crypto map statements is beyond the scope of this book.

Finally, another way to provide routing services to remote users is a hybrid solution using static and dynamic features. This is achieved by creating a static route pointing to the remote-access address pool and then

redistributing that particular static route into your routing protocol. The commands used are `ip route` and `redistribute static metric {metric_value}`, as described in Table 7-16.

Table 7-16. *redistribute static* Command

| Parameter                         | Description                                                                                                                                                                                                                                                                     |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>metric</b> <i>metric_value</i> | An optional parameter that specifies the EIGRP base metric, also referred to as the seed metric, in the order of bandwidth, delay, reliability, load, and maximum transmission unit, for the redistributed route, such as <b>redistribute static metric 3000 100 255 1 1500</b> |

The configuration shown in Example 7-61 is a simple implementation of the commands appearing in Table 7-16.

Example 7-61. Configuring Static Route and Redistribution

Code View: Scroll / Show All

```
R1(config)#ip route 10.254.254.0 255.255.255.0
192.168.1.2
R1(config)#
R1(config)#do show ip protocols
Routing Protocol is "eigrp 1"
 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 metric weight K1=1, K2=0, K3=1, K4=0, K5=0
 EIGRP maximum hopcount 100
 EIGRP maximum metric variance 1
 Redistributing: eigrp 1
 EIGRP NSF-aware route hold timer is 240s
 Automatic network summarization is not in effect
 Maximum path: 4
 Routing for Networks:
 10.6.6.0 255.255.255.0
 10.7.7.0 255.255.255.0
 192.168.1.0
 Passive Interface(s):
 FastEthernet0/0
 Routing Information Sources:
 Gateway Distance Last Update
 10.5.7.10 90 3d03h
 10.7.7.2 90 04:55:02
 10.2.7.1 90 3d03h
 10.3.7.1 90 3d00h

R1(config)#router eigrp 1
R1(config-router)#redistribute static
R1(config-router)#
```

In the example, we create a static route using the IP route 10.254.254.0 255.255.255.0 209.165.200.2. Notice that our static route points to R1 as the next hop, which is 209.165.200.2. This next hop is responsible for initiating and terminating VPN tunnels. We also use the **show ip protocols** command to confirm which routing protocol is already in use on R1. We discover that we are using EIGRP. Therefore, within EIGRP, we redistribute the static route. It is best practice to use route filters to ensure that only the desired routes are redistributed. It is also recommended that an appropriate metric be applied to these redistributed routes.

To verify proper redistribution, we look at the routing table of R2, shown in Example 7-62. We can see that R2 is aware of the remote-access VPN subnet, 10.254.254.0/24. As soon as our VPN clients connect to our corporate network, R2 will be able to route traffic back to them.

Example 7-62. Looking at the R2 Routing Table Following Redistribution of the Static Route

Code View: Scroll / Show All

R2#**sh ip route**

Codes: 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

Gateway of last resort is not set

172.31.0.0/24 is subnetted, 1 subnets

D 172.31.7.0 [90/20517120] via 10.7.7.1, 04:57:56, Serial2/5

10.0.0.0/24 is subnetted, 5 subnets

D EX 10.254.254.0 [170/20514560] via 10.7.7.1, 00:00:29, Serial2/5

C 10.200.200.0 is directly connected, Loopback0

C 10.7.7.0 is directly connected, Serial2/5

D 10.6.6.0 [90/20514560] via 10.7.7.1, 04:57:21, Serial2/5

C 192.168.1.0 is directly connected, FastEthernet0/0

D 209.165.200.0/24 [90/20514560] via 10.7.7.1, 04:53:46, Serial2/5

R2#

Tuning NAT for VPN Traffic Flows

We will now turn our attention to tuning NAT for proper VPN operations. The tuning is often necessary because

- The VPN router might be an Internet edge router providing NAT services.
- VPN traffic need not be translated. It remains private when tunneled across the VPN.
- A packet is processed through the NAT engine before it is forwarded to the IPsec engine.
- NAT should be bypassed for VPN traffic because mobile worker VPN traffic is easily identified. It uses the VPN address pool as a destination.

Recall that IPsec encapsulates packets that use private addresses within a larger packet that uses public addressing to be routed across a public network. This action creates the IPsec tunnel.

The edge router shown in Figure 7-25 is providing NAT services for Internet traffic to acquire a public address. You can use the **show ip nat statistics** command, shown in Example 7-63, to verify this, and we confirm that R1 is configured for NAT with its Fa0/0 and S0/0/0 interfaces configured as inside NAT and its Fa0/1 configured as outside NAT. Also, notice the ACL used by the NAT rule. The NAT-ACL access list is defining which source address will have the privilege of being translated by an address found in the NAT pool named NAT-POOL.



Example 7-63. *show ip nat statistics* on R1

```
R1#show ip nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)
Outside interfaces:
 FastEthernet0/1
Inside interfaces:
 FastEthernet0/1, Serial0/0/0
Hits: 20 Misses: 0
CEF Translated packets: 10, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list NAT-ACL pool NAT-POOL refcount 0
 pool NAT-POOL: netmask 255.255.255.0
 start 172.16.250.1 end 172.16.250.1
 type generic, total addresses 1, allocated 0 (0%), misses 0
Queued Packets: 0
R1#
```

When we think about it, VPN traffic is not really Internet traffic. It happens to be private traffic using the Internet as a medium. But the traffic is not destined for an Internet, public site. The VPN-bound traffic, generated by our users, does not require a public address because it is intended to have a destination in the private network. Therefore, the traffic exchanged through the VPN tunnel, between our mobile worker and some internal corporate resources, should not be translated.

If we consider a router, packets are processed in a certain order, according to the services running on that router. NAT is processed before outbound IPsec encryption, so VPN traffic would be translated before it is sent through the tunnel. But as discussed in the preceding paragraph, we do not want translation to be done on the VPN traffic. Therefore, the router needs to be tuned for VPN traffic to bypass the NAT function. This is accomplished using an extended ACL. All other traffic that does not meet the criteria of the extended ACL will be processed by NAT.

NAT tuning is achieved by using either access lists or route map commands to define the traffic that will bypass translation. In any case, whether we use access lists or route maps, a **permit** line means "translate" and a **deny** line means "do not translate."

Use the **access-list** command to define the addresses that will be subjected to translation in the **route-map** command. We will then need to apply the route map to the **ip nat inside source** command.

Example 7-64 demonstrates how an ACL and a route map are used to identify the traffic that will not be translated. We first create the access list, referencing the IP address pool as a destination address.

First, using an extended ACL, numbered 100, we define that traffic originating from any IP address, but with a destination of 10.254.254.0/24, addresses of our remote users, will be denied translation. We also add a second line to the ACL, specifying that all other IP traffic will be subjected to translation. The command is `access-list 100 deny ip any 10.254.254.0 0.0.0.255`. The VPN pool address is set as a destination, because the NAT configuration is relative to the view from the router, whose main job is to route packets, and therefore for the router, the VPN clients are from the outside interface, so they are destinations. An additional line is required: `access-list 100 permit ip any any`. Only VPN destinations should bypass translation. All other Internet-bound traffic must be translated. In Example 7-64, the route map called **VPNTRAFFIC** is set to match traffic according to an ACL 100, using the `route-map VPNTRAFFIC` command.

The next step is to apply the route map to the NAT configuration using the **ip nat inside source route-map VPNTRAFFIC pool NAT-POOL overload** command.

Example 7-64. Configuring VPN User Traffic to Bypass Address Translation

```
Code View: Scroll / Show All
R1#config t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#access-list 100 deny ip any 10.254.254.0 0.0.0.255
```

```
R1(config)#access-list 100 permit ip any any
R1(config)#route-map VPNTRAFFIC permit 10
R1(config-route-map)#match ip address 100
R1(config-route-map)#exit
R1(config)# ip nat inside source route-map VPNTRAFFIC pool NAT-POOL overload
```

#### Verifying IPsec VPN Configuration

The last step is to verify the existence and proper operations of the IPsec VPN. This will be achieved by having a look at the main configuration components and some commands to verify that the VPN is working. There are three generic steps to provision a router for VPN connectivity:

- Step 1.      Configure IPsec settings.
- Step 2.      Group setting together into a crypto map.
- Step 3.      Apply the crypto map to an interface.

As you can see, one of the main components of IPsec VPNs is the crypto map. Crypto maps are configuration objects that group and gather the VPN settings to apply them to an interface. Only then, when the crypto map is applied to router interfaces, will the router start encrypting and tunneling traffic through those interfaces. A detailed look at crypto maps and IPsec configurations is beyond the scope of this routing book.

Broadly explained, crypto maps have three components:

- Crypto ACL
- Security policies
- SA parameters

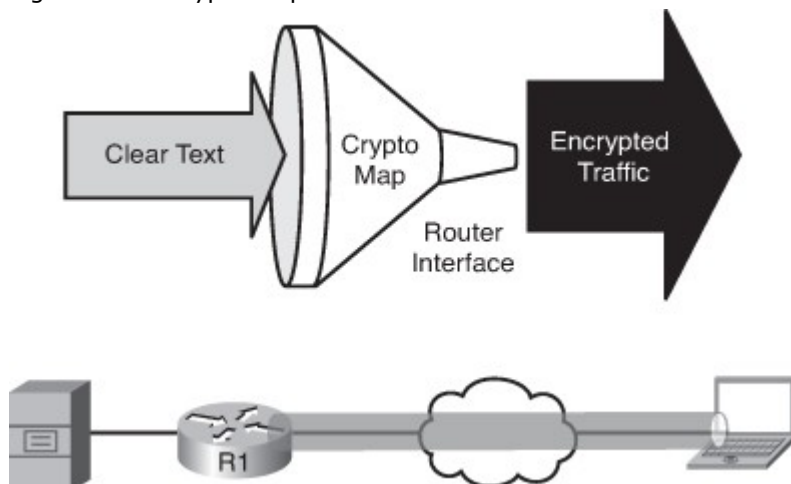
A crypto map uses ACLs to define the type of traffic to encrypt. A **permit** statement in a crypto ACL will result in the traffic being encrypted, whereas **deny** statement will result in the traffic being sent out by the Internet-facing interface in clear text. Examples of encryption algorithms that would be used for encrypting traffic are DES, 3DES, and AES.

Notice that with the remote worker crypto map components, we did not include the VPN peer address as we did for the branch office deployment. The reason is that crypto maps cannot use the VPN peer address to associate the proper crypto map to a peer, because by definition, mobile workers are connecting from different locations and therefore do not always use the same IP address. In this situation, the crypto map is referred to as a dynamic crypto map, a concept that is beyond the scope of this book.

Traffic defined by the ACL used by the crypto map will be encrypted according to configured cryptographic algorithms and policies, called security policies. Security policies are therefore also part of the crypto map. Finally, additional parameters, called security association (SA) parameters, such as tunnel lifetime in terms of time and volume, become part of the overall configuration of crypto maps.

Recall that crypto maps operate like a funnel, as shown in Figure 7-28.

Figure 7-28. Crypto Map Funnel.



This funnel channels all configuration components to the appropriate interface, serving as initiation and termination point of IPsec tunnels. You configure the IPsec settings, group them together in a crypto map, and then apply the crypto map to the interface. When traffic meets the criteria, it passes through the funnel, and its policies are enforced. Traffic that does not meet the criteria configured in the crypto maps leaves the Internet-facing interface unencrypted.

This crypto map-to-interface association will indicate where initiation and termination of the tunnel occur, thus identifying the interface requiring additional configuration for VPN to work, such as allowing dynamic routing and tunnel address translations. Crypto maps are applied to the VPN router interface facing the public.

To verify if the VPN configuration is functioning properly, use the **show crypto map** command, the **show crypto isakmp sa** command, the **show crypto sa** command, and the **show crypto engine connections active** command. It is also important to test the full connectivity, by having a remote user attempting to connect. Use the commands as follows:

- Use the **show crypto map** command to display the crypto map configuration.
- Use the **show crypto isakmp sa** command to display all current IKE SAs at a VPN peer. This command has no arguments or keywords.
- Use the **show crypto ipsec sa** command to display the settings used by current SAs.
- Use the **show crypto engine connections active** command to view the current active encrypted session connections for all crypto engines.

Example 7-65, displays the result of executing the `show crypto map` command on R1. R1 output indicates that the interface to which the crypto map is applied is Fa0/1. This information tells us that the IPsec VPN tunnels will terminate on interface Fa0/1.

Example 7-65. Example of the `show crypto map` Command

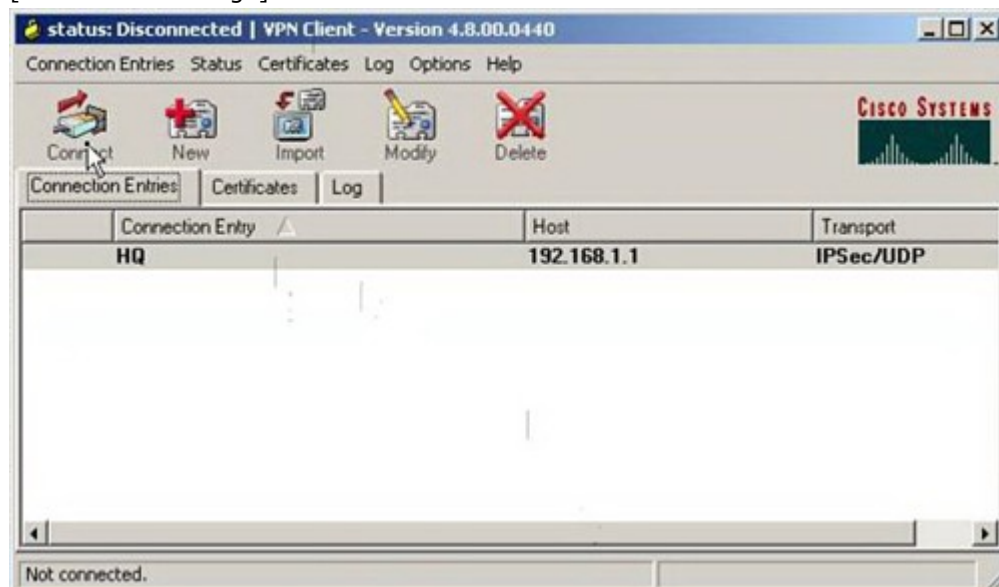
```
R1#show crypto map
Crypto Map "RAVPN" 10 ipsec-isakmp
 Dynamic map template tag: DYNO
 Interfaces using crypto map RAVPN:
 FastEthernet0/1
R1#
```

Having a crypto map assigned to an interface is a good start but does not prove that IPsec connectivity is fully functional. To be able to fully confirm the proper configuration and working of an IPsec VPN for mobile workers, it is imperative to have a remote user test the connection.

In our example, the VPN tunnel will be initiated by a remote user, using the Cisco VPN Client. The main window of the Cisco VPN Client, already installed on the mobile worker's computer, is depicted in Figure 7-29. The user clicks the Connect button after highlighting the HQ connection entry. This particular HQ connection entry was created ahead of time, providing it with the parameters to establish the IPsec connection, among which is the IP address of the headend VPN device, the Internet-facing interface of R1 at 209.165.200.1.

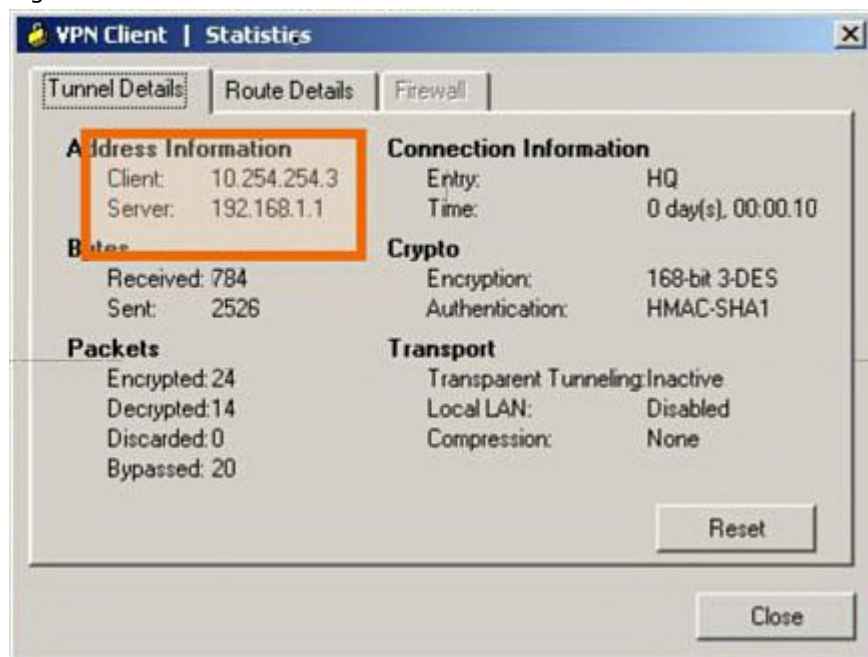
Figure 7-29. Cisco VPN Client.

[View full size image]



After a brief moment of negotiation between the client's software and the headend VPN router, the user will be asked to authenticate. After providing a successful authentication, the users will be connected on the corporate network via IPsec VPN. To confirm connectivity and that packets are passing through the VPN tunnel, the user can pull down the Status menu and select Statistics. Then, the user can click the Tunnel Details tab. The VPN Client will then show IP security information, listing the IPsec statistics for this VPN tunnel to the private network, as shown in Figure 7-30.

Figure 7-30. Cisco VPN Client Statistics.



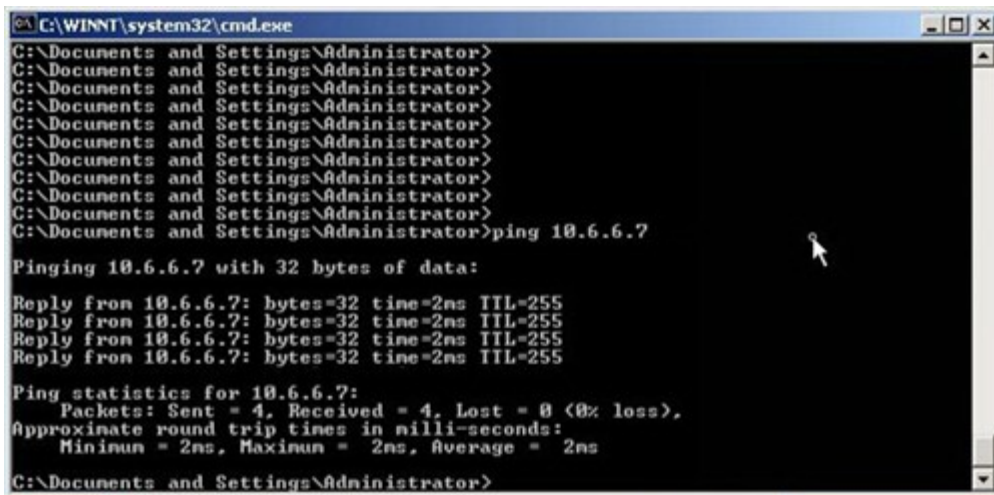
Referring to Figure 7-30, we can confirm that the client is connected to the VPN server at address 192.168.1.1. The client received the IP address of 10.254.254.3, which belongs to the VPN pool. Moreover, we are encrypting and decrypting traffic.

The user will then confirm full connectivity with the corporate network by trying to reach internal resources using ping. In Figure 7-31, the VPN user is successful at using ping against an internal resource 10.6.6.7. At this point, the remote user could go back to the statistics windows of the client software and would notice

that the encrypted and decrypted packets counters would have increased in value compared to the statistics shown in Figure 7-30.

Figure 7-31. VPN Client Ping an Internal Destination.

[View full size image]



```
C:\WINNT\system32\cmd.exe
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>
C:\Documents and Settings\Administrator>ping 10.6.6.7

Pinging 10.6.6.7 with 32 bytes of data:
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255

Ping statistics for 10.6.6.7:
 Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
 Approximate round trip times in milli-seconds:
 Minimum = 2ms, Maximum = 2ms, Average = 2ms

C:\Documents and Settings\Administrator>
```

An effective way to confirm IPsec connectivity, from the VPN router's perspective, is to use the `show crypto isakmp sa` command. In Example 7-66, we see that R1 has one active tunnel, which initiated from 172.31.7.10 to terminate on 192.168.1.1, which is the Internet-facing interface of R1. Although not shown here, the `show crypto ipsec sa` command could also be used to confirm remote connectivity.

Example 7-66. `show crypto isakmp sa` Command

```
R1#show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst src state conn-id slot status
192.168.1.1 172.31.7.10 QM_IDLE 1012 0 ACTIVE

IPv6 Crypto ISAKMP SA

R1#
```

Example 7-67 shows the output of another useful command to check connectivity from the VPN router's perspective. The `show crypto session` command enables you to confirm that there is an IPsec tunnel established from any source address going to the destination of 10.254.254.3 which is the remote user's VPN address.

Example 7-67. `show crypto session` Command

```
R1#show crypto session
Crypto session current status

Interface: FastEthernet0/1
Session status: UP-ACTIVE
Peer: 172.31.7.10 port 500
IKE SA: local 192.168.1.1/500 remote 172.31.7.10/500 Active
IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 host 10.254.254.3
 Active SAs: 2, origin: dynamic crypto map

R1#
```

Earlier, in Figure 7-30, you saw how the client can see the statistics on the packets being encrypted and decrypted. Similar statistics can be seen from the VPN router by using the `show crypto engine connections active` command shown in Example 7-68.

Example 7-68. `show crypto engine connections active` Command

```
R1#show crypto engine connections active
Crypto Engine Connections

 ID Interface Type Algorithm Encrypt Decrypt IP-Address

1012 Fa0/1 IKE SHA+3DES 0 0 192.168.1.1
2013 Fa0/1 IPsec 3DES+SHA 0 56 192.168.1.1
2014 Fa0/1 IPsec 3DES+SHA 42 0 192.168.1.1

R1#
```

We also need to confirm that packets reaching further down in the corporate network can be routed back to the router user. We would like to test full connectivity between the remote users and the loopback address of R2. Success of this step would prove that proper routing is taking place on the network, where R2 can make a routing decision for destination address of 10.254.254.0/24. Figure 7-32 shows the successful result of a ping from the remote host to IP address 10.200.200.1.

Figure 7-32. Successful Ping Between Remote Users and R2's Loopback Interface.

[View full size image]

```
C:\WINNT\system32\cmd.exe
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255
Reply from 10.6.6.7: bytes=32 time=2ms TTL=255

Ping statistics for 10.6.6.7:
 Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
 Approximate round trip times in milli-seconds:
 Minimum = 2ms, Maximum = 2ms, Average = 2ms

C:\Documents and Settings\Administrator>ping 10.200.200.1

Pinging 10.200.200.1 with 32 bytes of data:

Reply from 10.200.200.1: bytes=32 time=21ms TTL=254
Reply from 10.200.200.1: bytes=32 time=20ms TTL=254
Reply from 10.200.200.1: bytes=32 time=20ms TTL=254
Reply from 10.200.200.1: bytes=32 time=21ms TTL=254

Ping statistics for 10.200.200.1:
 Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
 Approximate round trip times in milli-seconds:
 Minimum = 20ms, Maximum = 21ms, Average = 20ms

C:\Documents and Settings\Administrator>
```

We have thus proven full connectivity from both the remote user's end and the VPN router. Moreover, we have proven that our remote user can reach resources further down in the corporate network, thus confirming that proper routing is taking place.

Reviewing Alternatives for Mobile Worker Connectivity

In the previous sections, you saw how to provide mobile worker connectivity from the perspective of routing facilities. The functionality we implemented permitted the routing of VPN address pools. We also tuned NAT so that remote user traffic would not be translated. We proved the proper functioning by having our remote user reach the remote loopback address of R2. In addition, the `show crypto isakmp sa` command showed us that the VPN tunnel with the remote was up.

Our goal was not to know the details of IPsec functioning but rather to know how to fine-tune the related components, such as ACLs. We also discussed the routing options, and you learned that the most dynamic option is reverse route injection. Our configuration in the previous section followed a less-complex but more-static configuration of using static route and then redistributing it through the EIGRP domain. Example 7-69 displays the static route and the VPN pool called EZVPN, which was defined in Example 7-60.

Example 7-69. Routing Table on the Branch router with the Remote VPN Client

Code View: Scroll / Show All



```

R1#show ip route static
 10.0.0.0 255.255.255.0 is subnetted, 4 subnets
S 10.254.254.0 [1/0] via 192.168.1.2
R1# show ip local pool

Pool Begin End Free In use
EZVPN 10.254.254.1 10.254.254.254 253 1
R1#

```

Table 7-17 gives a summary of the alternative solutions for routing when dealing with remote workers. Example 7-70 shows a brief example of the preferred method of RRI, without getting into too many details of the commands.

Table 7-17. Alternative Solutions for Routing Remote Workers

| Solution                                       | Advantages                                                             | Disadvantages                                                                      |
|------------------------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------------------|
| Use existing subnet                            | Simple. Saves IP address space. Does not require routing adjustment.   | Requires Proxy ARP on VPN router                                                   |
| Use new subnet                                 | Address pool easily identified, potentially one pool per user profile. | Uses the IP address space, requires dynamic routing or static routes               |
| Advertise using static routes and redistribute | Simple.                                                                | Not dynamic                                                                        |
| Advertise using RRI                            | VPN client routes automatically inserted in routing table.             | Extra processing on routing table and routing process if connections too ephemeral |

In Example 7-70, we remove the static routing we saw Example 7-69. We then configure our crypto dynamic map to perform RRI, with the reverse-route command. We confirm with the show ip route static command that the static route no longer exists.

We will then bring up a connection from our remote host to R1.

Once the remote VPN user is connected, we reissue the **show ip route static** command to the router and witness that a static entry was automatically introduced in the routing table. The moment the remote client disconnects, the static route to 10.254.254.4 will disappear from the routing table.

Example 7-70. Removing Static Routing and Configuring RRI

```

R1#config t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#no ip route 10.254.254.0 255.255.255.0
R1(config)#
R1(config)#crypto dynamic-map DYNO 10
R1(config-crypto-map)#reverse-route
R1(config-crypto-map)#do show ip route static

R1(config-crypto-map)#

<VPN users remotely connects>

```



```
R1(config-crypto-map)#do show ip route static
10.0.0.0 255.0.0.0 is variably subnetted, 4 subnets, 2 masks
S 10.254.254.4 255.255.255.255 [1/0] via 172.31.7.10
R1(config-crypto-map)#
```

One drawback of RRI is that entries are added with a host mask, a mask of 32 bits in the routing table. If you have hundreds of remote clients that connect simultaneously, that could result in a very long routing table.

You have now learned to use basic IPsec verification commands to monitor the health of connections. Those verifications were not thorough and detailed, but they will prove useful as you move forward and learn more about VPN connectivity and maintenance.

In most cases, the simplest routing solution is the most suitable. As you saw, static routes worked well in the scenario in the previous section. You also saw that when you are establishing a VPN tunnel, traffic between the remote host and the LAN resources should not be subjected to NAT. This is not an issue because the original packets will be encrypted into a newer packet by the IPsec process.

## Summary

In this chapter, you learned about implementing routing facilities for branch offices and mobile workers, including the following:

- Planning the branch office implementation
- Analyzing services in the branch office
- Planning for mobile worker implementations
- Routing traffic to the mobile worker

More precisely, you learned about the following:

- Branch office design considerations, such as connectivity technologies, resiliency, routing protocols, service mix, security and compliance, mobility.
- DSL broadband technologies, which are not complete end-to-end solutions, and therefore protocols such as PPPoE and PPPoA are used. You also saw a generic PPPoA configuration in this chapter.
- Configuring static routing. Using the **show ip protocols** command to check routing protocol configuration and exchanges. More precisely, for floating static route, using the **ip route prefix mask next-hop-ip-address distance** command
- Configuring NAT and tuning it, including the use of the following commands:
  - The **ip nat pool name start-ip end-ip {netmask netmask | prefix-length prefix-length}** command to define a pool of IP addresses for NAT.
  - The **ip nat inside source {list {access-list-number | access-list-name} | route-map name} {interface type number | pool name} [overload]** command to link the source IP address to the pool for translated address. If static translation is used, the command is **ip nat inside source {static {local-ip global-ip} [vrf name] [extendable] [no-alias] [no-payload] [route-map] [redundancy group-name] | {esplocal-ip interface type number}}**.
  - The **ip nat inside** and **ip nat outside** interface configuration commands are used to designate that traffic originating from or destined for a specific interface is subject to NAT.
  - The **show ip nat translations** command used to display active NAT translation and the **clear ip nat statistics** command to reset the counters to zero.
- The chapter briefly covered how HSRP can be used to decide to switch to another active router upon failure and how it would affect the traffic flow.
- The four options to route dynamic routing protocols through an IPsec tunnel:
  - Point-to-point generic routing encapsulation (P2P GRE)
  - Virtual tunnel interface (VTI)
  - Dynamic multipoint VPN (DMVPN)
  - Group encrypted transport VPN (GET VPN)
- IPsec, its major services of confidentiality, integrity, and authentication, and the benefits of encryption and encapsulation.

- Protocol and port numbers used by IPsec: ESP (protocol 50), AH (protocol 51), IKE/ISAKMP (port UDP 500), and NAT-T (port UDP 4500).
- Components of crypto maps (crypto ACL, remote peer IP address, security policies, and security association parameters), and how to verify the configuration with the **show crypto map** command, the **debug crypto ipsec** command, and the **show crypto session** command.
- How the **deny** statement in a NAT ACL is not for blocking the traffic but rather specifying that translation is not to be performed on that traffic. Similarly, in a crypto ACL, the **deny** statement is not for blocking the traffic but rather specifies that encryption is not to be performed on that traffic.
- Impact of NAT and IPsec, and how the NAT process takes place before the encryption process, and how the NAT ACL will need to be tuned with the **access-list** configuration command. You also learned how to verify whether translation was successful by using the **show ip nat statistics** command, **debug ip nat** command, and the **show ip access-lists name** command.
- Impact of IPsec on dynamic routing, because it does not carry multicast or broadcast, and how GRE can be used to circumvent that problem.
- GRE is a tunneling protocol developed by Cisco, capable of encapsulating a variety of network layer protocols packets inside IP tunnels, which makes it a common option to pass dynamic routing protocol traffic across an IPsec tunnel.
- GRE configuration steps are as follows:
  - Create tunnel interfaces for GRE with the **tunnel source {ip-address | interface-type interface-number}** command, and define the origin of the tunnel with **tunnel source {ip-address | interface-type interface-number}** and the destination of the tunnel with the **tunnel destination ip-address** command. The **tunnel mode gre ip** command was not required in our scenario because the default of encapsulation of GRE is IP.
  - Configure routing protocols to route through the GRE tunnel, with the **network subnet mask** command.
  - Change the crypto ACL to encrypt GRE traffic.
- There are two types of Cisco IOS firewalls: Classic firewalls are based on ACLs, and zone-based firewalls are based on security zones.
- To check and provide traffic routing to mobile workers, there are five generic steps:
  - Allow IPsec traffic.
    - Use the **show ip inspect** command to get the details on the classic firewall and the **show zone-pair security** command to display information on the zone-based firewall.
  - Define the address pool.
    - Use the **ip local pool poolname first-address—last-address** global command to create address pools.
    - Provide routing services with methods such as Proxy ARP, RRI, and static route redistribution.
    - Create a static route pointing to the remote access address pool with the **ip route prefix mask {ip-address | interface-type interface-number [ip-address]}** and redistribute it with the **redistribute static metric {metric\_value}** command.
  - Tune NAT for VPN flows.
    - Use the **access-list access-list-number {deny | permit} protocol source source-wildcard destination destination-wildcard** command to define the addresses that will be subjected to NAT in the **route-map map-tag [permit | deny] [sequence-number]** command.
    - Apply the route-map to the **ip nat inside source {list {access-list-number | access-list-name} | route-map name} {interface type number | pool name} [overload]** command.
  - Verify IPsec VPN configuration:
    - Use the **show crypto map** command, the **show crypto isakmp sa** command, the **show crypto ipsec sa** command, and the **show crypto engine connections active** command to verify the VPN configuration and its proper functioning.
    - Use the **show crypto map** command to display the crypto map configuration.
    - Use the **show crypto isakmp sa** command to display all current IKE SAs at a VPN peer.

- Use the **show crypto ipsec sa** command to display the settings used by current SAs.
- Use the **show crypto engine connections active** command to view the current active encrypted session connections for all crypto engines.

## References

For additional information, see these resources:

- “Cisco IOS Software Releases 12.4 Mainline” support page—  
[http://www.cisco.com/en/US/products/ps6350/tsd\\_products\\_support\\_series\\_home.html](http://www.cisco.com/en/US/products/ps6350/tsd_products_support_series_home.html)
- The Cisco IOS Command Reference  
at [http://www.cisco.com/en/US/products/ps6350/prod\\_command\\_reference\\_list.html](http://www.cisco.com/en/US/products/ps6350/prod_command_reference_list.html)
- Paquet, C. *Implementing Cisco IOS Network Security (IINS)—(CCNA Security exam 640-553) (Authorized Self-Study Guide)*. Cisco Press, 2009.

## Review Questions

Answer the following questions, and then see Appendix A, “Answers to Review Questions,” for the answers.

- List some considerations of branch office design.
- List some services that may be deployed on branch routers in a branch office design.
- When issuing the **show crypto map** command, you notice a transform set called VPNPOLICY. What is the purpose of a transform set?
- Indicate for each statement which IPsec service is being provided.
 

| Statement                                                                     | IPsec service |
|-------------------------------------------------------------------------------|---------------|
| Provides assurance that the data is exchanged with the rightful party         |               |
| Provides encryption during the exchange of the data                           |               |
| Provides a check to confirm that the data was not altered during transmission |               |
- Which of the following statements are true?
  - A **deny** statement in a NAT ACL blocks the traffic.
  - A **permit** statement in a crypto ACL encrypts the data.
  - A GRE tunnel destination must be a physical interface on the peer router.
  - RRI is a good substitute to redistribute static routes for remote users.
  - When VPN tunnels are used, the branch office LAN should be configured with the same subnet number as the one used on the headquarters LAN.
- In the **ip route prefix mask next-hop-ip-address distance** command, what is the *distance* parameter used for and how should it be set up for a floating static route?
- Fill in the blank: \_\_\_\_\_ uses a frequency range from approximately 20 KHz to 1 MHz.
- There are two types of Cisco IOS firewalls. What are they?
- You notice in the output of the **show crypto session detail** command the following line: IPSEC FLOW: permit 47. What does it mean?
- Select the ports and protocols that should be allowed on the inbound ACL applied to the Internet-facing interface if you want to terminate IPsec VPN tunnels?
  - Port UDP 51
  - Protocol 51
  - Protocol 4500
  - Port UDP 500
  - Port UDP 4500
  - Port TCP 50
  - Port TCP 500
  - Protocol 50
  - Port TCP 4500
- Write the command to create an address pool that will be used for incoming VPN session. The pool

has the name EZVPN with the address range 10.254.254.1 to 10.254.254.254.

12. Write the command to provide static routing to the next hop for the EZVPN pool configured in Question 11. The next-hop IP address is 209.165.200.2.
13. What would the command be to redistribute into EIGRP the static route created in Question 12?
14. In a NAT scenario, what command is used to link the source IP address to the pool for a translated address?
15. Which of the following are alternatives to GRE for IPsec tunnels?
  - a. IPS
  - b. GET VPN
  - c. HSRP
  - d. DMVPN
  - e. Zone-base firewalling
16. In which order should you configure a router as an Easy VPN server?  
Verify IPsec VPN configuration.  
Allow IPsec traffic.  
Tune NAT for VPN traffic flows.  
Provide routing services for VPN subnets.  
Define an address pool for connecting clients.
17. Where could a VPN client go to see that the VPN client is indeed connected and exchanging packets with the headend router?
18. Which **show** command could be issued to the headend VPN router to see that an ISAKMP security association has been established and the IP address of the remote host?
19. What is the purpose of the **ip nat pool name start-ip end-ip {netmask netmask | prefix-length prefix-length}[type rotary]** command?
20. While preparing to configure a branch and headquarters office for IPsec VPN connectivity, why is it a good idea to issue the **show ip dhcp pool [name]** command and maybe even the **show ip dhcp binding {ip-address}** command and the **show ip dhcp server statistics** command?

## Chapter 8. Implementing IPv6 in an Enterprise Network

This chapter introduces IPv6 and covers the following topics:

- Introducing IPv6
- IPv6 Addressing
- Configuring and Verifying IPv6 Unicast Addresses
- Routing IPv6 Traffic
- Transitioning IPv4 to IPv6
- Tunneling IPv6 Traffic
- Translation Using NAT-PT

IP Version 4 (IPv4) allows end systems to communicate and forms the foundation of the Internet as we know it today. However, IPv4 has many limitations. IP Version 6 (IPv6) is a technology developed to overcome these limitations.

This chapter introduces IPv6 and the IPv6 addressing scheme and explores how IPv6 addresses are configured. Routing protocols that support IPv6 are explored in detail, as is IPv6 policy routing and route redistribution. The chapter concludes with a discussion of how IPv4 networks can be transitioned to IPv6, including dual-stack, tunneling, and translation techniques.

### Introducing IPv6

The ability to scale networks for future demands requires a limitless supply of IP addresses and improved mobility. IPv6 combines expanded addressing with a more efficient and feature-rich header to meet these demands. While it has many similarities to IPv4, IPv6 satisfies the increasingly complex requirements of hierarchical addressing that IPv4 does not support.

### Note

IPv6 is defined in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification*.

This section reviews IPv4 issues that led to the IPv6 design, introduces the key features and benefits of IPv6, and describes the IPv6 packet header. The Cisco IOS supports IPv6 in Release 12.2(2)T and later.

### IPv4 Issues

As mentioned, IPv6 was designed to overcome issues associated with IPv4, including the following:

- **Address depletion**—One of the major shortcomings of IPv4 is its limited amount of address space. The explosion of new IP-enabled devices, the growth of undeveloped regions, and the rapid growth of other regions have fueled the need for more addresses. In the United States, the Department of Defense (DoD) is a primary driver for the adoption of IPv6 and had set a date of 2008 for all systems to be migrated to this new standard. However, this migration has really only just started.
  - Of course, all other industries are potential IPv6 users, including education, government agencies, enterprises, service providers, home networks, consumer appliances, distributed online gaming, voice services, and wireless services.
  - The number of Internet users is growing extremely fast; there were approximately 973 million users as of November 2005, and 1.7 billion by the end of 2009. The Internet will be transformed after IPv6 fully replaces its less-versatile parent years from now. Nevertheless, IPv4 is in no danger of disappearing overnight. Rather, it will coexist with and then gradually be replaced by IPv6. This change has already begun, particularly in Europe, Japan, and the Asia Pacific. These areas of the world are exhausting their allotted IPv4 addresses, which makes IPv6 all the more attractive. As noted, in addition to its technical and business potential, IPv6 offers a virtually unlimited supply of IP addresses, enough to allocate more than the entire IPv4 Internet address space to everyone on the planet.

- **Internet routing table expansion**—As more nodes come online, the Internet routing tables continue to grow. These large tables require a lot of processing power, memory, and overhead when stored within the Internet routers.

Note

According to the CIDR Report (<http://www.cidr-report.org/as2.0/>), there were over 310,000 active Border Gateway Protocol (BGP) entries in Internet routers at the beginning of 2010. This has grown from 200,000 at the beginning of 2007.

- **Lack of true end-to-end model**—IPv4 networks typically use Network Address Translation (NAT) as the solution to address depletion. However, NAT hides the true source address of traffic, which can cause other issues. (NAT does, however, provide some security advantages because it provides a layer of anonymity.)

As of January 2010, only 10 percent of the IPv4 addresses remain unallocated. As mentioned, the U.S. DoD is migrating to IPv6, but in North America there is not the sense of urgency felt in other parts of the world, because North American organizations have many IPv4 addresses allocated to them. Governments and organizations in other parts of the world do not have the luxury of having as many IPv4 addresses, and are thus making the transition to IPv6 more quickly.

Features of IPv6

IPv6 is a powerful enhancement to IPv4 with features that better suit current and foreseeable network demands, including the following:

- **Larger address space**— IPv6 addresses are 128 bits, compared to IPv4's 32 bits. This larger address space provides several benefits, including improved global reachability and flexibility, the ability to aggregate prefixes that are announced in routing tables, and easier multihoming to several Internet service providers (ISPs).
- **Elimination of public-to-private NAT**— The larger address space allows end-to-end communication without NAT.
- **Elimination of broadcast addresses**— IPv6 includes unicast addresses (one to one), multicast addresses (one to many), and a new type called anycast addresses (one to nearest). IPv6 does not have broadcast addresses.
- **Simplified header for improved router efficiency**— A simpler header provides several advantages over IPv4, including improved routing efficiency for performance and forwarding-rate scalability, no requirement for processing checksums, simpler and more efficient extension header mechanisms, and flow labels for per-flow processing with no need to examine the transport layer information to identify the various traffic flows.
- **Support for mobility and security**— Mobility and security help ensure compliance with mobile IP and IPsec standards.

Mobility enables people to move around in networks with mobile network devices, with many having wireless connectivity. Mobile IP is an Internet Engineering Task Force (IETF) standard available for both IPv4 and IPv6 that enables mobile devices to move without breaks in established network connections. Because IPv4 does not automatically provide this kind of mobility, supporting it requires additional configurations.

In IPv6, mobility is built in, which means that any IPv6 node can use it when necessary. The routing headers of IPv6 make mobile IPv6 much more efficient for end nodes than mobile IPv4 does.

IPsec is the IETF standard for IP network security, available for both IPv4 and IPv6. Although the functions are essentially identical in both environments, IPsec support is mandatory in IPv6. IPsec is enabled and is available for use on every IPv6 node, making the IPv6 Internet more secure. IPsec may use keys for each device, which implies global key deployment and distribution.

- **Transition richness**— There are a variety of ways to transition IPv4 to IPv6.

One approach is to have a dual stack, with both IPv4 and IPv6 configured on the interface of a network device. In this scenario, both IPv4 and IPv6 run simultaneously.

Another technique uses an IPv4 tunnel to carry IPv6 traffic. Implementations include IPv6-to-IPv4 (6to4) tunneling and IPv4-compatible tunneling.

Cisco IOS Software Release 12.3(2)T (and later) also allows NAT protocol translation (NAT-PT) between IPv6 and IPv4, providing direct communication between hosts that are using the different protocol suites.

Fortunately the transition to IPv6 from IPv4 does not have to happen overnight; rather, it can be a gradual change. For example, enterprises can keep their internal networks running IPv4 while simultaneously

supporting both protocols on their external web servers. As more applications and web sites move to supporting IPv6, enterprises can gradually transition their internal networks, until they are completely IPv6. Many organizations that have a large number of users, and many countries, including those in the Asia Pacific, have already started aggressively adopting IPv6, because they are running out of IPv4 addresses. For example, the IPv6 Promotion Council of Japan includes many working groups such as those for security and certification of IPv6 products. In China, IPv6 was used for some of the systems at the 2008 Olympics.

IPv6 addresses also provide flexibility not found in IPv4, including the following features:

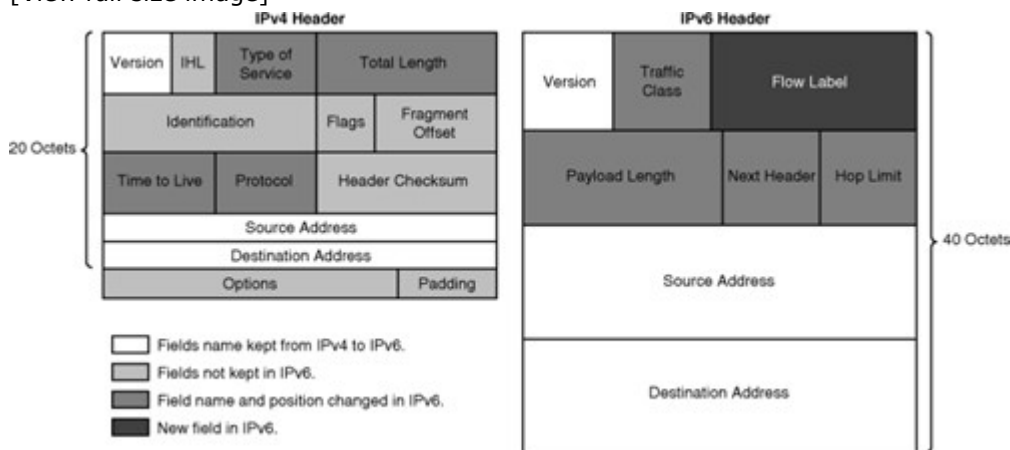
- **Stateless autoconfiguration**— IPv6 provides address autoconfiguration that includes the device's data link layer address in the IPv6 address for "plug-and-play" functionality. (A version of the stateful dynamic host control protocol [DHCP] is also provided; the DHCP server keeps track of the state of the DHCP client.)
- **Prefix renumbering**— IPv6 allows simplified mechanisms for address and prefix renumbering. The router advertises the new prefix and the other devices on the network can begin to use the new information. This process can be made nondisruptive to the hosts on the network by manipulating valid and preferred timers for each IPv6 address.
- **Multiple addresses per interface**— IPv6 interfaces can have multiple addresses of various types assigned to them; these addresses can be used simultaneously.
- **Link-local addresses**— IPv6 devices automatically create a link-local address on each interface; these addresses are used for many purposes. For example, the interior gateway protocols (IGPs) use a link-local address as the next hop when they are exchanging routing updates.
- **Provider-dependent or provider-independent addressing**— Because there are so many addresses available, enterprises can use addresses from an ISP, or use their own provider-independent addressing space.

#### IPv6 Packet Header

As shown in Figure 8-1, the IPv6 header has 40 octets, in contrast to the 20 octets in the IPv4 header. IPv6 has fewer fields, and the header is 64-bit aligned to enable fast, efficient, hardware-based processing. The IPv6 address fields are four times larger than in IPv4.

Figure 8-1. IPv4 and IPv6 Headers.

[View full size image]



The IPv4 header contains 12 basic header fields, followed by an options field and a data portion (which usually includes a transport layer segment). The basic IPv4 header has a fixed size of 20 octets, and the variable-length options field increases the size of the total IP header. IPv6 contains fields similar to 7 of the 12 IPv4 basic header fields (5 plus the source and destination address fields), but does not require the other fields.

The IPv6 header contains the following fields:

- **Version**— A 4-bit field, the same as in IPv4. For IPv6, this field contains the number 6. For IPv4, this field contains the number 4.



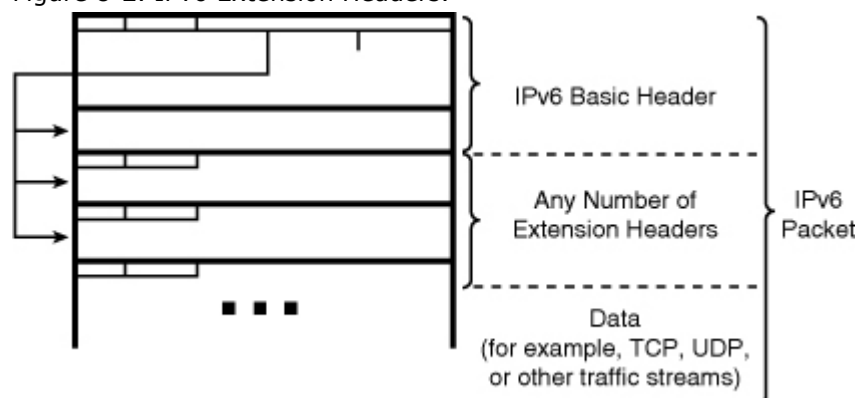
- **Traffic class**— An 8-bit field similar to the Type of Service (ToS) field in IPv4. This field tags the packet with a traffic class that it uses in differentiated services (DiffServ) quality of service (QoS). These functionalities are the same for IPv6 and IPv4.
- **Flow label**— This 20-bit field is new in IPv6. It can be used by the source of the packet to tag the packet as being part of a specific flow, allowing multilayer switches and routers to handle traffic on a per-flow basis rather than per-packet, for faster packet-switching performance. This field can also be used to provide QoS.
- **Payload length**— This 16-bit field, allowing payloads of up to 64 kilobytes (kB), is similar to the IPv4 total length field.
- **Next header**— The value of this 8-bit field determines the type of information that follows the basic IPv6 header. It can be a transport-layer packet, such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP), or it can be an extension header. The next header field is similar to the protocol field of IPv4.
- **Hop limit**— This 8-bit field specifies the maximum number of hops that an IP packet can traverse. Similar to the Time To Live (TTL) field in IPv4, each router decreases this field by one. Because there is no checksum in the IPv6 header, an IPv6 router can decrease the field without recomputing the checksum, whereas in IPv4 routers, the recomputation uses processing time. If this field ever reaches zero, a message is sent back to the source of the packet and the packet is discarded.
- **Source address**— This field has 16 octets or 128 bits. It identifies the source of the packet.
- **Destination address**— This field has 16 octets or 128 bits. It identifies the destination of the packet.
- **Extension headers**— The extension headers, if any, and the data portion of the packet follow the other eight fields. The Next header field defines the type of the next extension header. The number of extension headers is not fixed, so the total length of the extension header chain is variable.

Notice that the IPv6 header does not have a header checksum field. Because link-layer technologies perform checksum and error control and are considered relatively reliable, an IP header checksum is considered to be redundant. Without the IP header checksum, upper-layer checksums, such as within UDP, are mandatory with IPv6.

#### Extension Headers

IPv6 has extension headers that handle options more efficiently and enable a faster forwarding rate and faster processing by end-nodes. The next header field points to the next header in the chain, as shown in Figure 8-2.

Figure 8-2. IPv6 Extension Headers.



Most extension headers are not examined or processed by any node other than the node to which the packet is destined. The destination node examines the first extension header (if there is one). The contents of an extension header determine whether or not the node should examine the next header. Therefore, extension headers must be processed in the order they appear in the packet.

There are many types of extension headers. Only a hop-by-hop options header, if it is present, must be examined by every node along the path. This hop-by-hop options header, if present, must immediately follow the IPv6 header, and is indicated by a value of zero in the next-header field.

When multiple extension headers are used in the same packet, the order of the headers in the chain should be as follows:

1. **IPv6 header:** This is the basic IPv6 header.

2. **Hop-by-hop options header:** When this header is used, it is processed by all hops (routers) in the path of the packet. Example uses are for a Router Alert, including for Resource Reservation Protocol (RSVP) and Multicast Listener Discovery (MLD) messages (as defined in RFC 2711, *IPv6 Router Alert Option*), and for IPv6 Jumbograms (as defined in RFC 2675, *IPv6 Jumbograms*).
3. **Destination options header (when a routing header is used):** This header (with a next-header value = 60) follows any hop-by-hop options header, in which case the destination options header is processed at the final destination and also at each destination specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header, in which case the destination options header is processed only at the final destination. Mobile IPv6 is an example of when this header is used. (The destination options header should occur at most twice, once before a routing header and once before the upper-layer header).
4. **Routing header:** This header (with a next-header value = 43) is used for source routing and mobile IPv6. An IPv6 source lists one or more intermediate nodes that are to be visited on the way to a packet's destination in this header.
5. **Fragment header:** This header (with a next-header value = 44) is used when a source must fragment a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination device. The fragment header is used in each fragmented packet. (This header cannot be used in combination with the jumbo payload hop-by-hop option.)
6. **Authentication header and Encapsulating Security Payload header:** The Authentication Header (AH) (with a next-header value = 51) and the ESP header (with a next-header value = 50) are used within IPsec to provide authentication, integrity, and confidentiality of a packet. These headers are identical for both IPv4 and IPv6.
7. **Upper-layer header:** The upper-layer (transport) headers are the typical headers used inside a packet to transport the data. The two main transport protocols are TCP (with a next-header value = 6) and UDP (with a next-header value = 17).

#### MTU Discovery

In IPv4, routers handle fragmentation, causing a variety of processing issues.

IPv6 routers no longer perform fragmentation; instead, a discovery process is used by the source IPv6 device to determine the optimum MTU to use during a given session. In this discovery process, the source IPv6 device attempts to send a packet at the size that is specified by the upper IP layers, for example, the transport and application layers. If the source IPv6 device receives an Internet Control Message Protocol for IPv6 (ICMPv6) "packet too big" message, it retransmits the MTU discover packet with a smaller MTU. This process is repeated until the device receives a response that the discover packet arrived intact. The device then sets the MTU for the session. (Note that the device may still fragment IPv6 packets, as noted in the earlier discussion of fragment header.)

The ICMPv6 "packet too big" message contains the proper MTU size for the path. Each source device tracks the MTU size for each session. Generally, the tracking is done by creating a cache based on the destination address; however, it can also be done by using the flow label. Alternatively, if source-based routing is performed, the tracking of the MTU size can be done by using the source address.

The discovery process is beneficial because, as routing paths change, a new MTU might be more appropriate. When a source device receives an ICMPv6 "packet too big" message, it decreases its MTU size if the ICMPv6 message contains a recommended MTU that is less than the current MTU of the device. Devices perform an MTU discovery every 5 minutes to see whether the MTU has increased along the path. Application and transport layers for IPv6 accept MTU reduction notifications from the IPv6 layer. As discussed, if for some reason these upper layers do not accept the notifications, IPv6 has a mechanism for source devices to fragment packets that are too large. However, upper layers are encouraged to avoid sending messages that require fragmentation.

#### IPv6 Addressing

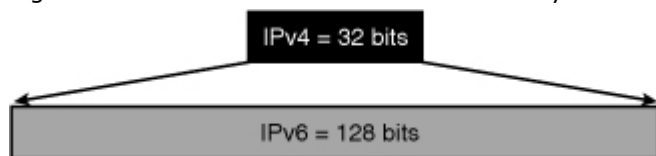
This section explores the IPv6 addressing in an enterprise, IPv6 address representation, interface identifiers, address types, unicast addresses, multicast addresses, anycast addresses, and compares IPv6 addresses with IPv4 addresses.

##### IPv6 Addressing in an Enterprise Network

IPv6 increases the number of address bits by a factor of 4, from 32 to 128, providing a very large number of addressable nodes.

Figure 8-3 shows the increased number of address bits. However, as in any addressing scheme, not all the addresses are used or available.

Figure 8-3. IPv6 Provides Four Times as Many Address Bits as IPv4.

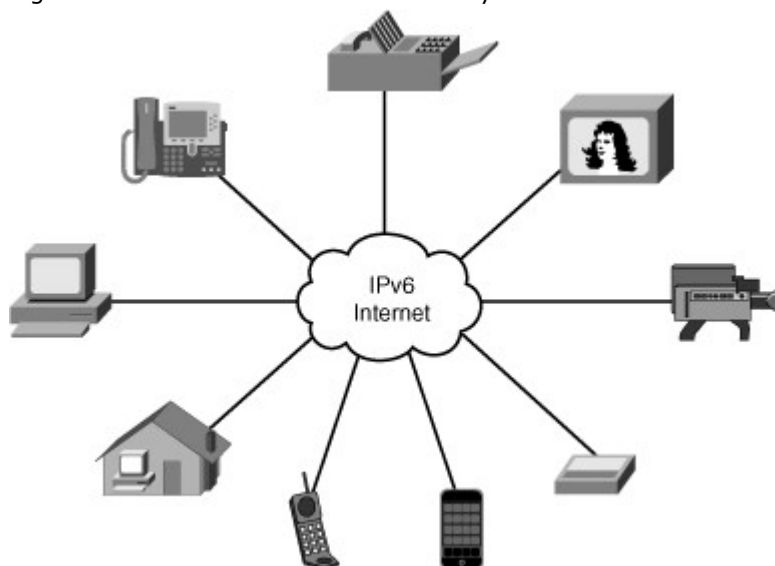


With 32 bits, IPv4 allows for approximately 4,200,000,000 possible addressable nodes, with some 2 billion usable addresses. Current IPv4 address use is extended by applying techniques such as private-to-public address space NAT and temporary address allocations (such as addresses leased by DHCP). However, the manipulation of the packet by intermediate devices complicates the advantages of peer-to-peer communication, end-to-end security, and QoS.

In contrast, the 128 bits in an IPv6 address allow for approximately  $3.4 \times 10^{38}$  possible addressable nodes, which works out to approximately  $5 \times 10^{28}$  addresses for every person on our planet!

Therefore, IPv6 has enough address space such that every user could have multiple global addresses that can be used for a wide variety of devices, such as wireless IP phones, entertainment set-top boxes, video equipment, security surveillance equipment, and so on, as illustrated in Figure 8-4. In the future other devices may be added, as technologies and requirements change. These device addresses would be reachable without using IP address translation, pooling, or temporary allocation techniques.

Figure 8-4. IPv6 Can Interconnect Many Different Devices.



Note, however, that increasing the number of bits for the address also increases the IPv6 header size. Because each IP header contains a source address and a destination address, the size of the header fields that contains the addresses is 256 bits for IPv6 compared to 64 bits for IPv4. (The IPv6 header is described in detail in the "IPv6 Packet Header" section, earlier in this chapter.)

IPv6's larger address spaces allow for sizable address allocations to ISPs and organizations. An ISP can aggregate all the prefixes of its customers into a single prefix and announce the single prefix to the IPv6 Internet. Aggregation of customer prefixes results in an efficient and scalable routing table, which is necessary for broader adoption of network functions. The increased address space is also sufficient to allow organizations to define a single prefix for their entire network.

#### IPv6 Address Representation

Rather than using dotted-decimal format, IPv6 addresses are written as hexadecimal numbers with colons between each set of four hexadecimal digits (which is 16 bits); we like to call this the "coloned hex" format. The format is  $x:x:x:x:x:x:x:x$ , where  $x$  is a 16-bit hexadecimal field; each  $x$  is therefore four hexadecimal digits. (Note that there are seven colons separating eight sections; each section has four hexadecimal digits.) An example address is as follows:

2035:0001:2BC5:0000:0000:087C:0000:000A

#### Note

The hexadecimal digits A, B, C, D, E, and F in IPv6 addresses are not case sensitive.

Fortunately, you can shorten the written form of IPv6 addresses, in the following ways:

- Leading 0s within each set of four hexadecimal digits can be omitted.
- A pair of colons (::) can be used, only once within an address, to represent any number ("a bunch") of successive 0s. An address parser identifies the number of missing 0s by separating the two parts of the address and entering 0s until it has 128 bits. If two (or more) :: notations were to be placed in the address, there would be no way to identify the size of each "bunch" of 0s.

For example, the previous address can be shortened to the following:

2035:1:2BC5::87C:0:A

Note that we used the :: only once within the address and we shortened the last set of 0000 to a single 0.

The following are some example IPv6 addresses and their shortened notation:

- The shortest IPv6 address is an all-0s address, 0:0:0:0:0:0:0:0 (which can be written as ::).
- 0:0:0:0:0:0:1 can be written as ::1.
- FF01:0:0:0:0:0:1 can be written as FF01::1.
- 2031:0000:130F:0000:0000:09C0:876A:130B can be written as 2031:0:130F::9C0:876A:130B. However, 2031::130F:9C0:876A:130B is not a valid address because it uses two sets of ::.

Similar to how IPv4 subnet masks can be written as a prefix (for example, /24), IPv6 uses prefixes to indicate the number of bits of network or subnet. Prefix notation is also known as classless interdomain routing (CIDR) notation.

Of course, Cisco routers allow you to enter IPv6 addresses using the shortened format, and using the prefix. For example, Example 8-1 illustrates the configuration of the shortened IPv6 address discussed earlier, on a loopback interface. The prefix /64 is used on this interface. (Notice the similarity of the command to configure an IPv6 address to the command to configure an IPv4 address.)

Note

The syntax of the IPv6 commands are detailed later in the chapter in the "Configuring and Verifying IPv6 Unicast Addresses" section, but some examples are provided here to illustrate various concepts.

Example 8-1. When Configuring an IPv6 Address You Can Use the Shortened Form and the Prefix

```
R1(config)#interface loopback 100
R1(config-if)#ipv6 address 2035:1:2BC5::87C:0:A?
WORD X:X:X:X::X X:X:X:X::X/<0-128>
R1(config-if)#ipv6 address
2035:1:2BC5::87C:0:A/64
R1(config-if)#
```

#### Interface Identifiers in IPv6 Addresses

In IPv6, a link is defined as a network medium over which network nodes communicate using the link layer. Interface identifiers (IDs) in IPv6 addresses are used to identify a unique interface on a link. They may also be thought of as the "host portion" of an IPv6 address. Interface IDs are required to be unique on a link, and may also be unique over a broader scope. When the interface identifier is derived directly from the data link layer address of the interface, the scope of that identifier is assumed to be universal (global).

Interface identifiers are always 64 bits and may be dynamically created based on Layer 2 media and encapsulation.

IPv6 is defined on most of the current data link layers, including those shown in Table 8-1.

Table 8-1. Data-Link Layers that Support IPv6

Ethernet[1]

Point-to-Point Protocol (PPP)[1]

High-Level Data Link Control (HDLC)[1]

Fiber Distributed Data Interface (FDDI)

Token Ring

Attached Resource Computer Network (ARCNET)

Nonbroadcast multiaccess (NBMA)

Asynchronous Transfer Mode (ATM)[2]

Frame Relay[3]

IEEE 1394[4]

[1] Data-link layers supported by Cisco.

[2] Cisco supports only ATM permanent virtual circuit (PVC) and ATM LAN Emulation (LANE).

[3] Cisco supports only Frame Relay PVCs.

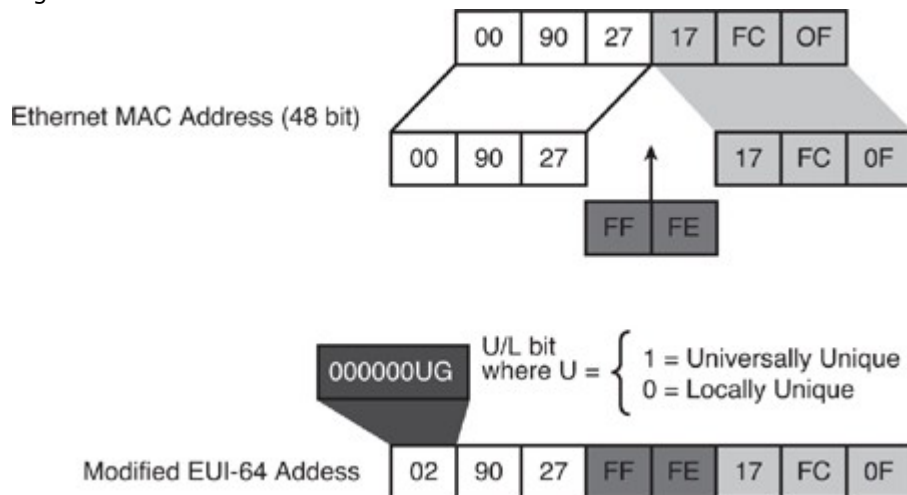
[4] A Standard for a High Performance Serial Bus, supporting data rates of up to 800 Mbps (in IEEE 1394b).

The data link layer defines how IPv6 interface identifiers are created and how neighbor discovery deals with data link layer address resolution. RFCs describe the behavior of IPv6 in each of these specific data link layers, but the Cisco IOS Software does not necessarily support all of them.

For Ethernet, the interface ID used is based on the Media Access Control (MAC) address of the interface and is in an extended unique identifier 64-bit (EUI-64) format. The EUI-64 format interface ID is derived from the 48-bit link-layer MAC address by inserting the 16-bit hexadecimal number FFFE between the upper three bytes (the organizationally unique identifier [OUI] field) and the lower 3 bytes (the vendor code or serial number field) of the link-layer address. The seventh bit in the high-order byte is set to 1 to indicate the uniqueness of the interface ID.

This process is illustrated in Figure 8-5.

Figure 8-5. EUI-64 Format IPv6 Interface Identifier.



The seventh bit in an IPv6 interface identifier is referred to as the Universal/Local (U/L) bit. This bit identifies whether this interface identifier is locally unique on the link or whether it is universally unique. When the interface identifier is created from an Ethernet MAC address, it is assumed that the MAC address is universally unique and, therefore, that the interface identifier is universally unique. The purpose of the U/L bit is for future use by upper-layer protocols to uniquely identify a connection, even in the context of a change in the far left part of the address. However, this feature is not yet used.

The eighth bit in an IPv6 interface identifier, also known as the G bit, is the group/individual bit for managing groups.

Example 8-2 illustrates the configuration of a loopback interface, this time using the EUI-64 format for the interface ID by specifying the `eui-64` keyword. After configuration, the interface is examined, and the highlighted line illustrates the resulting IPv6 address on the interface. Notice that the first part of the address is what we specified, and the last part is the EUI-64 interface ID. The subnet is the 64 bits that we specified.

Example 8-2. Specifying *eui-64* Causes the Router to Create the Interface ID from Its Data Link Layer Address

```
Code View: Scroll / Show All
R1(config)#interface loopback 100
R1(config-if)#ipv6 address 2001:8:85a3:4289::/64 ?
 anycast Configure as an anycast
 eui-64 Use eui-64 interface identifier
R1(config-if)#ipv6 address 2001:8:85a3:4289::/64 eui-64
<output omitted>
R1#show ipv6 interface loopback 100
Loopback100 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::21B:D5FF:FE5B:A408
Global unicast address(es):
 2001:8:85A3:4289:21B:D5FF:FE5B:A408, subnet is 2001:8:85A3:4289::/64 [EUI]
Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF5B:A408
MTU is 1514 bytes
ICMP error messages limited to one every 100 milliseconds
ICMP redirects are enabled
ND DAD is not supported
ND reachable time is 30000 milliseconds
Hosts use stateless autoconfig for addresses.
R1#
```

Because of privacy and security concerns, hosts may create a random interface identifier using the MAC address as a base. This is considered a privacy extension because, without it, creating an interface identifier from a MAC address allows activity to be tracked to the point of connection. For example, Microsoft Windows XP implements this capability and prefers to use this address for outgoing communication because the address has a short lifetime and will be regenerated periodically. This process is defined in RFC 4941, *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*.

#### IPv6 Address Types

IPv6 has three main types of addresses. Unicast and multicast addresses are similar to their IPv4 companions. A new type of address, anycast, has also been introduced in IPv6. The following describe these IPv6 address types:

- **Unicast**— Similar to an IPv4 unicast address, an IPv6 unicast address is for a single interface. A packet that is sent to a unicast address goes to the interface identified by that address. As in IPv4, a subnet prefix in IPv6 is associated with one link. The IPv6 unicast address space encompasses the entire IPv6 address range, with the exception of the FF00::/8 range (addresses starting with binary 1111 1111), which is used for multicast addresses.

##### Note

An embedded IPv4 address is an IPv6 address that has the IPv4 address of the interface embedded within it. These addresses are described further in the “Tunneling IPv6 Traffic” section.

##### Note

There are multiple types of unicast addresses, including global aggregatable (which is also called global unicast), link-local, embedded IPv4, and site-local. (Note that site-local addresses are now deprecated.)

##### Note



RFC 4193, *Unique Local IPv6 Unicast Addresses*, defines another new type of unicast address with the FC00::/7 prefix that is globally unique and is intended for local communications. It is not covered further in this chapter.

As described earlier, one difference between IPv4 and IPv6 unicast addresses is that the interface ID (the host portion of the address) can be automatically derived from the data link layer address for IPv6.

- **Multicast**— An IPv6 multicast address identifies a set of interfaces on different devices. Just as in IPv4, a packet sent to a multicast address is delivered to *all* the interfaces identified by the multicast address. The range of multicast addresses in IPv6 is larger than in IPv4, and for the foreseeable future, allocation of IPv6 multicast groups is not being limited.
- **Anycast**— An IPv6 anycast address is a new type of address that is assigned to a *set* of interfaces on *different* devices; an anycast address identifies multiple interfaces. A packet that is sent to an anycast address goes to the *closest* interface (as determined by the routing protocol being used) identified by the anycast address. Therefore, all nodes with the same anycast address should provide the same (uniform) service. Examples of when anycast addresses could be used are load balancing and content delivery services.

Anycast addresses are syntactically indistinguishable from global unicast addresses because anycast addresses are allocated from the global unicast address space.

Anycast addresses must not be used as the source address of an IPv6 packet.

In IPv4, broadcasting results in several problems, including generating interrupts in every computer on the network and, in some cases triggering malfunctions, known as *broadcast storms*, which can completely halt an entire network. It is important to notice that broadcasting does not exist in IPv6; broadcasts are replaced by multicasts and anycasts. Multicast enables efficient network operation by using specific multicast groups to send requests to a limited number of computers on the network. The multicast groups prevent most of the problems related to broadcast storms in IPv4.

Note that a single interface may be assigned multiple IPv6 addresses of any type (unicast, anycast, and multicast). Every IPv6-enabled interface must contain at least one loopback (::1/128) and one link-local address. Optionally, an interface may have multiple unique local and global addresses.

The various address types are detailed in the following sections.

#### IPv6 Global Unicast Addresses

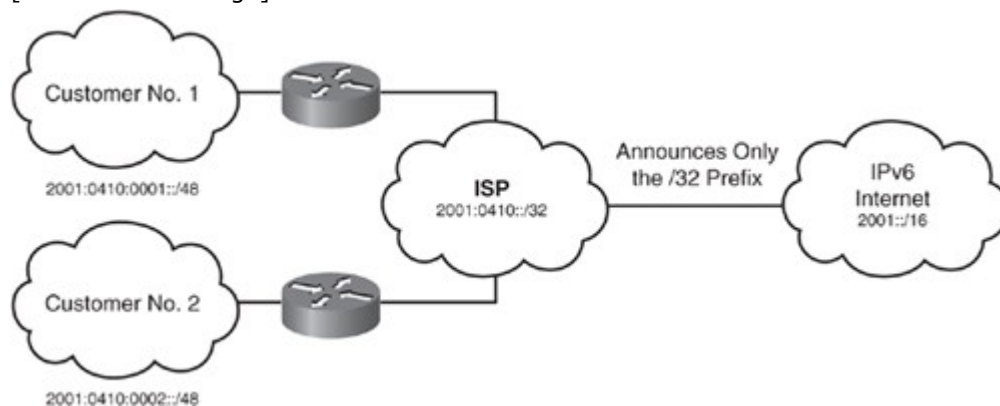
The IPv6 addressing architecture is defined in RFC 4291, *IP Version 6 Addressing Architecture*.

The IPv6 global aggregatable unicast address, also known as the IPv6 global unicast address, is the equivalent of the IPv4 global unicast address.

A global unicast address is an IPv6 address from the global unicast prefix. The structure of global unicast addresses enables aggregation of routing prefixes so that the number of routing table entries in the global routing table can be reduced. Global unicast addresses used on links are aggregated upward through organizations and eventually to the ISPs, as illustrated in Figure 8-6. This provides for more efficient and scalable routing within the Internet, and improved bandwidth and functionality for user traffic.

Figure 8-6. IPv6's Larger Address Space Enables Address Aggregation.

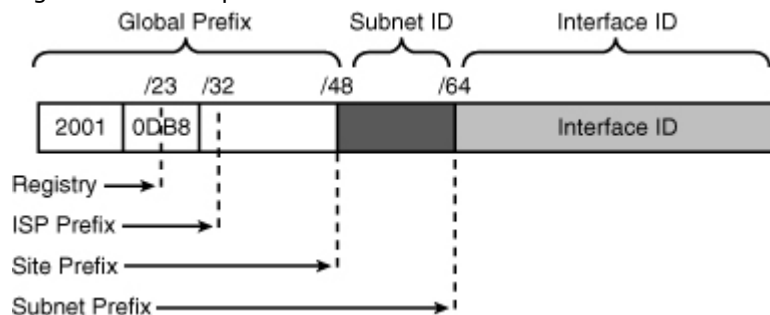
[View full size image]



The global unicast address typically consists of a 48-bit global routing prefix, a 16-bit subnet ID, and a 64-bit interface ID (typically in EUI-64 bit format), as illustrated in the example in Figure 8-7.



Figure 8-7. Example of an IPv6 Global Unicast Address.



The subnet ID can be used by individual organizations to identify subnets and create their own local addressing hierarchy. This 16-bit field allows an organization to use up to 65,536 individual subnets.

Addresses with a prefix of 2000::/3 (binary 001) through E000::/3 (binary 111), excluding the FF00::/8 (binary 1111 1111) multicast addresses, are required to have 64-bit interface identifiers in the EUI-64 format.

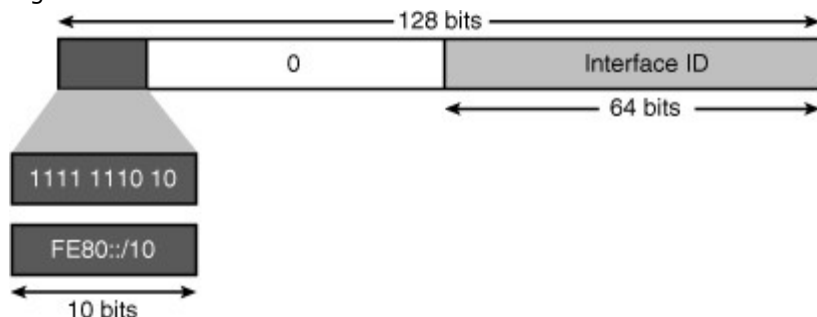
The current global unicast address assignment by the Internet Assigned Numbers Authority (IANA) uses the range of addresses that start with binary value 001. Another way to write this is 2000::/3, which means that they start with the same 3 bits as 2000; the 3 bits are 001. (Note that you have to write 16 bits, which is 4 hexadecimal digits, before the first colon.) This is one-eighth (12.5 percent) of the total IPv6 address space and is the largest block of assigned addresses.

The IANA is allocating the IPv6 address space in the ranges of 2001::/16 to the registries.

#### IPv6 Link-Local Unicast Addresses

Link-local addresses have a scope limited to the local link and are dynamically created on all IPv6 interfaces by using a specific link-local prefix FE80::/10 and a 64-bit interface identifier, as shown in Figure 8-8. Link-local addresses are used for automatic address configuration, neighbor discovery, router discovery, and by many routing protocols.

Figure 8-8. IPv6 Link-Local Address Structure.



A link-local unicast address can serve as a method to connect devices on the same local network without requiring global addresses.

When communicating with a link-local address, the outgoing interface must be specified because every interface is connected to FE80::/10. For example, if you want to ping from one router to its neighbor using the neighbor's link-local address, you will be asked to input the interface on which you want to ping, because the router cannot determine the outgoing interface by looking at a link-local destination address.

Example 8-3 repeats the show ipv6 interface output from the earlier Example 8-2, highlighting the automatically generated link-local address on the interface.

Example 8-3. A Link-Local Address is Automatically Created on all IPv6-Enabled Interfaces

Code View: Scroll / Show All

R1#**show ipv6 interface loopback 100**

Loopback100 is up, line protocol is up

IPv6 is enabled, link-local address is FE80::21B:D5FF:FE5B:A408

Global unicast address(es):

2001:8:85A3:4289:21B:D5FF:FE5B:A408, subnet is 2001:8:85A3:4289::/64 [EUI]

Joined group address(es):

FF02::1

FF02::2

FF02::1:FF5B:A408

MTU is 1514 bytes

ICMP error messages limited to one every 100 milliseconds

ICMP redirects are enabled

ND DAD is not supported

ND reachable time is 30000 milliseconds

Hosts use stateless autoconfig for addresses.

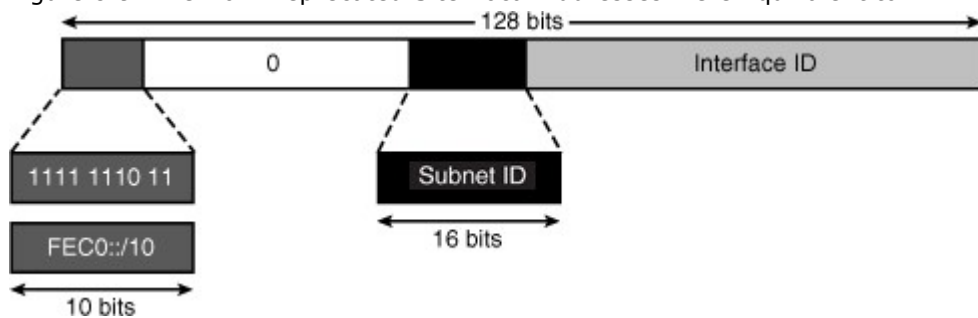
R1#

#### IPv6 Site-Local Unicast Addresses: Deprecated

Site-local addresses were originally created to be similar to IPv4 private addresses. Note that this address type has since been deprecated (it is no longer supported). Because of the huge IPv6 address space, private addresses are not needed and using them would mean that NAT would be required and that addresses would again not be end-to-end. Site-local addresses are mentioned here only for legacy reasons.

As illustrated in Figure 8-9, site-local addresses were in the FEC0::/10 range, meaning they started with the same 10 bits as FEC0 (1111 1110 11).

Figure 8-9. The Now-Deprecated Site-Local Addresses Were Equivalent to IPv4 Private Addresses.



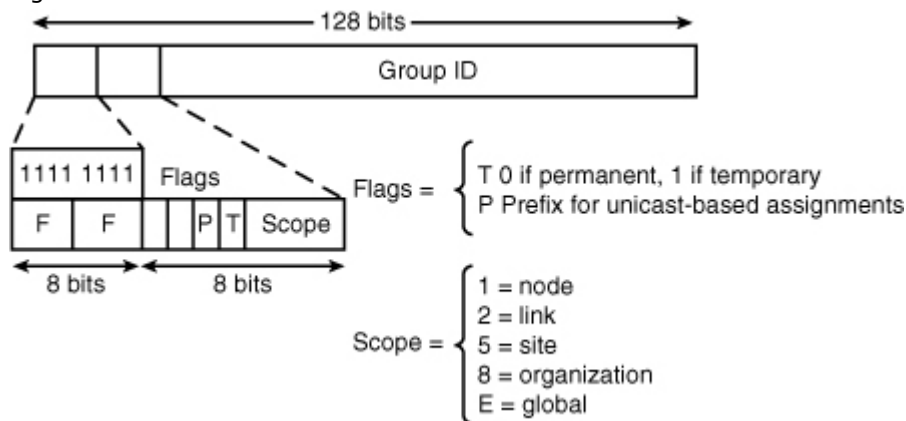
#### IPv6 Multicast Addresses

A multicast address identifies a group of interfaces; traffic sent to a multicast address travels to multiple destinations at the same time. An interface may belong to any number of multicast groups. Multicasting is extremely important to IPv6, because it is at the core of many IPv6 functions and it is a replacement for broadcast.

Multicast addresses are more efficient than broadcast addresses because traffic is sent only to the multicast recipients that have subscribed to the traffic. These recipients have joined the multicast group.

The format of an IPv6 multicast address is illustrated in Figure 8-10. IPv6 multicast addresses are defined by the prefix FF00::/8. The second octet of the address contains the prefix and transient (lifetime) flags, and the scope of the multicast address, as follows:

Figure 8-10. IPv6 Multicast Address Structure.



- The transient (T) flag parameter is equal to 0 for a permanent, or well-known, multicast address. The flag is equal to 1 for a temporary (also known as a transient or dynamically assigned) multicast address.
- The prefix (P) flag parameter indicates a prefix. This flag allows part of the multicast group address to include the unicast prefix of the source network.
- The 4-bit scope parameter values include—1 for the node scope (for loopback transmission), 2 for the link scope (similar to unicast link-local scope), 5 for the site-local scope, 8 for the organizational scope (for multiple sites), and E for the global scope.

For example, a multicast address starting with FF02::/16 is a permanent multicast address with a link-local scope. There is no TTL field in IPv6 multicast packets because the scoping is defined inside the address.

The multicast group ID consists of the lower 112 bits of the multicast address.

The multicast addresses FF00:: to FF0F:: have the T flag set to 0 and are reserved. Within that range, the following are some example assigned addresses. (Many more assignments are made, and assignments are tracked by IANA.)

- **FF02::1**—"All nodes" on a link (link-local scope).

Note

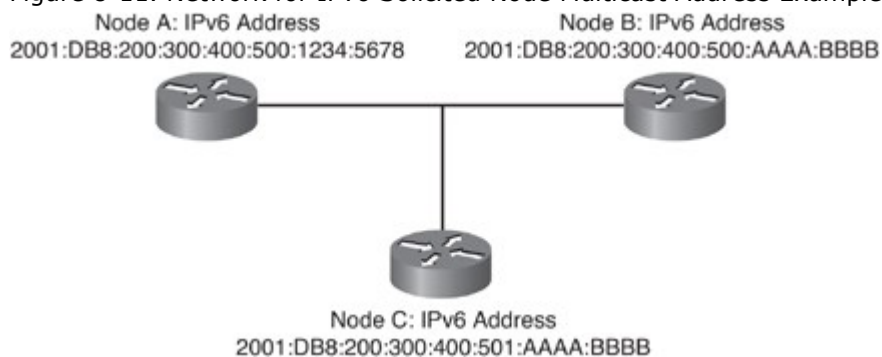
The FF02::1 multicast address is similar to a broadcast on a link.

- FF02::2—"All routers" on a link.
- **FF02::9**—"All routing information protocol (RIP) routers" on a link.
- **FF02::1:FFXX:XXXX**—Solicited-node multicast on a link, where the XX:XXXX is the far right 24 bits of the corresponding unicast or anycast address of the node. Solicited-node multicast addresses are used in neighbor solicitation (NS) messages, which are sent on a local link when a node wants to determine the link-layer address of another node on the same local link, similar to the Address Resolution Protocol (ARP) in IPv4. The neighbor discovery (ND) process is described in the following section.
- **FF05::101**—"All Network Time Protocol (NTP) servers" in the site (site-local scope). (The site-local multicast scope has an administratively assigned radius and has no direct correlation to the now deprecated site-local unicast prefix of FEC0::/10.)

#### Solicited-Node Multicast Addresses

Solicited-node multicast addresses are used in IPv6 during address resolution of an IPv6 address to a MAC address on a LAN segment. In rare cases, the far right 24 bits of the unicast address of the target will not be unique on a link, but this will not cause a problem, as illustrated by an example using the devices in Figure 8-11:

Figure 8-11. Network for IPv6 Solicited Node Multicast Address Example.



- Node A has IPv6 address 2001:DB8:200:300:400:500:1234:5678.
- Node B has IPv6 address 2001:DB8:200:300:400:500:AAAA:BBBB, and would therefore have solicited-node multicast address FF02::1:FFAA:BBBB, which can also be written as FF02::1:FFAA:BBBB.
- Node C has IPv6 address 2001:DB8:200:300:400:501:AAAA:BBBB, and would therefore have solicited-node multicast address FF02::1:FFAA:BBBB. Note that this is the same as node B's solicited-node multicast address.

The neighbor discovery process uses these solicited-node multicast addresses. When node A desires to exchange packets with node B, node A sends an NS message to the solicited-node multicast address of node B, FF02::1:FFAA:BBBB. This message contains, in addition to other data, the full IPv6 address that node A is looking for, 2001:DB8:200:300:400:500:AAAA:BBBB. This is called the *target address*.

#### Note

The neighbor discovery process is defined in RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)*.

Both node B and node C are listening to the same solicited-node multicast address, so they both receive and process the packet. Node B sees that the target address inside the packet is its own and responds with a neighbor advertisement that includes its MAC address. Meanwhile, node C sees that the target address inside the packet is not its own and does not respond.

In this manner, nodes can have the same solicited-node multicast address, but not cause the neighbor discovery process to malfunction.

#### Note

Neighbor discovery is described further in the "Stateless Autoconfiguration of IPv6 Addresses" section, later in this chapter.

### IPv6 Anycast Addresses

An IPv6 anycast address is a global unicast address that is assigned to more than one interface. The format is illustrated in Figure 8-12. For IPv6, anycast is defined as a way to send a packet to the nearest (or closest) interface that is a member of the anycast group, thus providing a discovery mechanism to the nearest point.

Figure 8-12. IPv6 Anycast Address Structure.



A sender creates a packet with an anycast address as the destination address and forwards the packet to its nearest router. The router routes the packet to the nearest anycast interface (the closest device or interface that shares that address). For directly connected neighbors, the nearest interface is found according to the first neighbor that is learned about. Otherwise, the nearest interface is found according to the measure of the metric of the routing protocol.

For example, interfaces on three servers may be assigned the same anycast address. A router receives a packet for this address; it determines which is closest, based on the metric of the routing protocol, and sends traffic to that closest recipient.

Anycast addresses are allocated from the unicast address space and have the same format as unicast addresses, so they are indistinguishable from unicast addresses. To devices that are not configured for

anycast, these addresses appear as unicast addresses. When a unicast address is assigned to more than one interface—thus turning it into an anycast address—the nodes to which the address is assigned must be explicitly configured to use and know that the address is an anycast address.

#### Note

On a Cisco router, an IPv6 address becomes an anycast address when the anycast keyword is added to the end of the ipv6 address command. This parameter can be observed in the output in the earlier Example 8-2.

The idea of anycast in IP was proposed in 1993. However, only a few IPv6 anycast addresses are currently assigned, including the router-subnet anycast and the Mobile IPv6 home agent anycast.

An anticipated use for anycast addresses is to control the paths across which traffic flows. An example use in a Border Gateway Protocol (BGP) multihomed network would be when a customer has multiple ISPs and multiple connections to each one. Each ISP could have a different anycast address. The same anycast address would be used for all routers of a given ISP. A customer device chooses which ISP to send the packet to; the routers along the path determine the closest router by which that ISP can be reached using the IPv6 anycast address.

Another use for an anycast address is when multiple routers are attached to a LAN. These routers can have the same IPv6 anycast address so that distant devices only need to identify the anycast address. Intermediate devices choose the best path to reach the closest entry point to that LAN.

#### Comparing IPv6 Addresses with IPv4 Addresses

Although there are many differences between IPv4 and IPv6, there are also many similarities, and you can leverage your IPv4 knowledge when learning and using IPv6.

In earlier sections, you saw the format of IPv6 addresses. In devices, though, these addresses, like IPv4 addresses, are kept in binary. And like IPv4 addresses, IPv6 addresses can be summarized by finding the bits that addresses have in common.

For example, consider the following IPv4 addresses: 172.16.12.0/24, 172.16.13.0/24, 172.16.14.0/24, and 172.16.15.0/24. These addresses obviously have 172.16 in common, but they also have the first 6 bits of the third octet in common (000011). Therefore, the summary of these four routes is 172.16.12.0/22.

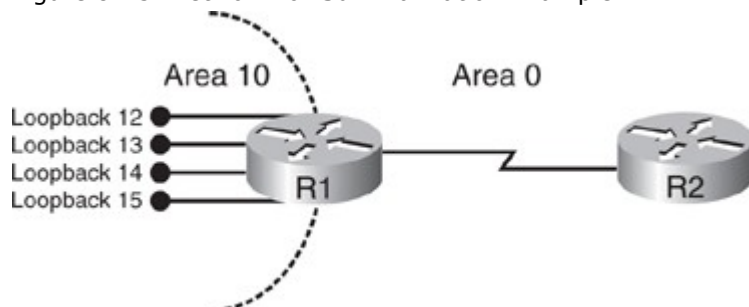
Similarly, IPv6 routes can be summarized. For example, let's use the same addresses, but in IPv6 hexadecimal format. If we write out 172.16.12.0 in binary, we get the following:

```
10101100.00010000.00001100.00000000
```

Converting this to hexadecimal, we get AC10:0C00. Therefore, we could use AC10:0C00::/24 as an IPv6 address. The other three addresses are AC10:0D00::/24, AC10:0E00::/24, and AC10:0F00::/24, respectively. The summary of these four addresses is AC10:0C00::/22, the same as we get for IPv4.

To see this on Cisco routers, consider the simple network shown in Figure 8-13.

Figure 8-13. Network for Summarization Example.



Example 8-4 shows the loopback interfaces' IPv4 and IPv6 addresses configured on R1. Notice that the IPv6 addresses have the interface ID field set to 1 on each of the networks described in the summarization discussion. (Each loopback interface also has an automatically created link-local address.)

Example 8-4. Loopback Interface Addresses Configured on R1

Code View: Scroll / Show All

R1# **show ip interface brief | begin Loop**

|            |             |               |    |
|------------|-------------|---------------|----|
| Loopback12 | 172.16.12.1 | YES manual up | up |
| Loopback13 | 172.16.13.1 | YES manual up | up |
| Loopback14 | 172.16.14.1 | YES manual up | up |

```

Loopback15 172.16.15.1 YES manual up up
Loopback100 unassigned YES unset up up
R1#

```

R1# **show ipv6 interface brief | begin Loop**

```

Loopback12 [up/up]
 FE80::21B:D5FF:FE5B:A408
 AC10:C00::1
Loopback13 [up/up]
 FE80::21B:D5FF:FE5B:A408
 AC10:D00::1
Loopback14 [up/up]
 FE80::21B:D5FF:FE5B:A408
 AC10:E00::1
Loopback15 [up/up]
 FE80::21B:D5FF:FE5B:A408
 AC10:F00::1
R1#

```

The routers are running Open Shortest Path First Version 2 (OSPFv2) for IPv4 and OSPF Version 3 (OSPFv3) for IPv6. OSPFv3 and its associated configuration and verification commands are described in detail in the later "OSPFv3" section. For this discussion we need just the understanding of how OSPF works in general, as described for IPv4 in Chapter 3, "Configuring the Open Shortest Path First Protocol."

Example 8-5 illustrates the routing table on R2, for both IPv4 and IPv6. R2 sees all of R1's IPv4 and IPv6 loopback interface addresses as interarea (O IA) routes.

Example 8-5. R2's Routing Tables

Code View: Scroll / Show All

R2# **show ip route ospf**

172.16.0.0/32 is subnetted, 4 subnets

```

O IA 172.16.13.1 [110/65] via 10.10.10.1, 00:01:49, Serial0/0/0
O IA 172.16.12.1 [110/65] via 10.10.10.1, 00:01:49, Serial0/0/0
O IA 172.16.15.1 [110/65] via 10.10.10.1, 00:01:49, Serial0/0/0
O IA 172.16.14.1 [110/65] via 10.10.10.1, 00:01:49, Serial0/0/0

```

R2#

R2# **show ipv6 route ospf**

IPv6 Routing Table - 6 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP

U - Per-user Static route

I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary

O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2

ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2

```

OI AC10:C00::1/128 [110/64]
 via FE80::1, Serial0/0/0
OI AC10:D00::1/128 [110/64]
 via FE80::1, Serial0/0/0
OI AC10:E00::1/128 [110/64]
 via FE80::1, Serial0/0/0
OI AC10:F00::1/128 [110/64]

```

via FE80::1, Serial0/0/0

To examine what happens when summarization is configured, the debug ip routing and debug ipv6 routing commands are entered on R2. The area 10 range 172.16.12.0 255.255.252.0 command is then configured on R1, to summarize the four IPv4 routes into 172.16.12.0/22. Example 8-6 illustrates the resulting debug messages on R2; the detailed routes are deleted and the summary route is added.

#### Caution

Use caution when executing **debug** commands, because they may consume a lot of router resources and could cause problems in a busy production network. Debugging output takes priority over other network traffic. Too much debug output may severely reduce the performance of the router or even render it unusable in the worst case.

#### Example 8-6. Debug Output on R2

Code View: Scroll / Show All

R2:

```
*Dec 18 01:21:12.995: RT: del 172.16.15.1/32 via 10.10.10.1, ospf metric
[110/65]
*Dec 18 01:21:12.999: RT: delete subnet route to 172.16.15.1/32
*Dec 18 01:21:12.999: RT: NET-RED 172.16.15.1/32
*Dec 18 01:21:12.999: RT: del 172.16.14.1/32 via 10.10.10.1, ospf metric
[110/65]
*Dec 18 01:21:12.999: RT: delete subnet route to 172.16.14.1/32
*Dec 18 01:21:12.999: RT: NET-RED 172.16.14.1/32
*Dec 18 01:21:12.999: RT: del 172.16.13.1/32 via 10.10.10.1, ospf metric
[110/65]
*Dec 18 01:21:12.999: RT: delete subnet route to 172.16.13.1/32
*Dec 18 01:21:12.999: RT: NET-RED 172.16.13.1/32
*Dec 18 01:21:12.999: RT: del 172.16.12.1/32 via 10.10.10.1, ospf metric
[110/65]
*Dec 18 01:21:12.999: RT: delete subnet route to 172.16.12.1/32
*Dec 18 01:21:12.999: RT: NET-RED 172.16.12.1/32
*Dec 18 01:21:12.999: RT: delete network route to 172.16.0.0
*Dec 18 01:21:12.999: RT: NET-RED 172.16.0.0/16
*Dec 18 01:21:17.903: RT: SET_LAST_RDB for 172.16.12.0/22
NEW rdb: via 10.10.10.1
*Dec 18 01:21:17.907: RT: add 172.16.12.0/22 via 10.10.10.1, ospf metric
[110/65]
*Dec 18 01:21:17.907: RT: NET-RED 172.16.12.0/22
```

Example 8-7 illustrates the resulting routing table on R2; only the summary route is present.

#### Example 8-7. R2's IPv4 Routing Table After Summarization

R2#**show ip route ospf**

172.16.0.0/22 is subnetted, 1 subnet

O IA 172.16.12.0 [110/65] via 10.10.10.1, 00:00:32, Serial0/0/0

R2#



Similarly, for IPv6, the area 10 range AC10:C00::/22 command is configured on R1, to summarize the four routes into AC10:0C00::/22. Example 8-8 illustrates the resulting debug messages on R2; the detailed routes are deleted and the summary route is added.

Example 8-8. Debug Output on R2

```
Code View: Scroll / Show All
R2:
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Route add AC10:C00::/22 [new]
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Add AC10:C00::/22 to table
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Adding next-hop FE80::1 over Serial0/0/0
for AC10:C00::/22, [110/64]
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Delete AC10:F00::1/128 from table
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Delete AC10:E00::1/128 from table
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Delete AC10:D00::1/128 from table
*Dec 18 01:33:47.991: IPv6RT0: ospf 10, Delete AC10:C00::1/128 from table
```

Example 8-9 illustrates the resulting routing table on R2; only the summary route is present.

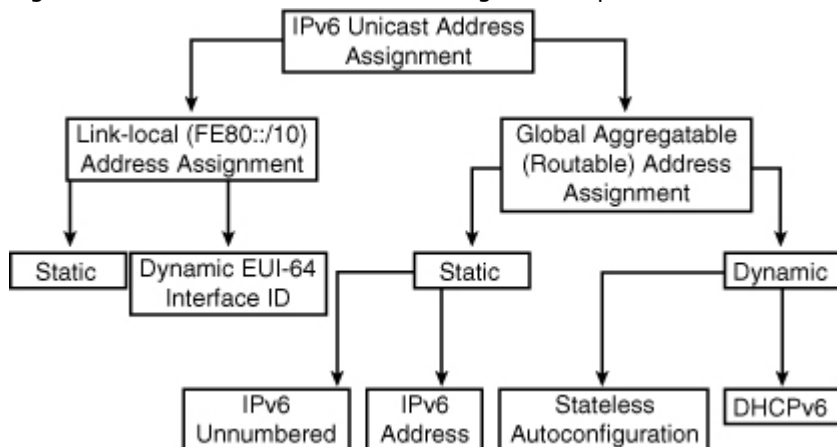
Example 8-9. R2's IPv6 Routing Table After Summarization

```
R2#show ipv6 route ospf
IPv6 Routing Table - 3 entries
<output omitted>
OI AC10:C00::/22 [110/64]
 via FE80::1, Serial0/0/0
R2#
```

### Configuring and Verifying IPv6 Unicast Addresses

This section examines how to configure and verify IPv6 unicast addresses on interfaces. As illustrated in Figure 8-14, various options are available when configuring an IPv6 unicast address. Notice that both the link-local unicast address (starting with FE80) and the global aggregatable unicast addresses can be assigned statically or dynamically.

Figure 8-14. IPv6 Unicast Address Assignment Options.



In the following section, the commands used to configure and verify IPv6 addresses are described. The sections after that discuss in detail the various options.

### IPv6 Unicast Address Configuration and Verification Commands

This section introduces many of the commands used in the following sections.

#### Note

Many IPv6 commands are introduced in this chapter, and in many cases there are more parameters for the commands than are identified here. For detailed command syntax, see the Cisco IOS Command References.

Use the **ipv6 unicast-routing** global configuration command to enable the forwarding of IPv6 unicast datagrams.

#### Note

We confirmed that the **ipv6 unicast-routing** command is only required on a router before configuring an IPv6 routing protocol. It is not needed before configuring IPv6 interface addresses. Therefore, the “IPv6 traffic forwarding” that this command enables is related to the ability of the interface to route IPv6 traffic, not to the ability of an interface to send IPv6 traffic. It is also required for the interface to provide stateless autoconfiguration. However, configuring **no ipv6 unicast-routing** turns off the IPv6 routing capabilities of the router; the router acts as an IPv6 end-station.

Optionally use the **ipv6 cef** global configuration command to enable Cisco Express Forwarding (CEF) for IPv6 (CEFv6). CEFv6 is an advanced, Layer 3 IP switching technology for the forwarding of IPv6 packets (and is required to be enabled for some other IPv6 features to operate). When CEFv6 is enabled, network entries that are added, removed, or modified in the IPv6 Routing Information Base (RIB), as dictated by the routing protocol in use, are reflected in the Forwarding Information Bases (FIBs), and the IPv6 adjacency tables maintain Layer 2 next-hop addresses for all entries that are in each FIB.

Use the **ipv6 address address/prefix-length [eui-64 | link-local]** interface configuration command to configure an IPv6 address and prefix (specified in the *address/prefix-length* parameters) for an interface and enable IPv6 processing on the interface. The **eui-64** parameter forces the router to complete the addresses’ low-order 64-bits using an EUI-64 format interface ID. The **link-local** parameter configures the address as the link-local address on the interface.

Use the **ipv6 address autoconfig [default]** interface configuration command to enable automatic configuration of IPv6 addresses using stateless autoconfiguration on an interface, and enable IPv6 processing on the interface. If a default router is selected on this interface, the optional **default** keyword causes a default route to be installed using that default router. The **default** keyword can be specified only on one interface.

Use the **ipv6 unnumbered interface-type interface-number** interface configuration command to enable IPv6 processing on an interface without assigning an explicit IPv6 address to the interface. The interface will use the IPv6 address of the interface specified by the *interface-type interface-number* parameters as the source address of traffic from the configured interface. The interface specified in the command must be in the “up” state.

Use the **ipv6 nd reachable-time milliseconds** interface configuration command to specify the number of milliseconds (from 0 to 3,600,000) that a remote IPv6 node is considered reachable after some reachability confirmation event has occurred, such as the neighbor discovery process. The default is 0 milliseconds (indicating an unspecified time) in router advertisements and 30,000 (30 seconds) for the neighbor discovery activity of the router itself. The configured time is used by the router to detect unavailable neighbors. Shorter configured times enable the router to detect unavailable neighbors more quickly; however, shorter times consume more IPv6 network bandwidth and processing resources in all IPv6 network devices. Very short configured times are not recommended in normal IPv6 operation.

Use the **ipv6 neighbor ipv6-address interface-type interface-number hardware-address** global configuration command to statically configure an entry in the IPv6 neighbor discovery cache, mapping the IPv6 address to the hardware address on an interface.

The **show ipv6 interface [brief] [type number] [prefix]** EXEC command displays the usability and status of interfaces configured for IPv6. The **brief** keyword displays a brief summary of the IPv6 status and configuration for each interface. The *type number* parameters indicate the type and number of the interface about which to display information. The **prefix** keyword displays the IPv6 neighbor discovery prefixes that are configured on a specified interface.

Use the **show ipv6 routers [interface-type interface-number] [conflicts]** EXEC command to display IPv6 router advertisement information received from onlink routers (those locally reachable on the link).

The *interface-type interface-number* parameters indicate the type and number of the interface about which to display information. The **conflicts** keyword displays information about routers advertising parameters that differ from the advertisement parameters configured for the specified interface on which the advertisements are received.

The **show ipv6 neighbors** [*interface-type interface-number* | *ipv6-address* | *ipv6-hostname* | **statistics**] EXEC command displays IPv6 neighbor discovery cache information for the specified neighbors. The optional **statistics** parameter displays neighbor discovery cache statistics.

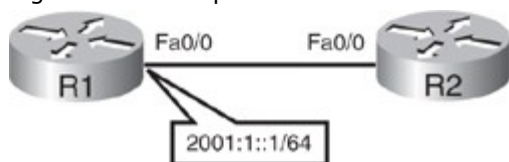
The **debug ipv6 nd** EXEC command displays messages associated with ICMPv6 neighbor discovery. As described earlier in the “Solicited-Node Multicast Addresses” section, ICMPv6 neighbor discovery is the IPv6 replacement for the IPv4 ARP.

The **debug ipv6 packet** [**access-list** *access-list-name*] [**detail**] EXEC command displays information associated with IPv6 packet processing. When an IPv6 access list is specified by using the *access-list-name* parameter, only packets permitted by the access list are displayed. Detailed information is displayed when the **detail** keyword is specified.

#### Static IPv6 Address Assignment

Figure 8-15 illustrates a simple network with two routers. This network is used to illustrate the configuration of various IPv6 addressing features. First, static address assignment on Router R1 is explored.

Figure 8-15. Simple Network for Address Assignment Examples.



#### Static Global Aggregatable Address Assignment

Initially, the **no ipv6 unicast-routing** command is used on R1 to disable the routing capabilities the router acts as an end station for IPv6. The **debug ipv6 nd** command is used to view IPv6 neighbor discovery events in real time, as shown at the top of Example 8-10.

Example 8-10. Configuration on R1

Code View: Scroll / Show All

R1#**debug ipv6 nd**

ICMP Neighbor Discovery events debugging in on

R1#

\*Aug 13 08:11:00.309: ICMPv6-ND: Received RA from FE80::219:55FF:FEDF:AD22 on FastEthernet0/0

R1#

R1(config)#**interface fa0/0**

R1(config-if)#**ipv6 address 2001:1::1/64**

R1(config-if)#

\*Aug 13 08:12:19.541: ICMPv6-ND: Adding prefix 2001:1::1/64 to FastEthernet0/0

\*Aug 13 08:12:19.541: ICMPv6-ND: Sending NS for 2001:1::1 on FastEthernet0/0

\*Aug 13 08:12:20.541: ICMPv6-ND: DAD: 2001:1::1 is unique.

\*Aug 13 08:12:20.541: ICMPv6-ND: Sending NA for 2001:1::1 on FastEthernet0/0

\*Aug 13 08:12:20.541: ICMPv6-ND: Address 2001:1::1/64 is up on FastEthernet0/0

\*Aug 13 08:13:37.941: ICMPv6-ND: Received RA from FE80::219:55FF:FEDF:AD22 on FastEthernet0/0

As soon as the debugging is enabled, router advertisements (RAs) start coming from R2 on Fast Ethernet 0/0, using its link-local address. As shown next in Example 8-10, an IPv6 address, 2001:1::1/64, is configured on R1's Fast Ethernet 0/0. In this example, the entire address is manually configured, so the EUI-64 format for the interface ID was not used. Immediately the debug output shows that the prefix 2001:1::1/64 is added to the interface, and R1 sends an NS for that address. The NS solicits the link to ensure that its address is unique. The duplicate address detection (DAD) message from R2 confirms that this

address is unique on the link. R1 then sends a neighbor advertisement (NA), which includes its address; NAs are sent when there is an address change on an interface, including when an address is first assigned to an interface. The debug output confirms that the address is up on the interface. The last line of the debug output indicates that R1 received another RA from R2.

Note

RAs, NAs, and DADs are described further in the “Stateless Autoconfiguration of IPv6 Addresses” section, later in this chapter.

It is important to note that the RAs are not routing updates (which are sent by routing protocols). Rather, RAs are sent by the ND process to advertise autoconfiguration parameters, including the IPv6 prefix on the link.

As shown next in Example 8-11, IPv6 routing is enabled on R1 and it sends RAs. The RAs include the MTU of the interface (1500 bytes), and the prefix configured on the interface (2001:1::/64). The lower part of this Example 8-11 illustrates output from the show ipv6 interface command, to verify the interface parameters. Interface Fast Ethernet 0/0 has a link-local address and a global unicast address; it has also joined several multicast groups, including FF02::1 (all hosts) and FF02::2 (all routers). The ND information, including how often RAs are sent and how long they live, is also included in the output of this show command.

Example 8-11. Configuration and Status of R1

Code View: Scroll / Show All

R1(config)#**ipv6 unicast-routing**

R1(config)#

\*Aug 13 08:25:51.093: ICMPv6-ND: Sending RA to FF02::1 on FastEthernet0/0

\*Aug 13 08:25:51.093: ICMPv6-ND: MTU = 1500

\*Aug 13 08:25:51.093: ICMPv6-ND: prefix = 2001:1::/64 onlink autoconfig

\*Aug 13 08:25:51.093: ICMPv6-ND: 2592000/604800 (valid/preferred)

R1(config)#

R1#**show ipv6 interface fa0/0**

FastEthernet0/0 is up, line protocol is up

IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60

Global unicast address(es):

2001:1::1, subnet is 2001:1::/64

Joined group address(es):

FF02::1

FF02::2

FF02::1:FF00:1

FF02::1:FF2C:9F60

MTU is 1500 bytes

ICMP error messages limited to one every 100 milliseconds

ICMP redirects are enabled

ND DAD is enabled, number of DAD attempts: 1

ND reachable time is 30000 milliseconds

ND advertised reachable time is 0 milliseconds

ND advertised retransmit interval is 0 milliseconds

ND router advertisements are sent every 200 seconds

ND router advertisements live for 1800 seconds

Hosts use stateless autoconfig for addresses.

R1#

### Assigning Multiple Global Aggregatable Addresses

As mentioned earlier, IPv6 interfaces can have multiple addresses of various types assigned to them. These addresses can be used simultaneously. There is no concept of secondary addresses, as is the case for IPv4.

To illustrate the difference between IPv4 and IPv6, the Fast Ethernet 0/0 interface on R1 in Figure 8-15 is examined the output is shown at the top of Example 8-12.

#### Example 8-12. R1's Fast Ethernet 0/0 Configuration

Code View: Scroll / Show All

R1#**show run interface fa0/0**

Building configuration...

Current configuration : 135 bytes

!

```
interface FastEthernet0/0
 ip address 10.10.10.1 255.255.255.0
 duplex auto
 speed auto
 ipv6 address 2001:1::1/64
 ipv6 enable
end
```

R1#

R1(config)#**interface fa0/0**

R1(config-if)#**ip address 10.20.20.1 255.255.255.0**

R1(config-if)#**do show run interface fa0/0**

Building configuration...

Current configuration : 135 bytes

!

```
interface FastEthernet0/0
 ip address 10.20.20.1 255.255.255.0
 duplex auto
 speed auto
 ipv6 address 2001:1::1/64
 ipv6 enable
end
```

R1(config-if)#**ip address 10.10.10.1 255.255.255.0**

R1(config-if)#**ipv6 address 2002:1::1/64**

R1(config-if)#**do show run interface fa0/0**

Building configuration...

Current configuration : 162 bytes

!

```
interface FastEthernet0/0
 ip address 10.10.10.1 255.255.255.0
 duplex auto
 speed auto
 ipv6 address 2001:1::1/64
```

```
ipv6 address 2002:1::1/64
ipv6 enable
end
R1(config-if)#
```

Notice that the interface has an IPv4 and an IPv6 address. The middle of Example 8-12 illustrates what happens when another IPv4 IP address is assigned and the configuration is viewed again. Notice that the interface has only one IPv4 address, 10.20.20.1; the original address is gone. Next, the original IPv4 address is restored, and then another IPv6 global aggregatable unicast address is configured. Notice at the end of Example 8-12, the new IPv6 address is added to the interface and did not overwrite the original IPv6 address. IPv6 does not have the concept of secondary addresses; in IPv6 an interface can have multiple addresses.

#### Note

RFC 3484, *Default Address Selection for Internet Protocol version 6 (IPv6)*, explains how the source address of IPv6 traffic coming from an interface with multiple addresses is determined.

#### IPv6 Unnumbered Interfaces

IPv6 supports unnumbered interfaces, similar to the IPv4 unnumbered interface feature, to enable IPv6 processing on an interface without assigning an explicit IPv6 address to the interface. In Example 8-13, a loopback interface is created and configured with an IPv6 address. The Serial 0/0/0 interface is then configured to use the IPv6 address of the loopback interface, with the `ipv6 unnumbered loopback` command. The output of the `show ipv6 interface s0/0/0` command illustrates that the Serial 0/0/0 interface uses the IPv6 address from interface loopback 10.

#### Example 8-13. IPv6 Unnumbered Configuration and Verification

Code View: Scroll / Show All

```
R1(config-if)#exit
R1(config)#interface loopback 10
R1(config-if)#ipv6 address 2003:1::10/64
R1(config-if)#interface s0/0/0
R1(config-if)#ipv6 unnumbered loopback 10
R1(config-if)#

R1(config-if)#do show ipv6 interface s0/0/0
Serial0/0/0 is administratively down, line protocol is down
 IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60 [TEN]
 Interface is unnumbered, using address of Loopback10
 No global unicast address is configured
 Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF2C:9F60
 MTU is 1500 bytes
 ICMP error messages limited to one every 100 milliseconds
 ICMP redirects are enabled
 ND DAD is not supported
 ND reachable time is 30000 milliseconds
 Hosts use stateless autoconfig for addresses.
R1(config-if)#
```

### Static Link-Local Address Assignment

Recall that a device's link-local address is assigned dynamically by default, using a prefix FE80::/10 and the EUI-64 format interface ID. The showcommand output in Example 8-13 illustrates the dynamically assigned link-local address on the R1 router. Such link-local addresses are easy to configure, because you do not have to do anything, but they are difficult to work with, because you need to know the data link layer address of the interface to determine where it belongs.

Link-local addresses can also be statically assigned, using either the EUI-64 format or a manually configured interface ID. Using a manually configured interface ID makes it much easier to remember, and therefore use, the link-local address. Example 8-14 illustrates the configuration of the link-local address on an interface, using the ipv6 address FE80::1 link-local command. The interface ID is set to 1. The lower part of Example 8-14 illustrates the output of the show ipv6 interface fa0/0 command. The link-local address has been overwritten; unlike the global unicast address, an interface can only have one link-local address.

#### Example 8-14. Configuration and Verification of Link-Local Address

```
R1(config-if)#interface fa0/0
R1(config-if)#ipv6 address FE80::1 ?
link-local use link-local address

R1(config-if)#ipv6 address FE80::1 link-local
R1(config-if)#do show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::1
Global unicast address(es):
 2001:1::1, subnet is 2001:1::/64
 2002:1::1, subnet is 2002:1::/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
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
ND advertised reachable time is 0 milliseconds
ND advertised retransmit interval is 0 milliseconds
```

### Stateless Autoconfiguration of IPv6 Addresses

This section details the various phases of stateless autoconfiguration.

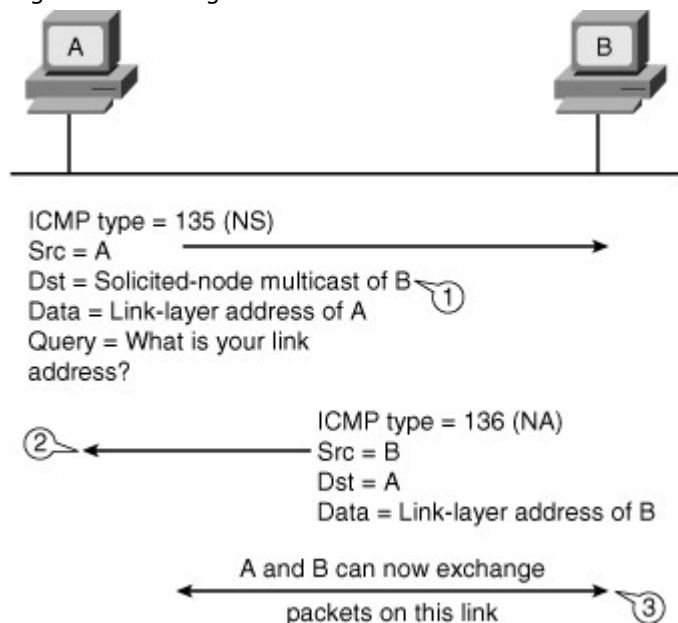
An IPv6 router on a local link can send (either periodically or upon a host's request) network information, such as the 64-bit prefix of the local link network and the default route, to all the nodes on the local link. Hosts can autoconfigure themselves by appending their IPv6 interface identifier (in EUI-64 format) to the local link 64-bit prefix.

As described earlier in the "Solicited-Node Multicast Addresses" section, the neighbor discovery or solicitation phase is similar to IPv4's ARP. It allows an IPv6 router to determine the link-layer address of neighbors on the same link, to find neighbor routers, and to keep track of those neighbors for stateless autoconfiguration. In the earlier "Static IPv6 Address Assignment" section, we saw debug output from the neighbor discovery (or solicitation) process, which works on any IPv6 device, including hosts and routers. Figure 8-16 illustrates how this process works. An ICMPv6 message type 135, NS, is sent on the link. The source address is the IPv6 address of the source node. The destination address is the solicited-node multicast address corresponding to the IPv6 address of the destination node. The message also includes the link-layer address of the source node, so the destination node will be able to use this address immediately. Each destination



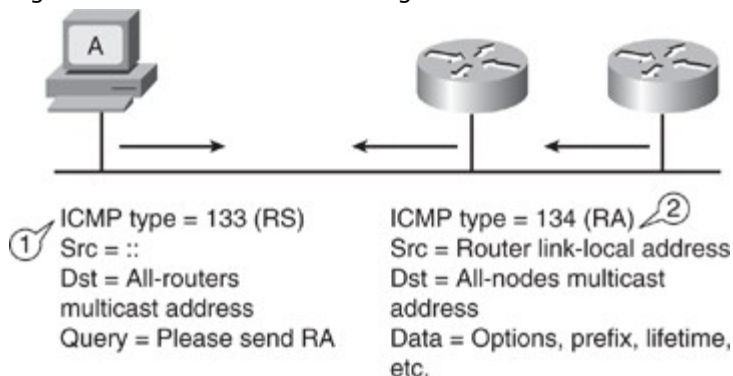
node that receives the NS responds with an ICMPv6 message type 136, NA. The source address of this message is the IPv6 address of the responding node, and the destination address is the IPv6 address of the original source node (which sent the NS). The data portion includes the link-layer address of the destination node (even though the link-layer address is of course also included in the frame). The two devices can now communicate on the link because they know each other's link-layer addresses.

Figure 8-16. Neighbor Solicitation.



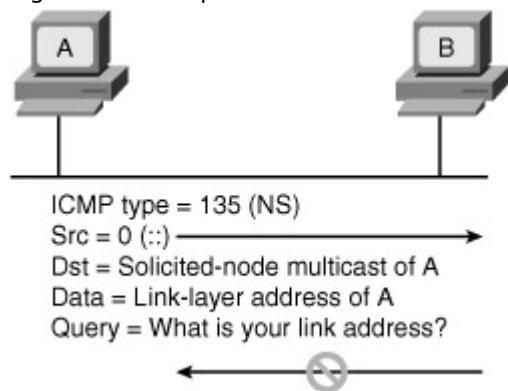
After the devices are communicating, a host may look for autoconfiguration information. Figure 8-17 illustrates how this process works. Stateless autoconfiguration uses the information in RA messages to configure hosts automatically and without human intervention. RAs are sent periodically, but a node can send out RS messages when it boots so that it doesn't have to wait for the next RA. All routers on the network reply to the RS immediately, with an RA sent to the all-nodes multicast address. The prefix included in the RA is used as the /64 prefix for the host address. The interface ID used is the EUI-64 format interface ID.

Figure 8-17. Stateless Autoconfiguration Uses Router Solicitations and Router Advertisements.



After obtaining an address, the DAD phase occurs, in which the node verifies that its new IPv6 address is unique on that link. Again, we saw debug output from this earlier. Figure 8-18 illustrates how this process works. DAD uses an NS to query if another node on the link has the same IPv6 address. It sends an NS packet to the solicited-node multicast address of its own IPv6 address. The source address of this packet is the unspecified address ::. If a node responds to the request, it means that the IPv6 address is already in use, and the requesting node should not use that address. DAD is used during the autoconfiguration process to ensure that no other device is using the autoconfiguration address.

Figure 8-18. Duplicate Address Detection Process.



Stateless autoconfiguration can also be used for renumbering devices easily; to renumber an entire site you only need renumber the routers. In this case, as shown in Figure 8-19, RA messages contain both the old and the new prefix. The lifetime of the old prefix is decreased, to indicate that the hosts should use the new prefix, but still keep their current connections open with the old prefix. During the overlap time, nodes have two unicast addresses. When the old prefix's lifetime expires, the RAs include only the new prefix.

Figure 8-19. Renumbering Is Simplified with Stateless Autoconfiguration.

- ICMP type = 134 (RA)
- Src = Router link-local address
- Dst = All-nodes multicast address
- Data = Two prefixes:
  - Current prefix (to be deprecated) with short lifetime
  - New prefix (to be used) with normal lifetime

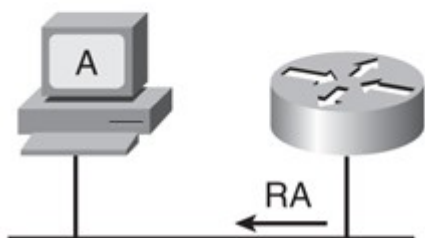


Figure 8-20 illustrates a simple network used to illustrate the stateless autoconfiguration process. Initially, the R2 router has the no ipv6 unicast-routing command configured. R1 will be configured as a stateless autoconfiguration client.

Figure 8-20. Network for Stateless Autoconfiguration Example.



Example 8-15 shows that IPv6 ICMPv6 ND debugging is enabled on R1 with the debug ipv6 nd command, and then its interface Fast Ethernet0/0 is configured with the ipv6 address autoconfig command. The interface creates a link-local address and verifies it with the DAD process. R1 then starts to send out RS messages, looking for autoconfiguration information. R2 is not yet configured, so the RS messages are not answered. The output of the show ipv6 interface command confirms that R1 has a link-local address, but not

a global unicast address. The blank output of the show ipv6 routers command confirms that there are no routers available to provide R1 with stateless autoconfiguration.

#### Example 8-15. Configuration and Verification of R1 for Stateless Autoconfiguration

Code View: Scroll / Show All

```
R1#debug ipv6 nd
```

ICMP Neighbor Discovery events debugging is on

```
R1#config t
```

Enter configuration commands, one per line. End with CNTL/Z.

```
R1(config)#interface fa0/0
```

```
R1(config-if)#ipv6 address autoconfig
```

```
R1(config-if)#
```

```
*Aug 13 14:02:02.172: ICMPv6-ND: Sending NS for FE80::219:56FF:FE2C:9F60 on FastEthernet0/0
```

```
*Aug 13 14:02:03.172: ICMPv6-ND: DAD: FE80::219:56FF:FE2C:9F60 is unique.
```

```
*Aug 13 14:02:03.172: ICMPv6-ND: Sending NA for FE80::219:56FF:FE2C:9F60 on FastEthernet0/0
```

```
*Aug 13 14:02:03.172: ICMPv6-ND: Address FE80::219:56FF:FE2C:9F60/10 is up on FastEthernet0/0
```

```
*Aug 13 14:02:05.172: ICMPv6-ND: Sending RS on FastEthernet0/0
```

```
*Aug 13 14:02:09.172: ICMPv6-ND: Sending RS on FastEthernet0/0
```

```
R1(config-if)#do show ipv6 interface fa0/0
```

FastEthernet0/0 is up, line protocol is up

IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60

No global unicast address is configured

Joined group address(es):

FF02::1

FF02::2

FF02::1:FF2C:9F60

MTU is 1500 bytes

ICMP error messages limited to one every 100 milliseconds

ICMP redirects are enabled

ND DAD is enabled, number of DAD attempts: 1

ND reachable time is 30000 milliseconds

```
R1(config-if)#do show ipv6 routers
```

```
R1(config-if)#
```

As shown in Example 8-16, on R2, IPv6 ICMPv6 ND debugging is enabled and then its Fast Ethernet 0/0 interface is configured with a global unicast address. In the debug output, notice that R2 adds the prefix and sends NS, DAD, and NA messages. However, it does not do any stateless autoconfiguration for R1. In the lower part of Example 8-16, the ipv6 unicast-routing command is configured on R2; this command is required for R2 to provide stateless autoconfiguration to R1. Immediately, R2 begins to send updates to the all-nodes multicast address FF02::1 on the Fast Ethernet 0/0 interface. This message includes the network prefix (2001:1:1001::/64) to be used by nodes to automatically configure themselves (as indicated by the "onlink autoconfig" notation).

#### Example 8-16. Configuration and Verification of R2 for Stateless Autoconfiguration

Code View: Scroll / Show All

```
R2#debug ipv6 nd
```

ICMP Neighbor Discovery events debugging is on

R2#**config t**

Enter configuration commands, one per line. End with CNTL/Z.

R2(config)#**interface fa0/0**

R2(config-if)#**ipv6 address 2001:1:1001::1/64**

R2(config-if)#

\*Aug 13 13:58:29.400: ICMPv6-ND: Adding prefix 2001:1:1001::1/64 to  
FastEthernet0/0

\*Aug 13 13:58:29.400: ICMPv6-ND: Sending NS for 2001:1:1001::1 on  
FastEthernet0/0

\*Aug 13 13:58:30.400: ICMPv6-ND: DAD: 2001:1:1001::1 is unique.

\*Aug 13 13:58:30.400: ICMPv6-ND: Sending NA for 2001:1:1001::1 on  
FastEthernet0/0

\*Aug 13 13:58:30.400: ICMPv6-ND: Address 2001:1:1001::1/64 is up on  
FastEthernet0/0

R2(config-if)#**exit**

R2(config)#**ipv6 unicast-routing**

R2(config)#

\*Aug 13 14:00:07.428: ICMPv6-ND: Sending RA to FF02::1 on FastEthernet0/0

\*Aug 13 14:00:07.428: ICMPv6-ND: MTU = 1500

\*Aug 13 14:00:07.428: ICMPv6-ND: prefix = 2001:1::/64 onlink autoconfig

\*Aug 13 14:00:07.428: ICMPv6-ND: 2592000/604800 (valid/preferred)

\*Aug 13 14:00:08.428: ICMPv6-ND: Received NA for  
2001:1:1001:0:219:56FF:FE2C:9F60 on FastEthernet0/0 from  
2001:1:1001:0:219:56FF:FE2C:9F60

The top of Example 8-17 illustrates the debug output on R1. R1 receives an RA, deletes some stale information, autoconfigures the address, and then does DAD for this address. The output of the show ipv6 interface command confirms that the interface has a global unicast address, and the prefix is the same as R2 advertised. The last line of this show command output lists the default router as R2.

Note

The 32-bit *valid lifetime* is the number of seconds that the prefix should be considered on-link and that autoconfigured addresses using the prefix can be used. The 32-bit *preferred lifetime* is the number of seconds that autoconfigured addresses using this prefix are preferred. Addresses configured via stateless autoconfiguration can be used until the preferred lifetime expires. If RAs stop coming, the preferred lifetime eventually expires and the address becomes "deprecated." New sessions will not use deprecated addresses but should choose "preferred" (nondeprecated) addresses, if available. However, existing sessions will continue to use the deprecated address. Eventually, the "valid lifetime" also runs out, and the deprecated address is removed from the interface, dropping any sessions that are still using the address.

Example 8-17. Stateless Autoconfiguration Debug Output on R1

Code View: Scroll / Show All

R1(config-if)#

\*Aug 13 14:13:41.612: ICMPv6-ND: Received NA for 2001:1:1001::1 on FastEthernet0/0  
from 2001:1:1001::1

\*Aug 13 14:15:18.640: ICMPv6-ND: Received RA from FE80::219:55FF:FEDF:AD22 on  
FastEthernet0/0

\*Aug 13 14:15:18.640: ICMPv6-ND: DELETE -> INCMP: FE80::219:55FF:FEDF:AD22

\*Aug 13 14:15:18.640: ICMPv6-ND: INCMP -> STALE: FE80::219:55FF:FEDF:AD22

\*Aug 13 14:15:18.640: ICMPv6-ND: Sending NS for 2001:1:1001:0:219:56FF:FE2C:9F60 on

```

FastEthernet0/0
*Aug 13 14:15:18.640: ICMPv6-ND: Autoconfiguring 2001:1:1001:0:219:56FF:FE2C:9F60 on
FastEthernet0/0
*Aug 13 14:15:19.640: ICMPv6-ND: DAD: 2001:1:1001::0:219:56FF:FE2C:9F60 is unique.
*Aug 13 14:15:19.640: ICMPv6-ND: Sending NA for 2001:1:1001:0:219:56FF:FE2C:9F60 on
FastEthernet0/0
*Aug 13 14:15:19.640: ICMPv6-ND: Address 2001:1:1001:0:219:56FF:FE2C:9F60/64 is up
on
FastEthernet0/0
R1(config-if)#do show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60
Global unicast address(es):
 2001:1:1001:0:219:56FF:FE2C:9F60, subnet is 2001:1:1001::/64 [PRE]
 valid lifetime 2591862 preferred lifetime 604662
Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF2C:9F60
MTU is 1500 bytes
ICMP error messages limited to one every 100 milliseconds
ICMP redirects are enabled
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
Default router is FE80::219:55FF:FEDF:AD22 on FastEthernet0/0
R1(config-if)#
R1(config-if)#do show ipv6 routers
Router FE80::219:55FF:FEDF:AD22 on FastEthernet0/0, last update 0 min
 Hops 64, Lifetime 1800 sec, AddrFlag=0, OtherFlag=0, MTU=1500
 HomeAgentFlag=0, Preference=Medium
 Reachable time 0 msec, Retransmit time 0 msec
 Prefix 2001:1:1001::/64 onlink autoconfig
 valid lifetime 2592000, preferred lifetime 604800
R1(config-if)#

```

The show ipv6 routers command output also shown in Example 8-17 illustrates that the default Router R2 is providing R1 with information is on R1's Fast Ethernet 0/0 interface, when the last update was, the prefix, and the valid lifetime and the preferred lifetime values.

Now that it is working, we can further understand the stateless autoconfiguration process by unraveling it to see what happens. First, the ipv6 unicast-routing is disabled on R2. The debug messages on both routers are shown in Example 8-18.

Example 8-18. Observing Stateless Autoconfiguration Removal

```

Code View: Scroll / Show All
R2(config)#no ipv6 unicast-routing
R2(config)#
*Aug 13 14:04:39.024: ICMPv6-ND: Sending Final RA on FastEthernet0/0

```

```

R1(config-if)#
*Aug 13 14:19:50.236: ICMPv6-ND: Received RA from FE80::219:55FF:FEDF:AD22
on
FastEthernet0/0
*Aug 13 14:19:50.236: ICMPv6-ND: zero lifetime, deleting
*Aug 13 14:19:50.236: ICMPv6-ND: STALE -> DELETE: FE80::219:55FF:FEDF:AD22

R1(config-if)#do show ipv6 routers

R1(config-if)#
R1(config-if)#do show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60
Global unicast address(es):
 2001:1:1001:0:219:56FF:FE2C:9F60, subnet is 2001:1:1001::/64 [PRE]
 valid lifetime 2591817 preferred lifetime 604617
Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF2C:9F60
MTU is 1500 bytes
ICMP error messages limited to one every 100 milliseconds
ICMP redirects are enabled
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
R1(config-if)#

```

R2 sends a final update as soon as the **no ipv6 unicast-routing** command is configured. R1 receives the update, which included a zero lifetime. R1 deletes the link-local address of R2.

The blank output of the **show ipv6 routers** command on R1 confirms the deletion. In the **show ipv6 interface** command output, notice that the default router configuration is not listed anymore. However, the host still has its global unicast address using the network prefix that was assigned by stateless autoconfiguration. When the address was assigned, the valid lifetime was 30 days (2,592,000 seconds) and the preferred time was 7 days (604,800 seconds). Therefore, the host retains its global address for 30 days, so long as the interface does not go down. If it does go down, the prefix assignment would be lost.

In Example 8-19, R1's Fast Ethernet 0/0 interface is shut down, and then enabled (with the **no shutdown** command). The results indicate that when the interface restarts, the NS, DAD, and NA messages are sent, for the link-link local address. R1 then starts to send RS messages, but they are not answered (because R2 is no longer configured to send RAs). The **show ipv6 interface** command confirms that the interface has a link-local address, but does not have a global unicast address.

Example 8-19. Shutting Down R1's Fast Ethernet 0/0 Interface

```

Code View: Scroll / Show All
R1(config-if)#shutdown
R1(config-if)#
*Aug 13 14:28:06.068: ICMPv6-ND: Address 2001:1:1001:0:219:56FF:FE2C:9F60/64 is
down
on FastEthernet0/0
*Aug 13 14:28:06.068: ICMPv6-ND: Address FE80::219:56FF:FE2C:9F60/10 is down on
FastEthernet0/0

```

```

*Aug 13 14:28:08.068: %LINK-5-CHANGED: Interface FastEthernet0/0, changed state to
administratively down
*Aug 13 14:28:09.068: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/0, changed state to down
R1(config-if)#no shutdown
*Aug 13 14:28:11.636: ICMPv6-ND: Sending NS for FE80::219:56FF:FE2C:9F60 on
FastEthernet0/0
*Aug 13 14:28:12.636: ICMPv6-ND: DAD: FE80::219:56FF:FE2C:9F60 is unique.
*Aug 13 14:28:12.636: ICMPv6-ND: Sending NA for FE80::219:56FF:FE2C:9F60 on
FastEthernet0/0
*Aug 13 14:28:12.636: ICMPv6-ND: Address FE80::219:56FF:FE2C:9F60/10 is up on
FastEthernet0/0
*Aug 13 14:28:13.624: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/0, changed stat
*Aug 13 14:28:14.636: ICMPv6-ND: Sending RS on FastEthernet0/0
R1(config-if)#do show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60
No global unicast address is configured
Joined group address(es):
FF02::1
FF02::2
FF02::1:FF2C:9F60
MTU is 1500 bytes
ICMP error messages limited to one every 100 milliseconds
ICMP redirects are enabled
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
R1(config-if)#

```

Notice the difference between when routers generate RS and RA messages:

- Routers configured with the **ipv6 unicast-routing** command generate ICMPv6 RA messages. They do not generate RS messages.
- Routers configured with the **ipv6 address auto-config** command, and not configured with the **ipv6 unicast-routing** command, generate RS messages only. They do not generate RA messages.

Note

Stateless autoconfiguration allows devices to “plug and play,” to connect themselves to the network without any configuration and without any servers (such as DHCP servers). DHCP Version 6 (DHCPv6), an updated version of DHCP for IPv4, can also be used to provide IPv6 addresses to devices.

Another option is stateless DHCPv6, which strikes a middle ground between stateless autoconfiguration and the thick-client approach of the stateful DHCPv6. Stateless DHCPv6 is also called DHCP-lite. For more details of stateless DHCPv6, see RFC 3736, *Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6*.

### Unicast Connectivity on Different Connection Types

This section examines in detail the processes used to connect IPv6 devices on broadcast multiaccess connections (for which Ethernet is used as an example), point-to-point connections, and point-to-multipoint connections (for which a Frame Relay NBMA network is used as an example).

#### Unicast Connectivity on Broadcast Multiaccess Links

As described earlier, the IPv4 ARP process to determine the MAC address-to-IPv4 address mapping has been replaced with the ICMPv6 neighbor discovery process for IPv6. For comparison, the results of the two



processes are displayed in Example 8-20, using the show ipv6 neighbors command for IPv6 and the show arp command for IPv4.

#### Example 8-20. IPv6 Uses ICMPv6 for Link-Layer Address Discovery

Code View: Scroll / Show All

R1#**show ipv6 neighbors**

| IPv6 Address             | Age | Link-layer Addr | State | Interface |
|--------------------------|-----|-----------------|-------|-----------|
| 2001:1::2                | 4   | 0019.55df.ad22  | STALE | Fa0/0     |
| FE80::219:55FF:FEDF:AD22 | 4   | 0019.55df.ad22  | STALE | Fa0/0     |

R1#**show arp**

| Protocol | Address      | Age (min) | Hardware Addr  | Type | Interface       |
|----------|--------------|-----------|----------------|------|-----------------|
| Internet | 172.16.100.1 | -         | 0019.562c.9f60 | ARPA | FastEthernet0/0 |
| Internet | 172.16.100.2 | 4         | 0019.55df.ad22 | ARPA | FastEthernet0/0 |

R1#

The difference between these two processes is that the ARP protocol is separate from the IP protocol in IPv4. ARP uses a broadcast link-layer destination address; the ARP messages are not sent in IP packets. In contrast, the ICMPv6 messages are not separate from the protocol; they are sent in IPv6 packets. Recall that the NS message is ICMPv6 type 135 and the NA message is ICMPv6 type 136. The NS messages use the solicited-node multicast address as the destination address; this address is displayed as part of the show ipv6 interface command output, as illustrated in Example 8-21. In the first command output, there are two solicited-node multicast addresses. The second one begins with FF02 and has the last six digits (2C:9F60) that match the last six digits of the link-local address. This is the solicited-node multicast address associated with the link-local address. Changing the link-local address results in changing this solicited node multicast address, as shown in the second output after the link-local address is changed. The solicited-node multicast address begins with FF02 and the last six digits (0B:000C, shortened to 0B:C) match the last six digits of the new link-local address. (Notice that the other solicited-node multicast address is associated with the global unicast address.)

#### Example 8-21. IPv6 Solicited-Node Multicast Address

Code View: Scroll / Show All

R1#**show ipv6 interface fa0/0**

FastEthernet0/0 is up, line protocol is up

IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60

Global unicast address(es):

2001:1::1, subnet is 2001:1::/64

Joined group address(es):

FF02::1

FF02::2

FF02::1:FF00:1

FF02::1:FF2C:9F60

MTU is 1500 bytes

ICMP error messages limited to one every 100 milliseconds

ICMP redirects are enabled

ND DAD is enabled, number of DAD attempts: 1

ND reachable time is 30000 milliseconds

R1#**config t**

Enter configuration commands, one per line. End with CNTL/Z.

R1(config)#**interface fa0/0**

R1(config-if)#**ipv6 address FE80::A:B:C link-local**

R1(config-if)#**do show ipv6 interface fa0/0**

FastEthernet0/0 is up, line protocol is up

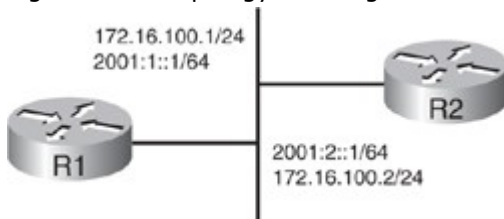
```

IPv6 is enabled, link-local address is FE80::A:B:C
Global unicast address(es):
 2001:1::1, subnet is 2001:1::/64
Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF00:1
 FF02::1:FF0B:C
MTU is 1500 bytes
ICMP error messages limited to one every 100 milliseconds
ICMP redirects are enabled
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
R1(config-if)#

```

Figure 8-21 illustrates a simple topology used to further examine the neighbor discovery process on a multiaccess network.

Figure 8-21. Topology For Neighbor Discovery over Multiaccess Network Example.



Example 8-22 illustrates that first a ping is sent from R1 to R2, and then the neighbor discovery information is displayed, showing both the global unicast address and the link-local address. By repeating the show ipv6 neighbors command, we observe that the global unicast address goes to a STALE state. The neighbor table on R2 indicates that both addresses are in a STALE state. The STALE state occurs when the specified address that was formerly in the REACH state has not been heard from within the time specified in the ipv6 nd reachable-time milliseconds command. The STALE state means that entry has not been used within in the reachable time. The REACH state means that the entry has been used in the reachable time.

Example 8-22. IPv6 Neighbor Discovery Information

Code View: Scroll / Show All

```
R1#ping 2001:1::2
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2001:1::2, timeout is 2 seconds:

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms

R1#

```
R1#show ipv6 neighbors
```

| IPv6 Address             | Age | Link-layer Addr | State | Interface |
|--------------------------|-----|-----------------|-------|-----------|
| 2001:1::2                | 0   | 0019.55df.ad22  | REACH | Fa0/0     |
| FE80::219:55FF:FEDF:AD22 | 0   | 0019.55df.ad22  | REACH | Fa0/0     |

R1#

```
R1#show ipv6 neighbors
```

| IPv6 Address             | Age | Link-layer Addr | State | Interface |
|--------------------------|-----|-----------------|-------|-----------|
| 2001:1::2                | 0   | 0019.55df.ad22  | REACH | Fa0/0     |
| FE80::219:55FF:FEDF:AD22 | 0   | 0019.55df.ad22  | REACH | Fa0/0     |

#### R1#show ipv6 neighbors

| IPv6 Address             | Age | Link-layer Addr | State | Interface |
|--------------------------|-----|-----------------|-------|-----------|
| 2001:1::2                | 0   | 0019.55df.ad22  | REACH | Fa0/0     |
| FE80::219:55FF:FEDF:AD22 | 0   | 0019.55df.ad22  | REACH | Fa0/0     |

#### R1#show ipv6 neighbors

| IPv6 Address             | Age | Link-layer Addr | State | Interface |
|--------------------------|-----|-----------------|-------|-----------|
| 2001:1::2                | 0   | 0019.55df.ad22  | STALE | Fa0/0     |
| FE80::219:55FF:FEDF:AD22 | 0   | 0019.55df.ad22  | REACH | Fa0/0     |

R1#

#### R2#show ipv6 neighbors

| IPv6 Address | Age | Link-layer Addr | State | Interface |
|--------------|-----|-----------------|-------|-----------|
| 2001:1::1    | 0   | 0019.562c.9f60  | STALE | Fa0/0     |
| FE80::A:B:C  | 0   | 0019.562c.9f60  | STALE | Fa0/0     |

R2#

Pinging a nonexistent address helps to illustrate the process used. In Example 8-23, the debug ipv6 nd command is entered and one ping is sent to a nonexistent IPv6 address 2001:1:10. Of course, the ping is unsuccessful. The state changes from DELETE to INCMP (incomplete). Three NS messages are sent, and then the state changes from INCMP to DELETE. The Layer 3 to Layer 2 address resolution was unsuccessful, resulting in the DELETE state. Example 8-23 also shows the ping again using the debug ipv6 packet detail command. This time an NS packet is observed, going to the solicited-node multicast address (FF02::1:FF00:10).

Example 8-23. Pinging a Nonexistent IPv6 Address

Code View: Scroll / Show All

R2#**debug ipv6 nd**

R2#**ping 2001:1::10 repeat 1**

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 2001:1::10, timeout is 2 seconds:

\*Aug 13 23:42:43.237: ICMPv6-ND: DELETE -> INCMP: 2001:1::10

\*Aug 13 23:42:43.237: ICMPv6-ND: Sending NS for 2001:1::10 on FastEthernet0/0

\*Aug 13 23:42:44.237: ICMPv6-ND: Sending NS for 2001:1::10 on FastEthernet0/0

Success rate is 0 percent (0/1)

R2#

\*Aug 13 23:42:45.237: ICMPv6-ND: Sending NS for 2001:1::10 on FastEthernet0/0

\*Aug 13 23:42:46.237: ICMPv6-ND: INCMP deleted: 2001:1::10

\*Aug 13 23:42:46.237: ICMPv6-ND: INCMP -> DELETE: 2001:1::10

R2#**undebug all**

R2#**debug ipv6 packet detail**

IPv6 unicast packet debugging is on (detailed)

R2#**ping 2001:1::10 repeat 1**

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 2001:1::10, timeout is 2 seconds:

\*Aug 13 23:44:08.721: IPv6: SAS picked source 2001:1::2 for 2001:1::10  
(FastEthernet0/0)

\*Aug 13 23:44:08.721: IPv6: source 2001:1::2 (local)

\*Aug 13 23:44:08.721: dest 2001:1::10 (FastEthernet0/0)

\*Aug 13 23:44:08.721: traffic class 0, flow 0x0, len 100+0, prot 58, hops  
64, originating

\*Aug 13 23:44:08.721: IPv6: source 2001:1::2 (local)

\*Aug 13 23:44:08.721: dest FF02::1:FF00:10 (FastEthernet0/0)

\*Aug 13 23:44:08.721: traffic class 224, flow 0x0, len 72+8, prot 58, hops  
255, originating

\*Aug 13 23:44:08.721: IPv6: Sending on FastEthernet0/0

\*Aug 13 23:44:08.725: IPv6: Encapsulation failed

\*Aug 13 23:44:08.725: IPv6: Resolving next hop 2001:1::10 on interface  
FastEthernet0/0

\*Aug 13 23:44:08.725: IPv6: Resolving next hop 2001:1::10 on interface  
FastEthernet0/0

Instead of relying on dynamic neighbor discovery, a static mapping between an IPv6 unicast address and a MAC address can be configured using the `ipv6 neighbor` command. Example 8-24 demonstrates the use of this command and the `show ipv6 neighbor` command to verify the configuration. The static entry does not have an age value and will be in a REACH state permanently.

Example 8-24. Configuring a Static IPv6-to-MAC Map

Code View: Scroll / Show All

R2(config)#**ipv6 neighbor 2001:1::10 fastEthernet 0/0 c000.0000.0000**

\*Aug 13 23:55:30.385: IPv6: source FE80::219:55FF:FEDF:AD22 (local)

\*Aug 13 23:55:30.385: dest FF02::1 (FastEthernet0/0)

\*Aug 13 23:55:30.385: traffic class 224, flow 0x0, len 104+1396, prot 58,  
hops 255, originating

\*Aug 13 23:55:30.385: IPv6: Sending on FastEthernet0/0)

R2(config)#**do show ipv6 neighbor**

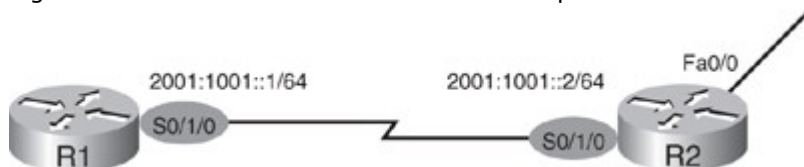
| IPv6 Address      | Age | Link-layer Addr       | State        | Interface    |
|-------------------|-----|-----------------------|--------------|--------------|
| 2001:1::1         | 25  | 0019.562c.9f60        | STALE        | Fa0/0        |
| <b>2001:1::10</b> | -   | <b>c000.0000.0000</b> | <b>REACH</b> | <b>Fa0/0</b> |
| FE80::A:B:C       | 24  | 0019.562c.9f60        | STALE        | Fa0/0        |

R2(config)#

Unicast Connectivity on Point-to-Point Links

This section examines IPv6 addresses on a point-to-point link. Figure 8-22 illustrates the topology used in this example. There is a point-to-point link between the routers.

Figure 8-22. Network for Point-to-Point Example.



By default, the Serial 0/1/0 interfaces use the Cisco HDLC encapsulation. Of course, serial interfaces do not have MAC addresses. As described earlier, MAC addresses are used in IPv6 dynamically created addresses. For serial interfaces, the router must therefore use a LAN MAC address. Example 8-25 illustrates debug output when an IPv6 address is configured on R2's interface. The router adds the prefix, and does NS and DAD for its link-local address. DAD confirms that the link-local address is unique. The same process confirms that the global unicast address is also unique. The output of the show ipv6 interface command confirms that the interface has a link-local and a global unicast address.

Example 8-25. Configuring IPv6 on R2's Serial 0/1/0 Interface

Code View: Scroll / Show All

R2#**debug ipv6 nd**

ICMP Neighbor Discovery events debugging is on

R2#**config t**

Enter configuration commands, one per line. End with CNTL/Z.

R2(config)#**interface s0/1/0**

R2(config-if)#**ipv6 address 2001:1001::2/64**

R2(config-if)#

\*Aug 13 16:21:40.299: ICMPv6-ND: Adding prefix 2001:1001::2/64 to Serial0/1/0

\*Aug 13 16:21:41.299: ICMPv6-ND: Sending NS for FE80::219:55FF:FE92:B212 on Serial0/1/0

\*Aug 13 16:21:42.299: ICMPv6-ND: DAD: FE80::219:55FF:FE92:B212 is unique.

\*Aug 13 16:21:42.299: ICMPv6-ND: Sending NA for FE80::219:55FF:FE92:B212 on Serial0/1/0

\*Aug 13 16:21:42.299: ICMPv6-ND: Address FE80::219:55FF:FE92:B212/10 is up on Serial0/1/0

\*Aug 13 16:21:42.299: ICMPv6-ND: Sending NS for 2001:1001::2 on Serial0/1/0

\*Aug 13 16:21:43.299: ICMPv6-ND: DAD: 2001:1001::2 is unique.

\*Aug 13 16:21:43.299: ICMPv6-ND: Sending NA for 2001:1001::2 on Serial0/1/0

\*Aug 13 16:21:43.299: ICMPv6-ND: Address 2001:1001::2/64 is up on Serial0/1/0

R2(config-if)#**do show ipv6 interface s0/1/0**

Serial0/1/0 is up, line protocol is up

IPv6 is enabled, link-local address is FE80::219:55FF:FE92:B212

Global unicast address(es):

2001:1001::2, subnet is 2001:1001::/64

Joined group address(es):

FF02::1

FF02::2

FF02::1:FF00:1

FF02::1:FF92:B212

MTU is 1500 bytes

ICMP error messages limited to one every 100 milliseconds

ICMP redirects are enabled

ND DAD is enabled, number of DAD attempts: 1

ND reachable time is 30000 milliseconds

Hosts use stateless autoconfig for addresses.

R2(config-if)#

Example 8-26 displays the output from the show interface command for the Fast Ethernet 0/0 interface, the only active LAN interface on this router. The MAC address of interface Fa0/0 is 0019.5592.B212. Notice that this MAC address was used to create the EUI-64 format interface ID for the Serial 0/1/0 link-local address FE80::219:55FF:FE92:B212. Recall that to transform the MAC address to the interface ID using EUI-64 format, FFFE is inserted into the middle of the MAC address, and the 7<sup>th</sup> bit of the first octet of the 64-bit interface ID is set to 1.

Example 8-26. R2's Serial 0/1/0 IPv6 Address Uses MAC Address from Fast Ethernet 0/0

```
R2(config-if)#do show interface fa0/0 | include addr
Hardware is Gt96k FE, address is 0019.5592.b212 (bia 0019.5592.b212)
R2(config-if)#do show ipv6 interface s0/1/0 | include link-local
IPv6 is enabled, link-local address is FE80::219:55FF:FE92:B212
R2(config-if)#
```

#### Note

The **include** keyword causes only those lines that start with the following text to be displayed in the command output. The text is case sensitive.

Although the link-local address is automatically assigned to serial interfaces, as we saw for LAN interfaces, best practice is to manually assign the address using the ipv6 address link-local command, to avoid confusion and to make troubleshooting easier. Example 8-27 illustrates doing this on Serial 0/1/0. The debug output shows that the old address goes down and the interface goes through the process to guarantee that the new address is unique on the link.

Example 8-27. R2's Serial 0/1/0 IPv6 Link-Local Address Can Be Manually Configured

```
Code View: Scroll / Show All
R2(config-if)#ipv6 address FE80::2 link-local
R2(config-if)#
*Aug 13 16:33:41.411: ICMPv6-ND: Address FE80::219:55FF:FE92:B212/10 is down
on
Serial0/1/0
*Aug 13 16:33:41.411: ICMPv6-ND: Sending NS for FE80::2 on Serial0/1/0
*Aug 13 16:33:42.411: ICMPv6-ND: DAD: FE80::2 is unique.
*Aug 13 16:33:42.411: ICMPv6-ND: Sending NA for FE80::2 on Serial0/1/0
*Aug 13 16:33:42.411: ICMPv6-ND: Address FE80::2/10 is up on Serial0/1/0
R2(config-if)#do show ipv6 interface s0/1/0 | include link-local
IPv6 is enabled, link-local address is FE80::2
R2(config-if)#
```

Similarly, IPv6 addressing on a link with PPP encapsulation is explored. As shown in Example 8-28, when the encapsulation is changed on the R2 interface, the address and interface go down, because the corresponding R1 interface is still using HDLC encapsulation. When the R1 encapsulation is also changed (as shown in the middle of Example 8-28), the R2 interface comes up, deletes the old link-local address, and performs the usual process to ensure that the address is unique on the link. Finally, the serial interface comes back up.

Example 8-28. IPv6 Addressing on a PPP Link

```
Code View: Scroll / Show All
R2(config-if)#encapsulation ppp
```

```

R2(config-if)#
*Aug 13 16:50:57.459: ICMPv6-ND: Address 2001:1001::2/10 is down on Serial0/1/0
*Aug 13 16:50:57.459: ICMPv6-ND: Address FE80::2 /10 is down on Serial0/1/0
*Aug 13 16:51:00.455: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0,
changed state to down

R1(config-if)#encapsulation ppp

R2(config-if)#
*Aug 13 16:51:19.627: ICMPv6-ND: DELETE -> INCMP: FE80::219:56FF:FE2C:9F60
*Aug 13 16:51:19.627: ICMPv6-ND: INCMP -> REACH: FE80::219:56FF:FE2C:9F60
*Aug 13 16:51:19.627: ICMPv6-ND: Sending NA for FE80::2 on Serial0/1/0
*Aug 13 16:51:19.627: ICMPv6-ND: Address FE80::2/10 is up on Serial0/1/0
*Aug 13 16:51:19.627: ICMPv6-ND: Sending NS for 2001:1001::2 on Serial0/1/0
*Aug 13 16:51:19.631: ICMPv6-ND: Received NA for FE80::219:56FF:FE2C:9F60 on
Serial0/1/0 from FE80::219:56FF:FE2C:9F60
*Aug 13 16:51:20.603: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0,
changed state to up
*Aug 13 16:51:20.627: ICMPv6-ND: DAD: 2001:1001::2 is unique.
*Aug 13 16:51:20.627: ICMPv6-ND: Sending NA for 2001:1001::2 on Serial0/1/0
*Aug 13 16:51:20.627: ICMPv6-ND: Address 2001:1001::2/64 is up on Serial0/1/0

```

Notice in Example 8-29 that the encapsulation is now PPP and that the PPP network control protocol (NCP) for IPv6, IPv6CP, is open. The example also illustrates the successful ping from R1 to R2's the link-local address. Notice that to ping the link-local address, the specific interface over which you want to ping must be entered, because the router cannot distinguish the interface by only knowing the address.

#### Note

A rather strange quirk of the **ping** command is that you must enter the full interface name (for example Serial 0/1/0), and not a shortcut (for example, s0/1/0).

#### Example 8-29. IPv6 Addressing on a PPP Link

```

Code View: Scroll / Show All
R2(config-if)#do show interface s0/1/0
Serial0/1/0 is up, line protocol is up
 Hardware is GT96K Serial
 Internet address is 10.10.10.2/24
 MTU 1500 bytes, BW 1544 Kbit/sec, DLY 20000 usec,
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation PPP, LCP Open
 Open: IPCP, CDPCP, IPV6CP, loopback not set
 Keepalive set (1 sec)
 Last input 00:00:26, output 00:00:00, output hang never
 Last clearing of "show interface" counters 00:01:51
 Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
 Queueing strategy: weighted fair
 Output queue: 0/1000/64/0 (size/max total/threshold/drops)
 Conversations 0/3/256 (active/max active/max total)
 Reserved Conversations 0/0 (allocated/max allocated)

```



```
R1#ping FE80::2
```

```
Output Interface: serial0/1/0
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to FE80::2, timeout is 2 seconds:
```

```
Packet sent with a source address of FE80::219:56FF:FE2C:9F60
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/15/16 ms
```

```
R1#
```

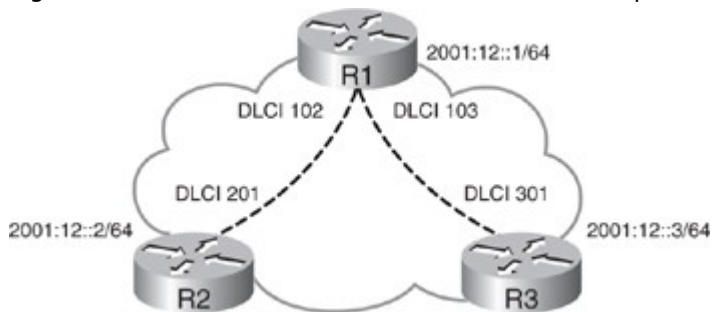
Point-to-point interfaces, with any encapsulation, can communicate in IPv6 by only using their link-local addresses. Therefore, these interfaces do not require a global unicast address to communicate. However, these interfaces would not be globally reachable; in other words, they would not be reachable from other parts of the network (because the link-local address is truly *local* in scope).

Unicast Connectivity on Point-to-Multipoint Links

Recall from Chapter 2, "Configuring the Enhanced Interior Gateway Routing Protocol," that Frame Relay is a switched WAN technology where virtual circuits (VCs) are created by a service provider (SP) through the network. Frame Relay allows multiple logical VCs to be multiplexed over a single physical interface. The VCs are typically PVCs that are identified by a data-link connection identifier (DLCI). Just as for IPv4, IPv6 addresses must be mapped to DLCIs. This mapping can be dynamic using IPv6 inverse ARP, or static using a frame-relay map interface configuration command. In this section, a point-to-multipoint Frame Relay connection is used to illustrate how an IPv6 address can be assigned on such a link and how the static Frame Relay mapping works.

Figure 8-23 provides an example topology in which R1 connects to R2 and R3 over a multipoint Frame Relay connection.

Figure 8-23. R1 Connects to R2 and R3 over a Multipoint Frame Relay Connection.



As shown in Example 8-30, the debug ipv6 packet and debug frame-relay packet commands are enabled on R1 and connectivity to R2's IPv6 address is tested with the ping command. The first four lines of debug output confirm that IPv6 routing is enabled. However, the next line states that there is an error related to encapsulation and that there is no map entry for that destination router. The blank output of the show frame-relay map command verifies the lack of map.

Example 8-30. IPv6 over Frame Relay Requires a Frame Relay Map Entry

```
Code View: Scroll / Show All
```

```
R1#debug ipv6 packet
```

```
IPv6 unicast packet debugging is on
```

```
R1#debug frame-relay packet
```

```
Frame Relay packet debugging is on
```

```
R1#
```

```
R1#ping 2001:12::2
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2001:12::2, timeout is 2 seconds:

```
*Aug 13 20:45:54.726: IPv6: SAS picked source 2001:12::1 for 2001:12::2 (Serial0/0/0)
*Aug 13 20:45:54.726: IPv6: source 2001:12::1 (local)
*Aug 13 20:45:54.726: dest 2001:12::2 (Serial0/0/0)
*Aug 13 20:45:54.726: traffic class 0, flow 0x0, len 100+0, prot 58, hops 64, originating
*Aug 13 20:45:54.726: Serial0/0/0:Encaps failed--no map entry link 79(IPV6)
*Aug 13 20:45:54.726: IPv6: Encapsulation failed
*Aug 13 20:45:54.726: IPv6: Resolving next hop 2001:12::2 on interface Serial0/0/0.
*Aug 13 20:45:56.726: IPv6: SAS picked source 2001:12::1 for 2001:12::2 (Serial0/0/0)
*Aug 13 20:45:56.726: IPv6: source 2001:12::1 (local)
*Aug 13 20:45:56.726: dest 2001:12::2 (Serial0/0/0)
*Aug 13 20:45:56.726: traffic class 0, flow 0x0, len 100+0, prot 58, hops 64, originating
*Aug 13 20:45:56.726: Serial0/0/0:Encaps failed--no map entry link 79(IPV6)
R1#show frame-relay map
```

R1#

Example 8-31 shows the configuration of the Frame Relay map on R1. After verifying the map entry exists, the ping is tried again. This time there is no encapsulation failure, but the ping still fails. Of course, this is because R2 needs a map back to R1; this configuration is also shown in the example. After both maps are configured and verified, debugging is disabled on R1, and the ping is tried again. This time the ping works.

Example 8-31. Adding Frame Relay Map Entries to R1 and R2

Code View: Scroll / Show All

```
R1(config)#interface s0/0/0
R1(config-if)#frame-relay map ipv6 2001:12::2 102
R1(config-if)#do show frame-relay map
Serial0/0/0 (up): ipv6 2001:12::2 dlci 102(0x66,0x1860), static,
 IETF, status defined, active
<output omitted>
```

R1#**ping 2001:12::2**

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2001:12::2, timeout is 2 seconds:

```
*Aug 13 20:49:54.194: IPv6: SAS picked source 2001:12::1 for 2001:12::2 (Serial0/0/0)
*Aug 13 20:49:54.194: IPv6: source 2001:12::1 (local)
*Aug 13 20:49:54.194: dest 2001:12::2 (Serial0/0/0)
*Aug 13 20:49:54.194: traffic class 0, flow 0x0, len 100+0, prot 58, hops 64, originating
```

```
*Aug 13 20:49:54.194: Serial0/0/0(o): dlci 102(0x1861), NLPID 0x38E(IPV6),
datagramsize 104
*Aug 13 20:49:54.194: IPv6: Sending on Serial0/0/0.
*Aug 13 20:49:56.194: IPv6: SAS picked source 2001:12::1 for 2001:12::2 (Se-
rial0/0/0)
*Aug 13 20:49:56.194: IPv6: source 2001:12::1 (local)
*Aug 13 20:49:56.194: dest 2001:12::2 (Serial0/0/0)
*Aug 13 20:49:56.194: traffic class 0, flow 0x0, len 100+0, prot 58, hops 64,
originating
*Aug 13 20:49:56.194: Serial0/0/0(o): dlci 102(0x1861), NLPID 0x38E(IPV6),
datagramsize 104
*Aug 13 20:49:56.194: IPv6: Sending on Serial0/0/0.
```

```
R2(config)#interface s0/0/0
R2(config-if)#frame-relay map ipv6 2001:12::1 201
R2(config-if)#
```

```
R1#undebg all
```

All possible debugging has been turned off

```
R1#ping 2001:12::2
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2001:12::2, timeout is 2 seconds:

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

R1#

There are two main differences between IPv4 and IPv6 Frame Relay configuration:

- In IPv6, a map is usually needed for link-local addresses and global unicast addresses. Although we didn't need link-local addresses yet in this example network, they are used by control-plane operations such as routing protocols, and as the next-hop address for routes installed in the routing table by an interior gateway protocol. If the next-hop link-local address is not reachable, remote networks will not be reachable.
- In IPv6, the **ipv6 unicast-routing** command must be configured when a routing protocol is used across the Frame Relay network for the routers to exchange updates.

The example is continued to examine these two differences. Example 8-32 illustrates the configuration used for R1 and R2, including manual IPv6 link-local addresses, and OSPFv3 commands. The "OSPFv3" section, later in this chapter, discusses OSPFv3 and its configuration in detail. For this example, the protocol is configured only to see the resulting addressing over the Frame Relay network.

Example 8-32. OSPFv3 Configuration for R1 and R2, for Frame Relay Example

Code View: Scroll / Show All

```
R1(config)#interface s0/0/0
R1(config-if)#ipv6 address FE80::1 link-local
R1(config-if)#exit
R1(config)#ipv6 unicast-routing
R1(config)#ipv6 router ospf 1
R1(config-rtr)#router-id 1.1.1.1
```

```

R1(config-rtr)#exit
R1(config)#interface s0/0/0
R1(config-if)#ipv6 ospf neighbor FE80::2
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#

R2(config)#interface s0/0/0
R2(config-if)#ipv6 address FE80::2 link-local
R2(config-if)#exit
R2(config)#ipv6 unicast-routing
R2(config)#ipv6 router ospf 1
R2(config-rtr)#router-id 2.2.2.2
R2(config-rtr)#exit
R2(config)#interface s0/0/0
R2(config-if)#ipv6 ospf neighbor FE80::1
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#

```

The OSPF adjacency between the routers will not be established until their link-local addresses are mapped to the appropriate DLCIs, as is done in Example 8-33, because routing protocols exchange hello and routing updates using these addresses. The broadcast keyword allows broadcasts and multicasts over the VC, thus permitting the use of dynamic routing protocols such as OSPFv3 over the VC. Notice that the OSPF adjacency is formed once both sides are configured with the appropriate link-layer address mapping. The configuration of the Serial 0/0/0 interface is also shown in this output.

Example 8-33. Frame Relay Mapping of Link-Local Addresses

```

Code View: Scroll / Show All
R2(config-if)#frame-relay map ipv6 FE80::1 201 broadcast
R2(config-if)#do show run interface s0/0/0
Building configuration...

Current configuration : 294 bytes
!
interface Serial0/0/0
 no ip address
 encapsulation frame-relay IETF
 ipv6 address 2001:12::2/64
 ipv6 address FE80::2 link-local
 ipv6 ospf neighbor FE80::1
 ipv6 ospf 1 area 0
 frame-relay map ipv6 2001:12::1 201
 frame-relay map ipv6 FE80::1 201 broadcast
 frame-relay lmi-type cisco
end
R2(config-if)#

R1(config-if)#frame-relay map ipv6 FE80::2 102 broadcast

```

```
R1(config-if)#
*Aug 13 22:03:41.922: %OSPFv3-5-ADJCHG: Process 1, Nbr 2.2.2.2 on Serial0/0/0
from LOADING to FULL, Loading Done
R1(config-if)#
```

## Routing IPv6 Traffic

This section describes the IPv6 routing protocols, and then explores each in detail. IPv6 policy routing and redistribution between the routing protocols is also examined.

### IPv6 Routing Protocols

IPv6 uses the same “longest-prefix match” route preference method that IPv4 CIDR uses. Updates to the existing IPv4 routing protocols were necessary for handling the longer IPv6 addresses and different header structures. Currently, the following updated routing protocols are available for use with IPv6:

- Static routes
- RIP new generation (RIPng) (defined in RFC 2080, *RIPng for IPv6*)
- OSPFv3 (defined in RFC 5340, *OSPF for IPv6*)
- Intermediate System-Intermediate System (IS-IS) for IPv6
- Enhanced Interior Gateway Routing Protocol (EIGRP) for IPv6
- Multiprotocol Border Gateway Protocol Version 4 (MP-BGP4 or MBGP) (defined in RFC 2545, *Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*, and RFC 4760, *Multiprotocol Extensions for BGP-4*)

Static routes, RIPng, OSPFv3, EIGRP for IPv6, and MBGP are described in the following sections.

#### Note

IS-IS for IPv6 is the same as IS-IS for IPv4, with the following extensions added: Two new Types, Lengths, Values (TLVs) (for IPv6 reachability and IPv6 interface address), and a new protocol identifier. IS-IS is not discussed further in this book.

Recall that the Cisco IOS **ipv6 unicast-routing** global configuration command for IPv6 enables IPv6 routing, and is required before any IPv6 routing protocol is configured.

### Static Routing

Static routes in IPv6 are used and configured similarly to static routes in IPv4. Static routes are useful in many situations, including for branch office to headquarters connectivity where the overhead of a routing protocol is not necessary.

#### Static Route Configuration and Verification Commands

IPv6 static routes can be configured with the `ipv6 route ipv6-prefix/prefix-length {ipv6-address | interface-type interface-number [ipv6-address]}` [administrative-distance] [administrative-multicast-distance | unicast | multicast] [next-hop-address] [tag tag] global configuration command. The parameters of this command are described in Table 8-2.

Table 8-2. *ipv6 route* Command Parameters

| Parameter                       | Description                                                                                    |
|---------------------------------|------------------------------------------------------------------------------------------------|
| ipv6-prefix/prefix-length       | Specifies the IPv6 network that is the destination of the static route, and its prefix length. |
| ipv6-address                    | Specifies the IPv6 address of the next hop that can be used to reach the specified network.    |
| interface-type interface-number | Specifies the interface through which the destination network can be reached.                  |
| administrative-distance         | Administrative distance. The default value is 1, which gives static routes precedence over any |

| Parameter                         | Description                                                                                                       |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------|
|                                   | other type of route except connected routes.                                                                      |
| administrative-multicast-distance | Specifies the administrative distance used when selecting this route for multicast Reverse Path Forwarding (RPF). |
| unicast                           | Specifies a route that must not be used in multicast RPF selection.                                               |
| multicast                         | Specifies a route that must not be populated in the unicast RIB.                                                  |
| Next-hop-address                  | Specifies the address of the next hop that can be used to reach the specified network.                            |
| <b>tag tag</b>                    | Tag value that can be used as a "match" value for controlling redistribution via route maps.                      |

Some of the many types of static routes that can be created with this command are as follows:

- A *directly attached static route* is created using only the *interface-type* and *interface-number* parameters. The specified interface must be up and have IPv6 enabled. The destination prefix is assumed to be reachable out of the specified interface; the packet's destination address is used as the next-hop address. An example is **ipv6 route 2001:CC1E::/32 serial 0/0/0**, which specifies that 2001:CC1E::/32 is reachable via the Serial 0/0/0 interface.
- A *recursive static route* is created using only the *next-hop address* parameter. An example is **ipv6 route 2001:CC1E::/32 2001:12::1**, which specifies that 2001:CC1E::/32 is reachable via the neighbor with address 2001:12::1. The router uses its routing table to determine the appropriate interface to use, which is the interface used to reach the next-hop address, so there must be such a route in the routing table. IPv6 recursive static routes are checked at 1-minute intervals. Therefore a recursive static route may take up to one minute to be inserted into the routing table once its next hop becomes valid. Likewise, it may take a minute or so for the route to disappear from the table if the next hop becomes invalid.
- A *fully specified static route* includes both the outgoing interface and the next hop address. An example is **ipv6 route 2001:CC1E::/32 serial 0/0/0 2001:12::1**, which specifies that 2001:CC1E::/32 is reachable via the neighbor with address 2001:12::1 on the Serial 0/0/0 interface. Fully specified static routes are used on multiaccess interfaces, on which there could be multiple devices so specifying only the interface is not sufficient.
- A *floating static route* can also be configured, which again operates similar to its IPv4 counterpart. The administrative distance of the static route is set to a value higher than the administrative distance of any dynamic routing protocol or other method used to reach a particular destination. The static routes function as a backup to the routes discovered by the dynamic routing protocol and will not be installed in the routing table unless the entry for the dynamic routing protocol is deleted, for example because of a link failure.

The `show ipv6 route [ipv6-address | ipv6-prefix/prefix-length | protocol | interface-type interface-number]` EXEC command displays the current contents of the IPv6 routing table. The parameters of this command are described in Table 8-3.

Table 8-3. *show ipv6 route* Command Parameters

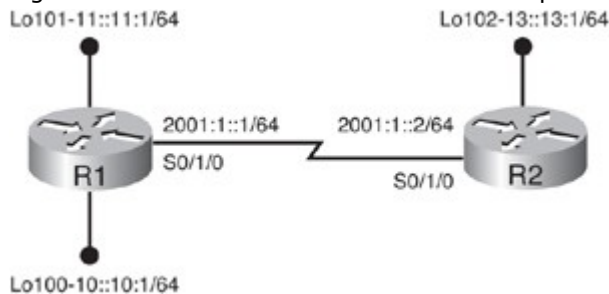
| Parameter                 | Description                                                                                                                       |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| ipv6-address              | Specifies that routing information for the specified address should be displayed.                                                 |
| ipv6-prefix/prefix-length | Specifies that routing information for the specified network should be displayed                                                  |
| protocol                  | Specifies that routes for the specified routing protocol should be displayed, using any of these keywords: <b>bgp,isis, ospf,</b> |

| Parameter                       | Description                                                                                                                                                                                    |
|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 | or <b>rip</b> . Or, specifies that routes for the specified type of route should be displayed, using any of these keywords: connected, local, static, or interface (for a specific interface). |
| interface-type interface-number | Specifies that only the given interface-specific routes are displayed.                                                                                                                         |

#### Static Route Configuration and Verification Example

Figure 8-24 shows a network used for a static route example. The R1 router could, for example, be at the headquarters of an organization, and the R2 router at a branch office.

Figure 8-24. Network for Static Route Example.



Example 8-34 shows the configuration and verification of the R1 and R2 routers. Notice that the ipv6 unicast-routing command is required even with static routes. R1 has a static route to the R2 loopback via its Serial 0/1/0 interface. R2 has a default static route, using the ::/0 prefix/length combination. The show ipv6 route static command verifies that the static route is in the routing table.

Example 8-34. Static Route Configuration and Verification

Code View: Scroll / Show All

```
R1(config)#ipv6 unicast-routing
```

```
R1(config)#ipv6 route 13::/64 s0/1/0
```

```
R1(config)#do show ipv6 route static
```

IPv6 Routing Table – 9 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

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 13::/64 [1/0]
```

```
via ::, Serial0/1/0
```

```
R1(config)#
```

```
R2(config)#ipv6 unicast-routing
```

```
R2(config)#ipv6 route ::/0 s0/1/0
```

```
R2(config)#do show ipv6 route static
```

IPv6 Routing Table – 7 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

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 ::, Serial0/1/0
R2(config)#
```

Example 8-35 illustrates that the static routes are working. We can ping from R1 to R2's loopback address, and ping from R2 to both of R1's loopback addresses.

Example 8-35. Static Route Verification

```
R1#ping 13::13:1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 13::13:1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/13/16 ms
R1#

R2#ping 11::11:1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 11::11:1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/15/16 ms
R2#

R2#ping 10::10:1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10::10:1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/12/16 ms
R2#
```

## RIPng

Similar to IPv4's RIP, RIPng is a distance vector routing protocol with a metric limit of 15 hops that uses split horizon and poison reverse to prevent routing loops. IPv6 features include the following:

- RIPng is based on IPv4 RIP Version 2 (RIPv2).
- RIPng uses IPv6 for transport.
- RIPng uses link-local addresses as source addresses.
- RIPng uses an IPv6 prefix and a next-hop IPv6 address.
- RIPng uses the multicast address FF02::9, the all RIPng routers multicast address, as the destination address for RIPng updates.
- The RIPng administrative distance is 120.
- RIPng updates are sent on UDP port 521.

## RIPng Configuration and Verification Commands

The **ipv6 rip *name* enable** interface configuration command enables a RIPng process on an interface. The *name* parameter is the name of the RIPng routing process; it is automatically created if it does not already exist.

The **ipv6 router rip *name*** global configuration command configures the RIPng routing process and enters router configuration mode. The *name* parameter is the name of the RIPng routing process.

The **no split-horizon** router configuration command disables the split horizon processing of RIPng updates.

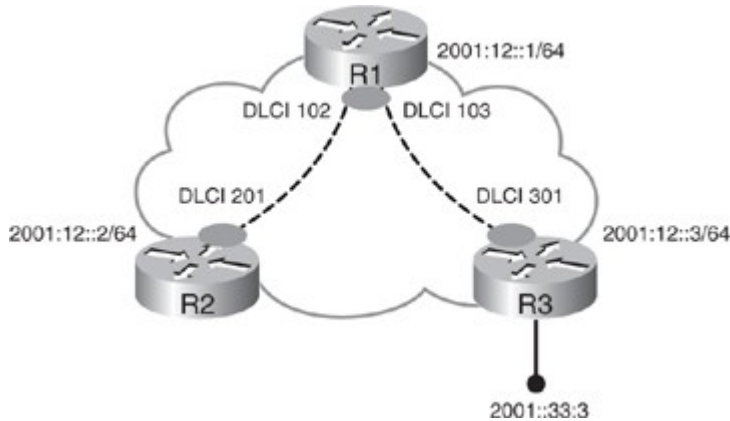
The **show ipv6 protocols [summary]** EXEC command displays the parameters and current state of the active IPv6 routing protocol processes. The **summary** keyword specifies that only the configured routing protocol process names are displayed.

The **debug ipv6 rip** [*interface-type interface-number*] EXEC command displays RIPng routing transaction debug messages. The *interface-type interface-number* specifies the interface about which to display debug messages.

#### RIPng Configuration and Verification Example

Figure 8-25 illustrates a network used for a RIPng example. The routers are connected via a Frame Relay network. The R1 router could, for example, be at the headquarters of an organization, and the R2 and R3 routers at branch offices.

Figure 8-25. Network for RIPng Example.



Example 8-36 illustrates, for the R1, R2, and R3 routers, the current configuration of the Frame Relay interfaces, and the addition of mapping statements for the global unicast addresses. Each router already had an IPv6 address configured on its Frame Relay interface. On Router R1, maps to both R2 and R3 are added. On Router R2 and R3, only a map to R1 is added.

Example 8-36. Global Unicast Frame Relay Map Configuration

Code View: Scroll / Show All

**R1#show run interface s0/0/0**

Building configuration...

Current configuration : 132 bytes

!

interface Serial0/0/0

no ip address

encapsulation frame-relay IETF

ipv6 address 2001:12::1/64

frame-relay lmi-type cisco

end

R1#

**R1#config t**

R1(config)#**interface s0/0/0**

R1(config-if)#**frame-relay map ipv6 2001:12::2 102**

R1(config-if)#**frame-relay map ipv6 2001:12::3 103**

R1(config-if)#

**R2(config)#do show run interface s1/1.7**

Building configuration...

Current configuration : 80 bytes

!

interface Serial1/1.7 multipoint

ipv6 address 2001:12::2/64

cdp enable

```

end
R2(config)#interface s1/1.7
R2(config-subif)#frame-relay map ipv6 2001:12::1
201
R2(config-subif)#

R3(config)#do show run interface s1/1.7
Building configuration...
Current configuration : 80 bytes
!
interface Serial1/1.7 multipoint
 ipv6 address 2001:12::3/64
 cdp enable
end
R3(config)#interface s1/1.7
R3(config-subif)#frame-relay map ipv6 2001:12::1
301
R3(config-subif)#

```

Example 8-37 shows the results of the connectivity verification between the sites. R1 can ping both R2 and R3; R2 and R3 can ping R1.

Example 8-37. Verification of Connectivity Between Sites

```

Code View: Scroll / Show All
R1#ping 2001:12::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:12::2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
R1#ping 2001:12::3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:12::3, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
R1#

R2#ping 2001:12::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:12::1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms
R2#

R3#ping 2001:12::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:12::1, timeout is 2 seconds:

```

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms  
R3#

However, there is no connectivity between the LANs behind the routers. In this network, RIPng will be used to provide connectivity. Like all the IPv6 routing protocols, RIPng uses link-local addresses. Recall that on a Frame Relay network we must configure maps for the link-local addresses, not only for the global unicast addresses. The link-local addresses for each interface can be obtained using the show ipv6 interface command on destination routers; simply copy and paste the addresses into the sending router configuration. Example 8-38 illustrates the configuration of the maps to the link-local addresses, on all three routers.

Example 8-38. Configuration of Link-Local Address Maps

```
R1(config)#interface s0/0/0
R1(config-if)#frame-relay map ipv6 FE80::250:73FF:FE3D:6A20
103
R1(config-if)#frame-relay map ipv6 FE80::2B0:64FF:FE33:FB60
102
R1(config-if)#

R2(config)# interface s1/1.7
R2(config-subif)#frame-relay map ipv6 FE80::219:56FF:FE2C:9F60
201
R2(config-subif)#

R3(config)# interface s1/1.7
R3(config-subif)#frame-relay map ipv6 FE80::219:56FF:FE2C:9F60
301
R3(config-subif)#
```

The next step is to enable RIPng, as shown in Example 8-39. The ipv6 unicast-routing command is configured first. A RIPng process called RIPTag is next created on the serial interface of each router; notice that this command is entered on an interface. However, it causes the global RIPTag process to be created automatically. (In other words, the process does not have to be created separately in global configuration mode.) The output of the show ipv6 protocols command confirms that the RIPng process RIPTag is configured.

Example 8-39. Configuring RIPng

```
Code View: Scroll / Show All
R1(config)#ipv6 unicast-routing
R1(config)#interface s0/0/0
R1(config-if)#ipv6 rip RIPTag ?
 default-information Configure handling of default route
 enable Enable/disable RIP routing
 metric-offset Adjust default metric increment
 summary-address Configure address summarization
R1(config-if)#ipv6 rip RIPTag enable
R1(config-if)#do show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
```

IPv6 Routing Protocol is "rip RIPTag"

Interfaces:

Serial0/0/0

Redistribution:

None

R1(config-if)#

R2(config)#**ipv6 unicast-routing**

R2(config)#**interface s1/1.7**

R2(config-subif)#**ipv6 rip RIPTag enable**

R2(config-subif)#

R3(config)#**ipv6 unicast-routing**

R3(config)# **interface s1/1.7**

R3(config-subif)#**ipv6 rip RIPTag enable**

R3(config-subif)#**do show ipv6 protocols**

IPv6 Routing Protocol is "connected"

IPv6 Routing Protocol is "static"

IPv6 Routing Protocol is "rip RIPTag"

Interfaces:

Serial1/1.7

Redistribution:

None

R3(config-subif)#

To confirm whether RIPng is working correctly, examine the debug ipv6 rip output on R1 shown in Example 8-40. Notice R1 is only sending RIPng messages; it is not receiving any messages from other routers.

Example 8-40. Verifying RIPng on R1

Code View: Scroll / Show All

R1#**debug ipv6 rip**

RIP Routing Protocol debugging is on

R1#

\*Aug 14 04:19:12.908: **RIPng: Sending multicast update on Serial0/0/0 for RIPTag**

\*Aug 14 04:19:12.908: src=FE80::219:56FF:FE2C:9F60

\*Aug 14 04:19:12.908: **dst=FF02::9 (Serial0/0/0)**

\*Aug 14 04:19:12.908: sport=521, dport=521, length=32

\*Aug 14 04:19:12.908: command=2, version=1, mbz=0, #rte=1

\*Aug 14 04:19:12.908: tag=0, metric=1, prefix=2001:12::/64

\*Aug 14 04:19:40.524: **RIPng: Sending multicast update on Serial0/0/0 for RIPTag**

\*Aug 14 04:19:40.524: src=FE80::219:56FF:FE2C:9F60

\*Aug 14 04:19:40.524: **dst=FF02::9 (Serial0/0/0)**

\*Aug 14 04:19:40.524: sport=521, dport=521, length=32

\*Aug 14 04:19:40.524: command=2, version=1, mbz=0, #rte=1

\*Aug 14 04:19:40.524: tag=0, metric=1, prefix=2001:12::/64

To understand why RIPng is not working, recall that RIPng uses multicast packets; notice the destination address FF02::9 in Example 8-40. When configuring the Frame Relay maps, the broadcast keyword was not specified; this is the keyword that allows both broadcast and multicast packets to go across the Frame Relay network. Without this keyword, multicast and broadcast packets do not go out. Therefore, the RIPng packets are not going across the PVCs, even though the debug output says they are going out the interface. Example 8-41 illustrates the output of the show frame-relay map command on R1; observe that there is no broadcast indicated. Example 8-41 also shows the corrected configuration of the Frame Relay maps on the three routers.

Example 8-41. Configuring Frame Relay Maps with the *broadcast* Keyword

Code View: Scroll / Show All

R1#**show frame-relay map**

Serial0/0/0 (up): ipv6 FE80::250:73FF:FE3D:6A20 dlci 103(0x67, 0x1870), static,  
IETF, status defined, active

Serial0/0/0 (up): ipv6 FE80::2B0:64FF:FE33:FB60 dlci 102(0x66, 0x1860), static,  
IETF, status defined, active

Serial0/0/0 (up): ipv6 2001:12::2 dlci 102(0x66, 0x1860), static,  
IETF, status defined, active

Serial0/0/0 (up): ipv6 2001:12::3 dlci 103(0x67, 0x1870), static,  
IETF, status defined, active

R1#**config t**

R1(config)#**interface s0/0/0**

R1(config-if)#**frame-relay map ipv6 FE80::250:73FF:FE3D:6A20 103 broadcast**

R1(config-if)#**frame-relay map ipv6 FE80::2B0:64FF:FE33:FB60 102 broadcast**

R1(config-if)#

R2(config-subif)#**frame-relay map ipv6 FE80::219:56FF:FE2C:9F60 201  
broadcast**

R3(config-subif)#**frame-relay map ipv6 FE80::219:56FF:FE2C:9F60 301  
broadcast**

Note that the **broadcast** keyword is only required on the Frame Relay maps for the link-local addresses for RIPng (and other routing protocols) to function correctly, because these protocols use the link-local addresses to communicate between routers. This keyword is not required on the Frame Relay maps for the global unicast addresses, because they are not used by the routing protocols. (The keyword may be required for other traffic, but not for the routing protocols.)

Example 8-42 displays the output of the debug ipv6 rip command again on R1; RIPng updates are now being sent and received.

Example 8-42. Debug Output

Code View: Scroll / Show All

R1#**debug ipv6 rip**

R1#

RIP Routing Protocol debugging is on

\*Aug 14 04:21:48.508: RIPng: response received from FE80::2B0:64FF:FE33:FB60 on  
Serial0/0/0 for RIPTag

\*Aug 14 04:21:48.508: src=FE80::2B0:64FF:FE33:FB60 (Serial0/0/0)

```

*Aug 14 04:21:48.508: dst=FF02::9
*Aug 14 04:21:48.508: sport=521, dport=521, length=32
*Aug 14 04:21:48.508: command=2, version=1, mbz=0, #rte=1
*Aug 14 04:21:48.512: tag=0, metric=1, prefix=2001:12::/64
*Aug 14 04:21:59.276: RIPng: Sending multicast update on Serial0/0/0 for RIPTag
*Aug 14 04:21:59.276: src=FE80::219:56FF:FE2C:9F60
*Aug 14 04:21:59.276: dst=FF02::9 (Serial0/0/0)
*Aug 14 04:21:59.276: sport=521, dport=521, length=32
*Aug 14 04:21:59.276: command=2, version=1, mbz=0, #rte=1
*Aug 14 04:21:59.276: tag=0, metric=1, prefix=2001:12::/64

```

The R3 loopback interface also must be configured to participate in the RIPTag process, for it to be advertised by R3. Example 8-43 shows this configuration. Also shown is the show ipv6 route rip output on R1, in which the R3 loopback interface address is visible. A ping from R1 to this loopback address is successful. However, R2's routing table does not contain the route to this address.

Example 8-43. Adding R3's Loopback Interface to RIPng

Code View: Scroll / Show All

```
R3(config)#interface loopback 103
```

```
R3(config-if)#ipv6 rip RIPTag enable
```

```
R3(config-if)#
```

```
R1#show ipv6 route rip
```

IPv6 Routing Table – 9 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 2001:33::/64 [120/2]
 via FE80::250:73FF:FE3D:6A20, Serial0/0/0
```

```
R1#ping 2001:33::3
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2001:33::3, timeout is 2 seconds:

```
!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/60/68 ms

```
R1#
```

```
R2#show ipv6 route rip
```

IPv6 Routing Table – 4 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R2#
```



R2 is not learning about the R3 loopback address because of the split-horizon feature of RIPng. As a distance vector protocol, RIPng uses split horizon to help prevent routing loops. The split-horizon rule states that a router will not send out a route learned on an interface (such as the route to R3's loopback that R1 learned on its Serial 0/0/0 interface) out of the same interface (such as to R2). Using subinterfaces is the best way to fix this problem. Alternatively, split horizon can be turned off. Here, we use the latter fix (but understand that routing loops could potentially be introduced). The no split-horizon command is configured on R1, which fixes the problem, as verified in Example 8-44.

Example 8-44. Turning Off Split Horizon on R1

```
R1(config)#ipv6 router rip RIPTag
R1(config-rtr)#no split-horizon
R1(config-rtr)#

R2#show ipv6 route rip
IPv6 Routing Table - 5 entries
Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP
 U – Per-user Static route
 I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary
 O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2
 ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2
R 2001:33::/64 [120/3]
 via FE80::219:56FF:FE2C:9F60, Serial1/1.7
R2#ping 2001:33::3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:33::3, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5) round-trip min/avg/max = 140/141/144ms
R2#
```

### OSPFv3

OSPFv3 is a new protocol implementation for IPv6. It uses the same mechanisms as OSPFv2, but is a major rewrite of the internals of the protocol.

OSPFv3 distributes (transports) IPv6 prefixes and runs directly over IPv6.

If both OSPFv2 and OSPFv3 are configured on a router, they run completely separate from each other and run a separate shortest path first (SPF) instance. In other words, the two protocols are like “ships in the night,” passing without knowing of the other’s existence.

OSPFv3 includes the following IPv6-specific features:

- Every OSPFv2 IPv4-specific semantic is removed.
- Uses 128-bit IPv6 addresses.
- Uses link-local addresses as source addresses.
- Multiple addresses and OSPF instances per interface are permitted.
- Supports authentication (using IPsec).
- Runs over a link rather than a subnet.

Recall that OSPF is link-state routing protocol. A link is an interface on a networking device, and a link-state protocol makes its routing decisions based on the states of the links that connect source and destination devices. The state of a link is a description of the interface and its relationship to its neighboring networking devices.

For OSPFv3, the interface information includes the IPv6 prefix of the interface, the network mask, the type of network it is connected to, the routers connected to the network, and so forth. This information is propagated in various types of link-state advertisements (LSAs). A router’s collection of LSA data is stored in

a link-state database (LSDB). The contents of the database, when subjected to Dijkstra's algorithm, result in the creation of the OSPF routing table.

#### Similarities Between OSPFv2 and OSPFv3

Although most of the algorithms of OSPFv2 are the same as those of OSPFv3, some changes have been made in OSPFv3, particularly to handle the increased address size in IPv6 and the fact that OSPFv3 runs directly over IPv6. The similarities between OSPFv3 and OSPFv2 include the following:

- OSPFv3 uses the same basic packet types as OSPFv2, as shown in Table 8-4—hello, database description (DBD) (also called database description packets [DDP]), link-state request (LSR), link-state update (LSU), and link-state acknowledgment (LSAck). Some of the fields within the packets have changed.

Table 8-4. OSPFv3 Packet Types

| Packet Type | Name  |
|-------------|-------|
| 1           | Hello |
| 2           | DBD   |
| 3           | LSR   |
| 4           | LSU   |
| 5           | LSAck |

- The mechanisms for neighbor discovery and adjacency formation are identical.
- OSPFv3 operation over NBMA topologies is the same as OSPFv2. The RFC-compliant nonbroadcast and point-to-multipoint modes are supported, and OSPFv3 also supports the Cisco modes such as point-to-point and broadcast.
- LSA flooding and aging are the same.

All the optional capabilities of OSPFv2, including on-demand circuit support, stub areas and not-so-stubby areas (NSSAs), and the extensions to Multicast OSPF (MOSPF), are also supported in OSPFv3. All areas must be connected to area 0, unless virtual links are configured, the same as in OSPFv2.

#### Differences Between OSPFv2 and OSPFv3

Because OSPFv2 is heavily dependent on the IPv4 address for its operation, changes were necessary in the OSPFv3 protocol to support IPv6, as outlined in RFC 5340. Some of the notable changes include platform-independent implementation, protocol processing per-link rather than per-node, explicit support for multiple instances per link, and changes in authentication and packet format.

Like RIPng, OSPFv3 uses IPv6 for transport and uses link-local addresses as source address.

All OSPFv3 packets have a 16-byte header, in comparison to OSPFv2's 24-byte header. The two headers are illustrated in Figure 8-26.

Figure 8-26. OSPFv2 and OSPFv3 Packet Headers.

[View full size image]

| OSPFv2 Header  |      |                     | OSPFv3 Header |      |               |
|----------------|------|---------------------|---------------|------|---------------|
| Version        | Type | Packet Length       | Version       | Type | Packet Length |
| Router ID      |      |                     | Router ID     |      |               |
| Area ID        |      |                     | Area ID       |      |               |
| Checksum       |      | Authentication Type | Checksum      |      | Instance ID   |
| Authentication |      |                     | 0             |      |               |
| Authentication |      |                     |               |      |               |

OSPFv2 does not define or allow for multiple instances per link, although similar functionality can be implemented by using other mechanisms such as subinterfaces. In contrast, OSPFv3 has explicit support for multiple instances per link through the instance ID field in the packet header. This feature allows separate routing domains, each running OSPF, to use a common link. A single link could belong to multiple areas. Two instances need to have the same instance ID to communicate with each other. By default, the instance ID is 0, and it is increased for any additional instances.

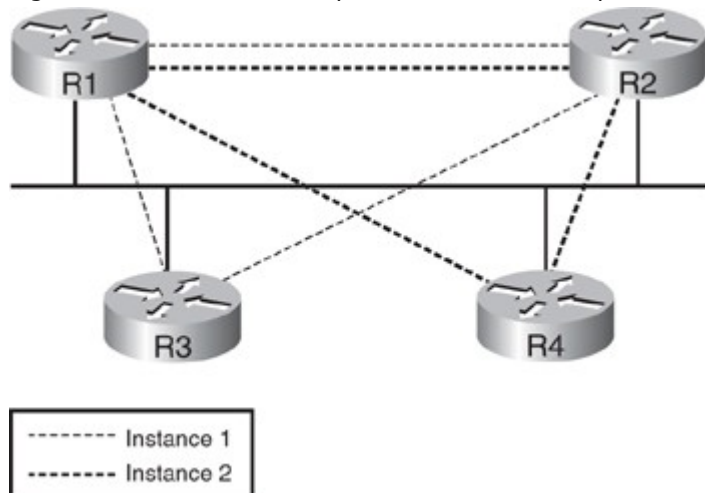
Authentication is no longer part of OSPFv3; it is now the job of IPv6 to make sure the right level of authentication is in use.

OSPFv3 uses IPv6 link-local addresses to identify the OSPFv3 adjacency neighbors.

OSPFv2 is primarily concerned with the *subnet* (or prefix) on which it is operating, whereas OSPFv3 is concerned with the *links* to which the router is connected. As discussed, IPv6 uses the term *link* to indicate a communication facility or medium over which nodes can communicate at the link layer; OSPF interfaces connect to links instead of to IP subnets. Multiple IPv6 subnets can be assigned to a single link, and two nodes can talk directly over a single link, even if they do not share a common IPv6 subnet (IPv6 prefix), because they use the link-local addresses, rather than the global unicast addresses, to communicate. OSPFv3 therefore runs per-link instead of the IPv4 behavior of per-IP-subnet, and the terms *network* and *subnet* are generally replaced by the term *link*. This change affects the receiving of OSPFv3 protocol packets, and the contents of hello packets and network LSAs.

Figure 8-27 illustrates an example of two instances of OSPFv3 running on the same physical network segment, using two different IPv6 prefixes.

Figure 8-27. OSPFv3 Runs per Link Rather Than per IP Subnet.



The multicast addresses used by OSPFv3 are as follows:

- **FF02::5**— This address represents all OSPFv3 routers on the link-local scope. It is equivalent to 224.0.0.5 in OSPFv2.
- **FF02::6**— This address represents all designated routers (DRs) on the link-local scope. It is equivalent to 224.0.0.6 in OSPFv2.

Address semantics that were in OSPFv2 have been removed in OSPFv3, as follows:

- IPv6 addresses are not present in the OSPFv3 packet header (rather they are part of payload information).
- OSPFv3 router LSAs and network LSAs do not carry IPv6 addresses.
- The router ID, area ID, and link-state ID remain at 32 bits and are written in an IPv4-address format (dotted decimal).
- The DR and backup designated router (BDR) are now identified by their router ID, not by their IP address.

For security, OSPFv3 uses IPv6 AH and ESP extension headers rather than the variety of authentication and confidentiality security mechanisms defined in OSPFv2.

OSPFv3 Configuration and Verification Commands

Many of the OSPFv3 commands are similar to their OSPFv2 counterparts, with the **ip** keyword replaced by the **ipv6** keyword.

One difference between OSPFv2 and OSPFv3 configuration is the way that IPv6 networks that are part of the OSPFv3 network are identified. The `network` area command used in OSPFv2 is not used in OSPFv3. Rather, in OSPFv3, interfaces are directly configured to specify which IPv6 networks are part of the OSPFv3 network. OSPFv3 is enabled on each interface, using the `ipv6 ospf process-id area area-id [instance instance-id]` interface configuration command. The parameters of this command are described in Table 8-5. In OSPFv3, all addresses on an interface are included by default. There is no limit to the number of `ipv6 ospf area` commands you can use on the router.

Table 8-5. *ipv6 ospf area* Command Parameters

| Parameter                          | Description                                                                                                                                                                                                                                                                                                                    |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>process-id</code>            | Internal identifier for the OSPF process that is locally assigned and can be any positive integer. This is the same number used when enabling the OSPF routing process.                                                                                                                                                        |
| <b>area</b> <i>area-id</i>         | Specifies the area that is to be associated with the OSPF interface.                                                                                                                                                                                                                                                           |
| <b>instance</b> <i>instance-id</i> | (Optional) Instance identifier. An OSPF instance (also known as an OSPF process) can be considered a logical router running OSPF in a physical router. Use the <i>instance-id</i> to control selection of other routers as neighboring routers. The router becomes neighbors only with routers that have the same instance ID. |

There is a separate native IPv6 router mode under which OSPFv3 parameters are defined. The **ipv6 router ospf process-id** global configuration mode enters OSPFv3 router configuration mode. The *process-id* parameter identifies a unique OSPFv3 process local to the router and can be any positive integer.

Note

The **ipv6 router ospf process-id** global configuration command also places you in router configuration mode, which for OSPFv3 is identified by the Router(config-rtr)# prompt, not the Router(config-router)# prompt that is used by OSPFv2 and other IPv4 routing protocols.

The **router-id {ip-address}** router configuration command defines a router ID for OSPFv3. The *ip-address* is a number in IPv4 address format; in other words, it is a 32-bit number in dotted-decimal format. The router ID must be unique on each router.

Note

The documentation for the **router-id** command states that the router ID could alternatively be an IPv6 address, but this is not the case on the routers.

The router ID selection process is the same as for OSPFv2. If the router ID is explicitly configured, it is used. Alternatively, if there is a loopback interface configured with an IPv4 address, the highest such IPv4 address is chosen as the router ID. If neither of those conditions exist, then if an IPv4 address exists on an interface when OSPFv3 is enabled, that IPv4 address is used as the router ID. For OSPFv3, if there are no IPv4 addresses configured, the router ID must be explicitly configured.

The OSPF priority, used in DR election, can be changed using the **ipv6 ospf priority number-value** interface configuration command. The *number-value* can range from 0 to 255; the default is 1. The router with the higher router priority takes precedence in an election. If there is a tie, the router with the higher router ID takes precedence. A router with a router priority set to zero is ineligible to become the DR or BDR.

The OSPF cost of sending a packet on an interface can be specified using the **ipv6 ospf cost interface-cost** interface configuration command. The *interface-cost* can be a value in the range from 1 to 65535. The default cost is related to the bandwidth of the interface, the same as it is for OSPF for IPv4.

As for OSPFv2, if interfaces that are faster than 100 Mbps are being used, use the **auto-cost reference-bandwidth ref-bw** router configuration command on all routers in the network to ensure accurate route calculations. The *ref-bw* parameter is the reference bandwidth in megabits per second. The range is from 1 to 4,294,967; the default is 100.

The **area area-id stub [no-summary]** router configuration command defines an area as a stub area, just as it does for IPv4. The *area-id* parameter identifies the area. The **no-summary** parameter, configured on the Area Border Router (ABR) only, indicates that the area is a totally stub area.

The **area area-id range ipv6-prefix /prefix-length [advertise | not-advertise] [cost cost]** router configuration command summarizes routes at an area boundary. Notice the similarity to the corresponding IPv4 command. Table 8-6 describes the parameters of this command.

Table 8-6. *area range* Command Parameters

| Parameter                 | Description                                                                                                                                                                                   |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| area-id                   | Specifies the area about which routes are to be summarized.                                                                                                                                   |
| ipv6-prefix/prefix-length | Specifies the IPv6 address and prefix length for the range of addresses in the summary route.                                                                                                 |
| Advertise                 | (Optional) Sets the address range status to advertise, and generates a type 3 summary LSA.                                                                                                    |
| not-advertise             | (Optional) Sets the address range status to DoNotAdvertise. The type 3 summary LSA is suppressed, and the component networks remain hidden from other networks.                               |
| <b>cost</b> <i>cost</i>   | (Optional) Specifies the metric or cost for this summary route, which is used during OSPF SPF calculation to determine the shortest paths to the destination. The value can be 0 to 16777215. |

The cost of the summarized routes is the *highest* cost of the routes being summarized. For example, consider the following routes, with the cost highlighted:

```
OI 2001:0DB8:0:0:7::/64 [110/20]
via FE80::A8BB:CCFF:FE00:6F00, FastEthernet0/0
OI 2001:0DB8:0:0:8::/64 [110/100]
via FE80::A8BB:CCFF:FE00:6F00, FastEthernet0/0
OI 2001:0DB8:0:0:9::/64 [110/20]
via FE80::A8BB:CCFF:FE00:6F00, FastEthernet0/0
```

If they are summarized, they become one route, as follows:

```
OI 2001:0DB8::/48 [110/100]
via FE80::A8BB:CCFF:FE00:6F00, FastEthernet0/0
```

The **clear ipv6 ospf** [*process-id*] {**process** | **force-spf** | **redistribution** | **counters** [**neighbor** [*neighbor-interface* | *neighbor-id*]]} EXEC command triggers SPF recalculation and repopulation of the RIB.

The show ipv6 ospf [*process-id*] [*area-id*] neighbor [*interface-type interface-number*] [*neighbor-id*] [*detail*] EXEC command displays OSPFv3 neighbor information. Table 8-7 describes the parameters of this command.

Table 8-7. *show ipv6 ospf neighbor* Command Parameters

| Parameter                       | Description                                                                                                     |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------|
| process-id                      | Specifies the internal identifier for the OSPF process that is locally assigned and can be any positive integer |
| area-id                         | Specifies the area for which information is to be displayed                                                     |
| interface-type interface-number | Specifies the interface type and number for which information is to be displayed                                |
| neighbor-id                     | Specifies the neighbor ID for which information is to be displayed                                              |
| detail                          | Displays detailed information about all neighbors                                                               |

The show ipv6 ospf [*process-id*] [*area-id*] interface [*type number*] [*brief*] EXEC command displays OSPFv3 interface information. Table 8-8 describes the parameters of this command.

Table 8-8. *show ipv6 ospf interface* Command Parameters

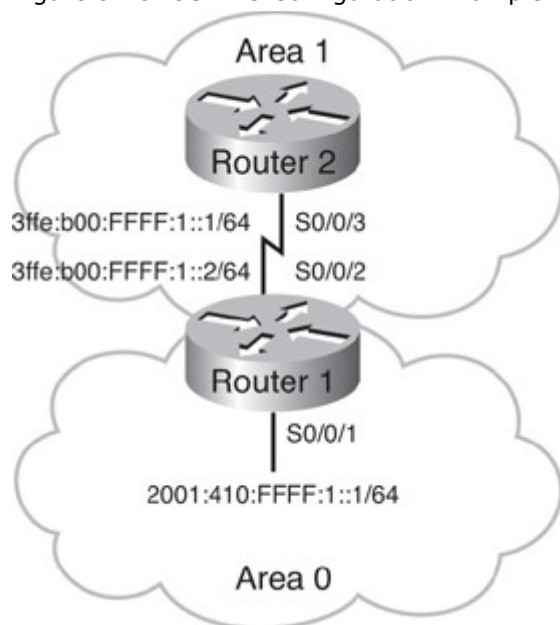
| Parameter   | Description                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------|
| process-id  | Specifies the internal identifier for the OSPF process that is locally assigned and can be any positive integer |
| area-id     | Specifies the area for which information is to be displayed                                                     |
| type number | Specifies the interface type and number for which information is to be displayed                                |
| brief       | Displays brief overview information for all interfaces, states, addresses and masks, and areas on the router    |

The **show ipv6 ospf** [*process-id*] [*area-id*] command displays general information about the IPv6 OSPF processes.

#### OSPFv3 Configuration and Verification Examples

Figure 8-28 shows an OSPFv3 network of two routers and two areas, area 0 and area 1, used for the first OSPFv3 example. The configuration of Router 1 is shown in Example 8-45, and the configuration of Router 2 is shown in Example 8-46. The interface-specific commands `ipv6 ospf 100 area 0` and `ipv6 ospf 100 area 1` create the "ipv6 router ospf 100" process dynamically. The area 0 range `2001:410::/32` command in Router 1 summarizes area 0's routes to the `2001:410::/32` route.

Figure 8-28. OSPFv3 Configuration Example.



Example 8-45. Configuration of Router 1

```

interface Serial0/0/1
 ipv6 address 2001:410:ffff:1::1/64
 ipv6 ospf 100 area 0
!
interface Serial0/0/2
 ipv6 address 3FFE:B00:FFFF:1::2/64
 ipv6 ospf 100 area 1
!
ipv6 router ospf 100
 router-id 10.1.1.3
 area 0 range 2001:410::/32

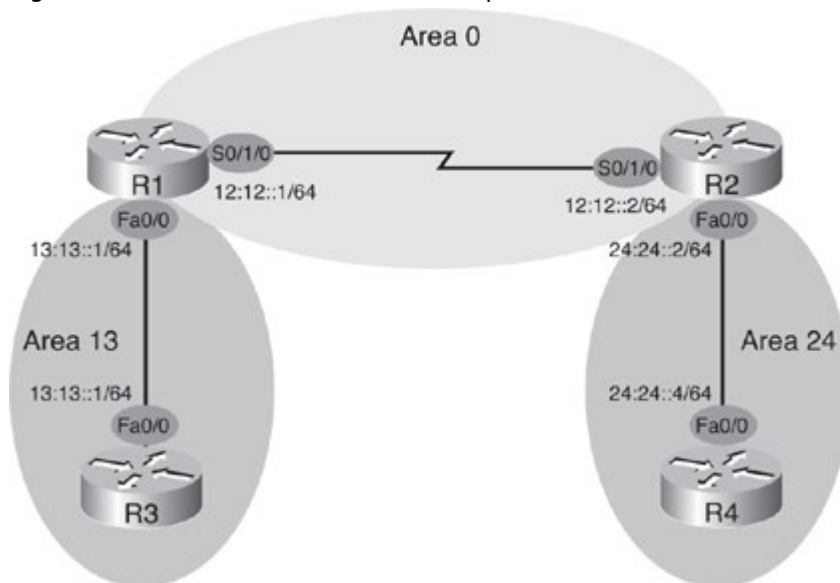
```

Example 8-46. Configuration of Router 2

```
interface Serial0/0/3
 ipv6 address 3FFE:B00:FFFF:1::1/64
 ipv6 ospf 100 area 1
!
ipv6 router ospf 100
 router-id 10.1.1.4
```

Figure 8-29 illustrates a network used in our second OSPFv3 configuration example. This network has three areas.

Figure 8-29. Network for OSPFv3 Example.



The top part of Example 8-47 illustrates the initial OSPFv3 configuration of the R1 router. IPv6 routing is first enabled with the `ipv6 unicast-routing` command, and then the S0/1/0 interface is configured to participate in OSPF process 1 in area 0 with the `ipv6 ospf 1 area 0` command. The error message indicates that the router ID must be configured. In this router, no IPv4 addresses are configured, so the 32-bit router ID is configured with the `router-id` command. Notice that the router ID is set to the router number for simplicity. In the lower part of Example 8-47, R2 is configured similarly. Notice that the two neighbors form a FULL adjacency immediately after they are both configured.

Example 8-47. Initial OSPFv3 Configuration on R1 and R2

```
Code View: Scroll / Show All
R1(config)#ipv6 unicast-routing
R1(config)#interface s0/1/0
R1(config-if)# ipv6 ospf 1 area 0
R1(config-if)#
*Aug 14 06:24:23.040: %OSPFv3-4-NORTRID: OSPFv3 process 1 could not pick a router-
id, please configure manually
R1(config-if)#exit
R1(config)#ipv6 router ospf 1
R1(config-rtr)#router-id 0.0.0.1
R1(config-rtr)#exit
R1(config)#interface s0/1/0
```



```
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#
```

```
R2(config)#ipv6 unicast-routing
R2(config)#ipv6 router ospf 1
R2(config-rtr)#router-id 0.0.0.2
R2(config-rtr)#exit
R2(config)#interface s0/1/0
R2(config-if)#ipv6 ospf 1 area 0
*Aug 14 06:15:14.836: %OSPFv3-5-ADJCHG: Process 1, Nbr 0.0.0.1 on Serial0/1/0 from
LOADING to FULL, Loading Done
R2(config-if)#
```

To verify the configuration, the show ipv6 ospf neighbor and show ipv6 ospf interface commands are used, as shown in Example 8-48. This output is from R2, and indicates that R2 has a FULL relationship with R1. Similar to OSPFv2, the link type is POINT\_TO\_POINT, and the hello and dead intervals are 10 and 40 seconds, respectively. Notice that for OSPFv3, the IPv6 link-local address is used for communication.

Example 8-48. OSPFv3 Verification

Code View: Scroll / Show All

```
R2(config-if)#do show ipv6 ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Interface ID | Interface   |
|-------------|-----|-------|-----------|--------------|-------------|
| 0.0.0.1     | 1   | FULL/ | 00.00.33  | 6            | Serial0/1/0 |

```
R2(config-if)#
```

```
R2(config-if)#do show ipv6 ospf interface
```

Serial0/1/0 is up, line protocol is up

Link Local Address FE80::219:55FF:FE92:B212, Interface ID 6

Area 0, Process ID 1, Instance ID 0, Router ID 0.0.0.2

Network Type POINT\_TO\_POINT, Cost: 64

Transmit Delay is 1 sec, State POINT\_TO\_POINT,

Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5

Hello due in 00:00:09

Index 1/1/1, flood queue length 0

Next 0x0(0)/0x0(0)/0x0(0)

Last flood scan length is 1, maximum is 2

Last flood scan time is 0 msec, maximum is 0 msec

Neighbor Count is 1, Adjacent neighbor count is 1

Adjacent with neighbor 0.0.0.1

Suppress hello for 0 neighbor(s)

```
R2(config-if)#
```

In Example 8-49, area 24 is configured on R2 and R4, and the relationship between the two routers goes to the FULL state. The OSPF portion of R4's routing table is also displayed, and includes the 12:12::/64 interarea route. This is the link between R1 and R2. (The 2001:1::/64 route is a global unicast address configured on R1.)

#### Example 8-49. OSPFv3 Area 24 Configuration and Verification

```
Code View: Scroll / Show All
R2(config)#interface fa0/0
R2(config-if)#ipv6 ospf 1 area 24
R2(config-if)#

R4(config)#ipv6 unicast-routing
R4(config)#ipv6 router ospf 1
R4(config-rtr)#router-id 0.0.0.4
R4(config-rtr)#exit
R4(config)#interface fa0/0
R4(config-if)#ipv6 ospf 1 area 24
*Aug 14 06:34:36.992: %OSPFv3-5-ADJCHG: Process 1, Nbr 0.0.0.2 on FastEthernet0/0
from LOADING to FULL, Loading Done
R4(config-if)#

R4(config-if)#do show ipv6 route ospf
IPv6 Routing Table - 6 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI 12:12::/64 [110/65]
 via FE80::219:55FF:FE92:B212, FastEthernet0/0
OI 2001:1::/64 [110/65]
 via FE80::219:55FF:FE92:B212, FastEthernet0/0
R4(config-if)#
```

In Example 8-50, area 13 is configured on R3 and on R1. The R3 routing table confirms that it has learned all appropriate routes, and a ping test is successfully conducted to verify connectivity from R3 to R4.

#### Example 8-50. OSPFv3 Area 13 Configuration and Verification

```
Code View: Scroll / Show All
R3(config)#ipv6 unicast-routing
R3(config)#ipv6 router ospf 1
R3(config)#
*Aug 14 06:24:09.976: %OSPFv3-4-NORTRID: OSPFv3 process 1 could not pick a
router-id, please configure manually
R3(config-rtr)#router-id 0.0.0.3
R3(config-rtr)#exit
R3(config)#interface fa0/0
R3(config-if)#ipv6 ospf 1 area 13
R3(config-if)#

R1(config-if)#interface fa0/0
```

```

R1(config-if)#ipv6 ospf 1 area 13
R1(config-if)#
*Aug 14 06:40:43.804: %OSPFv3-5-ADJCHG: Process 1, Nbr 0.0.0.3 on FastEthernet0/0
from LOADING to FULL, Loading Done
R1(config-if)#

R3#show ipv6 route ospf
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI 12:12::/64 [110/65]
 via FE80::219:56FF:FE2C:9F60, FastEthernet0/0
OI 24:24::/64 [110/66]
 via FE80::219:56FF:FE2C:9F60, FastEthernet0/0
OI 2001:1::/64 [110/129]
 via FE80::219:56FF:FE2C:9F60, FastEthernet0/0
R3#
R3#ping 24:24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24:24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/16/16 ms
R3#

```

As mentioned, similar to OSPFv2, OSPFv3 can be customized to make the routing more efficient. For example, currently R3 has three routes, and they all go to R1, the ABR for area 13. Area 13 can be configured to be a totally stubby area since the area has only one ABR which it can use to get anywhere outside of the area. The configuration to do this, on R1 and R3, is the same as it is for OSPFv2, and is shown in Example 8-51. R1, the ABR, is configured with the area 13 stub no-summary command, and R3 is configured with the area 13 stub command. Just as in OSPFv2, the no-summary keyword is only required on the ABR. Notice that when R1 is configured, its adjacency with R3 is reset because the two routers do not agree on the stub area status of area 13. As soon as R3 is configured, the adjacency between the two routers becomes FULL again. Example 8-51 also shows the resulting routing table on R3. R3 now only has a default route (::/0), but it can still ping R4 successfully.

Example 8-51. Configuration and Verification of Area 13 as a Totally Stubby Area

```

Code View: Scroll / Show All
R1(config)#ipv6 router ospf 1
R1(config-rtr)#area 13 stub no-summary
R1(config-rtr)#
*Aug 14 06:54:11.780: %OSPFv3-5-ADJCHG: Process 1, Nbr 0.0.0.3 on
FastEthernet0/0 from FULL to DOWN, Neighbor Down: Adjacency forced to reset
R1(config-rtr)#

R3(config)#ipv6 router ospf 1

```

```

R3(config-rtr)#area 13 stub
R3(config-rtr)#
*Aug 14 06:40:17.716: %OSPFv3-5-ADJCHG: Process 1, Nbr 0.0.0.1 on
FastEthernet0/0 from LOADING to FULL, Loading Done
R3(config-rtr)#

R3#show ipv6 route ospf
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI ::/0 [110/2]
 via FE80::219:56FF:FE2C:9F60, FastEthernet0/0
R3#
R3#ping 24:24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24:24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/14/16 ms
R3#

```

## EIGRP for IPv6

EIGRP for IPv6 is available in Cisco IOS Release 12.4(6)T and later. EIGRP for IPv4 and EIGRP for IPv6 are configured and managed separately. However, the configuration and operation of EIGRP for IPv4 and IPv6 is similar.

EIGRP for IPv6 uses the same protocol number (88) as EIGRP for IPv4 does, and it includes all the same features, including keeping neighbor routing table information in a topology table, and using queries if no feasible successors are available.

A router ID is required for EIGRP for IPv6 configuration. Similar to OSPFv3, the router ID is a 32-bit IPv4 address. In an IPv6-only environment, there are no IPv4 addresses assigned, so the router ID must be configured manually.

EIGRP for IPv6 is configured on a per-interface basis, similar to OSPFv3; the **network** command is not used. And, also similar to OSPFv3, link-local addressing is used for establishing neighbor adjacencies. Therefore, for EIGRP for IPv6, it is possible for routers to become neighbors even if they do not have global unicast addresses assigned.

The EIGRP for IPv6 routing protocol has a shutdown feature, and in fact it starts in this state.

EIGRP for IPv4 automatically summarizes on classful network boundaries. This is not the case with EIGRP for IPv6; it does not automatically summarize.

## EIGRP for IPv6 Configuration and Verification Commands

Many of the EIGRP for IPv6 commands are similar to their EIGRP for IPv4 counterparts, with the **ip** keyword replaced by the **ipv6** keyword.

The **ipv6 eigrp as-number** interface configuration command enables EIGRP for IPv6 on an interface. The *as-number* parameter identifies the EIGRP autonomous system (AS) that the interface participates in.

The **ipv6 router eigrp as-number** global configuration command creates an EIGRP for IPv6 routing process and puts the router in router configuration mode.

The **eigrp router-id {ip-address}** router configuration command defines a router ID for EIGRP for IPv6. The *ip-address* is a number in IPv4 address format; in other words it is a 32-bit number in dotted-decimal format. The router ID must be unique on each router.

#### Note

Alternatively, the **router-id** {ip-address} router configuration command may be used on some versions of the IOS. This command was changed in Cisco IOS 12.4(6)T to add support for EIGRP for IPv6. (However, we noted in testing that not all newer versions support this command for EIGRP for IPv6.)

Some documentation for the **router-id** command states that the router ID could alternatively be an IPv6 address, but this is not the case on the routers.

The **no shutdown** router configuration command enables the EIGRP process.

#### Note

The **shutdown** command is not in the EIGRP for IPv6 documentation, but testing confirmed that it is required on the routers. (Testing was conducted on Cisco IOS 12.4(20)T1.)

Similar to EIGRP for IPv4, use the **eigrp stub** [receive-only | connected | static | summary | redistributed] router configuration command to configure a router as an EIGRP stub.

#### Note

Effective with Cisco IOS Release 15.0(1)M and 12.2(33)SRE, the **eigrp stub** command replaced the **stub** command.

The `ipv6 summary-address eigrp as-number ipv6-address [admin-distance]` interface configuration command configures a summary aggregate address for an interface. Table 8-9 describes the parameters of this command.

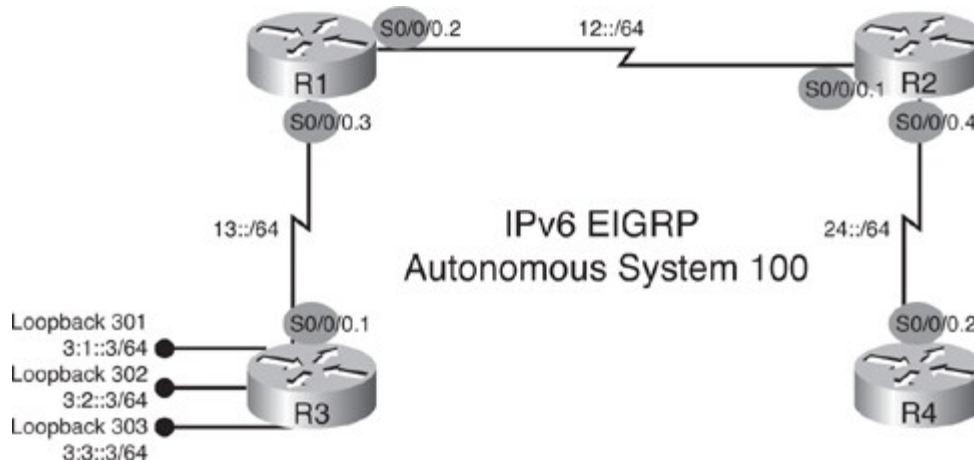
Table 8-9. *ipv6 summary-address eigrp* Command Parameters

| Parameter      | Description                                                                                           |
|----------------|-------------------------------------------------------------------------------------------------------|
| as-number      | Specifies the EIGRP autonomous system number for which routes are to be summarized.                   |
| ipv6-address   | Specifies the IPv6 address of the summary route.                                                      |
| admin-distance | (Optional) Specifies the administrative distance, a value from 0 through 255. The default value is 5. |

#### EIGRP for IPv6 Configuration and Verification Example

Figure 8-30 illustrates the network used to examine EIGRP for IPv6 configuration and verification. All interfaces have IPv6 addresses, including the three loopback interfaces on R3.

Figure 8-30. Network for EIGRP for IPv6 Example.



Example 8-52 shows the configuration of R1 and R3 for EIGRP for IPv6. The `ipv6 unicast-routing` command is configured on each router. On each interface (including the loopback interfaces on R3) the `ipv6 eigrp 100` command is enabled; 100 is the EIGRP autonomous system number. Recall that this must be the same on all routers for them to communicate.

Example 8-52. Initial Configuration of R1 and R3 for EIGRP for IPv6

```
R1(config)#ipv6 unicast-routing
R1(config)#interface Serial0/0/0.2 point-to-point
R1(config-subif)#ipv6 eigrp 100
R1(config-subif)#interface Serial0/0/0.3 point-to-point
R1(config-subif)#ipv6 eigrp 100
R1(config-subif)#exit
R1(config)#

R3(config)#ipv6 unicast-routing
R3(config)#interface Serial0/0/0.1 point-to-point
R3(config-subif)#ipv6 eigrp 100
R3(config-subif)#interface loopback 301
R3(config-if)#ipv6 eigrp 100
R3(config-if)#interface loopback 302
R3(config-if)#ipv6 eigrp 100
R3(config-if)#interface loopback 303
R3(config-if)#ipv6 eigrp 100
R3(config-if)#
```

No messages are displayed on the router's console indicating that EIGRP neighbor relationships have established. The `show ipv6 eigrp neighbor` command output, provided at the top of Example 8-53 indicates why: EIGRP 100 is shut down. For IPv6, the EIGRP process must be enabled, with the `no shutdown` command. This is configured on both R1 and R3, as also shown in the example.

Example 8-53. Enabling the EIGRP for IPv6 Process on R1 and R3

```
R3(config-if)#do show ipv6 eigrp neighbor
IPv6-EIGRP neighbors for process 100
% EIGRP 100 is in SHUTDOWN
R3(config-if)#exit
R3(config)#ipv6 router eigrp 100
R3(config-rtr)#no shutdown

R1(config)#ipv6 router eigrp 100
R1(config-rtr)#no shutdown
R1(config-rtr)#
```

However, an adjacency is still not established. Example 8-54 illustrates the output of the `show ipv6 eigrp neighbor` command on R3. This time the problem is that there is no router ID defined. Example 8-54 also shows the configuration of the router ID on both routers. The neighbor relationship immediately comes up. The EIGRP portion of R1's routing table is shown in the example; R3's loopback interface addresses have been learned. Notice that, unlike EIGRP for IPv4, the networks are not automatically summarized.

Example 8-54. Enabling the EIGRP for IPv6 Router ID on R1 and R3

```
Code View: Scroll / Show All
R3(config)#do show ipv6 eigrp neighbor
IPv6-EIGRP neighbors for process 100
% No router ID for EIGRP 100
R3(config-rtr)#eigrp router-id 3.3.3.3
R3(config-rtr)#
```

```

R1(config-rtr)#eigrp router-id 1.1.1.1
R1(config-rtr)#
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::3 (Serial0/0/0.3) is up: new adjacency
R1(config-rtr)#do show ipv6 route eigrp
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
 D - EIGRP, EX - EIGRP external
D 3:1::/64 [90/2297856]
 via FE80::3, Serial0/0/0.3
D 3:2::/64 [90/2297856]
 via FE80::3, Serial0/0/0.3
D 3:3::/64 [90/2297856]
 via FE80::3, Serial0/0/0.3
R1(config-rtr)#

```

Routers R2 and R4 are configured similarly, as shown in Example 8-55.  
 Example 8-55. Enabling EIGRP for IPv6 on R2 and R4

```

Code View: Scroll / Show All
R2(config)#ipv6 unicast-routing
R2(config)#!
R2(config)#interface Serial0/0/0.1 point-to-point
R2(config-subif)#ipv6 eigrp 100
R2(config-subif)#interface Serial0/0/0.4 point-to-point
R2(config-subif)#ipv6 eigrp 100
R2(config-subif)#ipv6 router eigrp 100
R2(config-rtr)#eigrp router-id 2.2.2.2
R2(config-rtr)#no shutdown
R2(config-rtr)#
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::1 (Serial0/0/0.1) is up: new adjacency
R2(config-rtr)#

R4(config)#ipv6 unicast-routing
R4(config)#interface Serial0/0/0.2 point-to-point
R4(config-subif)#ipv6 eigrp 100
R4(config-subif)#ipv6 router eigrp 100
R4(config-rtr)#eigrp router-id 4.4.4.4
R4(config-rtr)#no shutdown
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::2 (Serial0/0/0.2) is up: new

```



```
adjacency
R4(config-rtr)#
```

R4's routing table contains the appropriate routes, including to R3's loopback addresses, as shown Example 8-56. The successful ping from R4 to one of these loopbacks is also illustrated.

Example 8-56. Verifying R4's Routing Table

```
Code View: Scroll / Show All
R4(config-rtr)#do show ipv6 route eigrp
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
 D - EIGRP, EX - EIGRP external
D 3:1::/64 [90/3321856]
 via FE80::2, Serial0/0/0.2
D 3:2::/64 [90/3321856]
 via FE80::2, Serial0/0/0.2
D 3:3::/64 [90/3321856]
 via FE80::2, Serial0/0/0.2
D 12::/64 [90/2681856]
 via FE80::2, Serial0/0/0.2
D 13::/64 [90/3193856]
 via FE80::2, Serial0/0/0.2
R4(config-rtr)#do ping 3:1::3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 3:1::3, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/88/88 ms
R4(config-rtr)#
```

As mentioned, EIGRP for IPv6 supports many other features. To illustrate the stub routing feature, EIGRP event debugging is enabled on R4 with the `debug ipv6 eigrp` command, and then the first loopback interface on R3 is shut down, as shown in Example 8-57. The resulting debug output is shown. Notice that a query packet about the lost route goes from R3 all the way to R4.

Example 8-57. EIGRP for IPv6 Behavior

```
Code View: Scroll / Show All
R4(config-rtr)#do debug ipv6 eigrp
IP-EIGRP Route Events debugging is on
R4(config-rtr)#

R3(config-if)#interface loopback 301
R3(config-if)#shutdown
```

```
R3(config-if)#
```

```
R4(config-rtr)#
```

```
IPv6-EIGRP(0:100): Processing incoming QUERY packet
IPv6-EIGRP(0:100): Int 3:1::/64 M 4294967295 - 0 4294967295 SM 4294967295 -
0
4294967295
IPv6-EIGRP(0:100): 3:1::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): 3:1::/64 (90/-1) added to RIB
IPv6-EIGRP(0:100): 3:1::/64 - do advertise out Serial0/0/0.2
IPv6-EIGRP(0:100): Int 3:1::/64 metric 4294967295 - 0 4294967295
IPv6-EIGRP(0:100): 3:1::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): 3:1::/64 - not in IPv6 routing table
IPv6-EIGRP(0:100): Int 3:1::/64 metric 4294967295 - 0 4294967295
IPv6-EIGRP(0:100): Processing incoming UPDATE packet
IPv6-EIGRP(0:100): Int 3:1::/64 M 4294967295 - 0 4294967295 SM 4294967295 -
0
4294967295
```

Recall from Chapter 2 that with the EIGRP stub routing feature, a stub router indicates in its hello packet to all neighboring routers its status as a stub router. Any router that receives a packet informing it of its neighbor's stub status does not query the stub router for any routes. Therefore, a router that has a stub peer does not query that peer. Instead the router connected to the stub router answers the query on behalf of the stub router. Because R4 is a stub router, it should be configured as such, to stop unnecessary queries going to it. Example 8-58 illustrates the configuration of R4 as a stub, and the resulting debug output. (The stub command is used on R4 in this example; recall that the newer version of this command, `eigrp stub`, could have been used.) As soon as the stub command is entered, the EIGRP peer is reset and the router processes all the EIGRP updates again.

Example 8-58. Configuring a Router as an EIGRP Stub

Code View: Scroll / Show All

```
R4(config-rtr)#stub
```

```
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::2 (Serial0/0/0.2) is down: peer
info changed
```

```
R4(config-rtr)#
```

```
IPv6-EIGRP(0:100): 3:3::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): 3:2::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): 12::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): 13::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::2 (Serial0/0/0.2) is up: new
adjacency
IPv6-EIGRP(0:100): Processing incoming UPDATE packet
IPv6-EIGRP(0:100): Processing incoming UPDATE packet
IPv6-EIGRP(0:100): Int 12::/64 M 2681856 - 1657856 1024000 SM 2169856 - 1657856
512000
IPv6-EIGRP(0:100): 12::/64 (90/2681856) added to RIB
IPv6-EIGRP(0:100): Int 24::/64 M 2681856 - 1657856 1024000 SM 2169856 - 1657856
512000
IPv6-EIGRP(0:100): 24::/64 routing table not updated
```

```
IPv6-EIGRP(0:100): Int 13::/64 M 3193856 – 1657856 1536000 SM 2681856 – 1657856
1024000
IPv6-EIGRP(0:100): 13::/64 (90/3193856) added to RIB
IPv6-EIGRP(0:100): Int 3:2::/64 M 3321856 – 1657856 1664000 SM 2809856 – 1657856
1152000
IPv6-EIGRP(0:100): 3:2::/64 (90/3321856) added to RIB
IPv6-EIGRP(0:100): Int 3:3::/64 M 3321856 – 1657856 1664000 SM 2809856 – 1657856
1152000
IPv6-EIGRP(0:100): 3:3::/64 (90/3321856) added to RIB
IPv6-EIGRP(0:100): 24::/64 – do advertise out Serial0/0/0.2
IPv6-EIGRP(0:100): Int 24::/64 metric 2169856 – 1657856 512000
R4(config-rtr)#
```

To confirm that the stub configuration is working, the R3 loopback interface is enabled and then disabled again. This process and the resulting debugoutput on R4 are illustrated in Example 8-59. Notice that there are no queries; only an update and a reply message are received, telling R4 about the deleted route.

Example 8-59. Verifying the EIGRP Stub Configuration

```
Code View: Scroll / Show All
R3(config-if)#no shutdown
R3(config-if)#shutdown

R4(config-rtr)#
IPv6-EIGRP(0:100): Processing incoming UPDATE packet
IPv6-EIGRP(0:100): Int 3:1::/64 M 4294967295 – 0 4294967295 SM 4294967295 -
0
4294967295
IPv6-EIGRP(0:100): Int 3:1::/64 metric 4294967295 – 0 4294967295
IPv6-EIGRP(0:100): Processing incoming REPLY packet
IPv6-EIGRP(0:100): Int 3:1::/64 M 4294967295 – 0 4294967295 SM 4294967295 -
0
4294967295
IPv6-EIGRP(0:100): 3:1::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
R4(config-rtr)#
```

Another feature supported by EIGRP for IPv6 is route summarization. As we have seen, this protocol does not automatically summarize, but properly configured summarization is useful for fault isolation and performance optimization, and results in a more stable network. In the example, summarization of the three loopback interface networks on R3 is configured with the `ipv6 summary-address eigrp 100 3::/16` command. The configuration is shown in Example 8-60. Notice that the neighbor adjacency is reset immediately. The resulting debug output on R4 and the R4 routing table is also shown. As expected, the routing table contains the summary of the three loopback addresses, not the addresses themselves.

Example 8-60. Configuring and Verifying EIGRP Summarization Configuration

```
Code View: Scroll / Show All
R3(config-if)#interface serial 0/0/0.1
R3(config-subif)#ipv6 summary-address eigrp 100 3::/16
```

```

R3(config-subif)#
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::1 (Serial0/0/0.1) is down:
summary configured
%DUAL-5-NBRCHANGE: IPv6-EIGRP(0) 100: Neighbor FE80::1 (Serial0/0/0.1) is up: new
adjacency

R4(config-rtr)#
IPv6-EIGRP(0:100): Int 3:3::/64 metric 4294967295 -1657856 4294967295
IPv6-EIGRP(0:100): Int 3:2::/64 metric 4294967295 -1657856 4294967295
IPv6-EIGRP(0:100): Processing incoming REPLY packet
IPv6-EIGRP(0:100): Int 3:3::/64 M 4294967295 - 1657856 4294967295 SM 4294967295
-
1657856 4294967295
IPv6-EIGRP(0:100): 3:3::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): Int 3:2::/64 M 4294967295 - 1657856 4294967295 SM 4294967295
-
1657856 4294967295
IPv6-EIGRP(0:100): 3:2::/64 deleted FE80::2(FE80::2)/Serial0/0/0.2
IPv6-EIGRP(0:100): Processing incoming UPDATE packet
IPv6-EIGRP(0:100): Int 3::/16 M 3321856 - 1657856 1664000 SM 2809856 - 1657856
1152000
IPv6-EIGRP(0:100): 3::/16 (90/3321856) added to RIB
R4(config-rtr)#
R4(config-rtr)#do show ipv6 route eigrp
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
 D - EIGRP, EX - EIGRP external
D 3::/16 [90/3321856]
 via FE80::2, Serial0/0/0.2
D 12::/64 [90/2681856]
 via FE80::2, Serial0/00.2
D 13::/64 [90/3193856]
 via FE80::2, Serial0/0/0.2
R4(config-rtr)#

```

If you wanted to illustrate how summarization helps with fault isolation, one of the loopback interfaces on R3 could be shutdown, and then enabled. No messages would appear on R4, illustrating that by summarizing, the scope of the failure domain would be reduced, and the routing overhead and routing table size would be decreased.

MBGP

To make Border Gateway Protocol Version 4 (BGP-4) available for other network layer protocols, including Multiprotocol Label Switching (MPLS) virtual private network (VPN) and IPv6, RFC 4760 defines multiprotocol extensions for BGP-4. RFC 2545 defines how these extensions are used for IPv6.

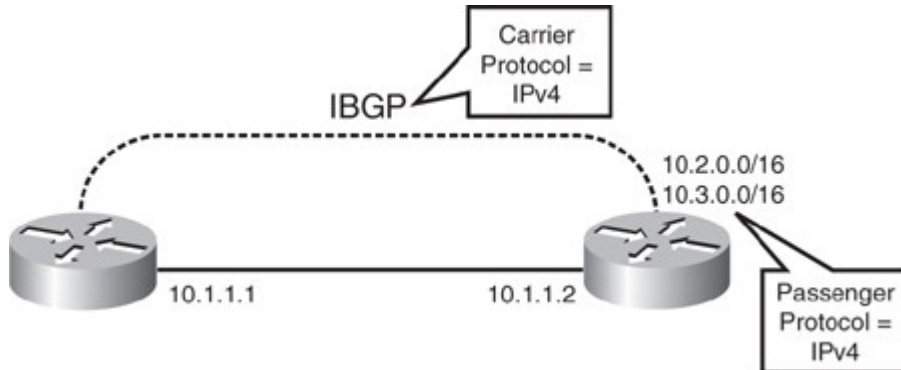
IPv6-specific extensions incorporated into MBGP include the following:

- A new identifier for the IPv6 address family.
- Scoped addresses—The NEXT\_HOP attribute contains a global IPv6 address and potentially a link-local address (only when there is link-local reachability with the peer).
- The NEXT\_HOP and Network Layer Reachability Information (NLRI) attributes are expressed as IPv6 addresses and prefixes. (The NLRI field in a BGP update message lists the networks reachable on the BGP path described by the update message.)

MBGP can of course operate with multiple protocols. It operates by identifying two separate protocols: the carrier protocol and the passenger protocol.

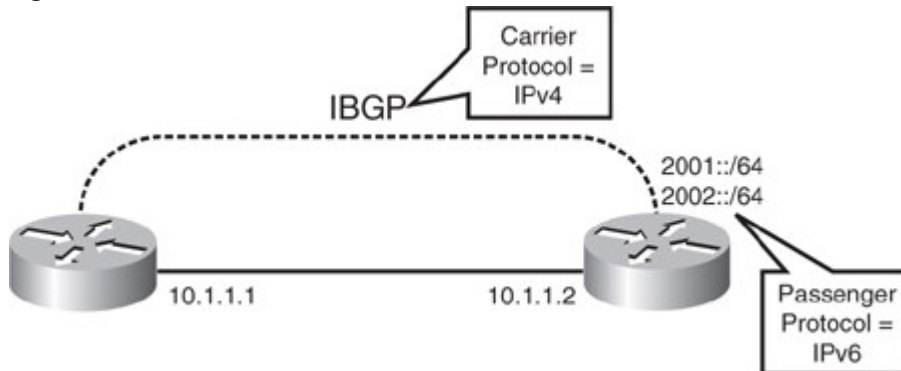
In an all-IPv4 environment, BGP establishes sessions using IPv4 (using TCP port 179); IPv4 is the carrier protocol. The routes that BGP advertises are also IPv4. Therefore, IPv4 is also the passenger protocol. This scenario is illustrated in Figure 8-31.

Figure 8-31. BGP with IPv4 as Both the Carrier and Passenger Protocol.



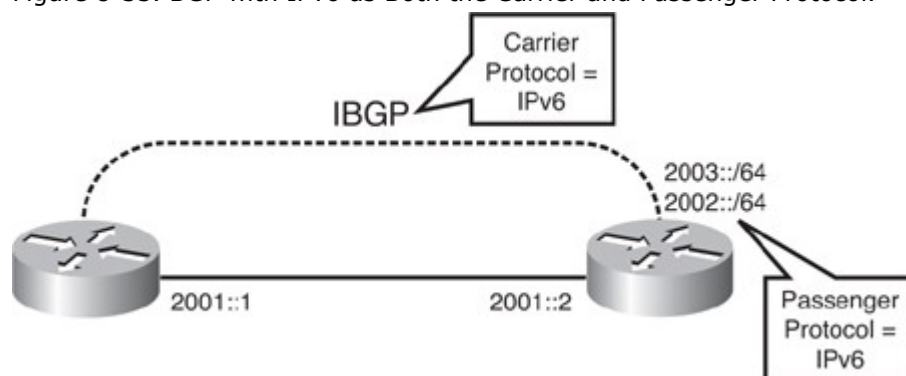
Protocols other than IPv4, including MPLS VPN, multicast, and IPv6, also need to advertise reachability information. MBGP extensions to allow these other protocols to be carried using BGP. An analogy is that BGP is a truck that can transport multiple payloads. For example, the BGP “truck” could be IPv4 and it could be used to transport IPv6 (or other protocol) “payloads.” In this case, the carrier protocol is IPv4 and the passenger protocol is IPv6, the IPv6 prefixes being advertised. This scenario is illustrated in Figure 8-32.

Figure 8-32. BGP with IPv4 as the Carrier Protocol and IPv6 as the Passenger Protocol.



In an all IPv6 environment, BGP can be used as both the carrier and passenger protocol, as illustrated in Figure 8-33. In this case, IPv6 is used to establish BGP sessions, and BGP advertises IPv6 prefixes; both the truck and the payload are IPv6.

Figure 8-33. BGP with IPv6 as Both the Carrier and Passenger Protocol.



IPv6 routes are carried with the use of the MBGP extension IPv6 NLRI, which defines a new *address family* that can be advertised using MBGP. Each protocol that uses MBGP has an address family defined.

#### Note

Either IPv4 or IPv6 can be used as the carrier protocol for other passenger protocols, such as multicast or MPLS VPNs.

#### MBGP Configuration and Verification Commands

MBGP for IPv6 is configured similarly to configuring BGP for IPv4, as explored in Chapter 6, “Implementing a Border Gateway Protocol Solution for ISP Connectivity.”

The same **router bgp autonomous-system** global configuration command is used to enter BGP configuration mode, and identify the local autonomous system in which this router belongs. The *autonomous-system* parameter identifies the local autonomous system.

The **bgp router-id ip-address** router configuration command configures a fixed router ID for the local BGP routing process. The *ip-address* parameter is an IPv4-address format 32-bit dotted-decimal number. This command is required in an IPv6-only network.

The neighbor {ipv6-address | peer-group-name} remote-as autonomous-system-number router configuration command is similar to its IPv4 counterpart and is used to activate a BGP session for external and internal neighbors and to identify a peer router with which the local router will establish a session. The router runs internal BGP (IBGP) with internal neighbors and external BGP (EBGP) with external neighbors. The parameters in this command are described in Table 8-10.

Table 8-10. *neighbor remote-as* Command Parameters

| Parameter         | Description                                    |
|-------------------|------------------------------------------------|
| ipv6-address      | Identifies the peer router                     |
| peer-group-name   | Identifies the name of a BGP peer group        |
| autonomous-system | Identifies the peer router’s autonomous system |

The address-family ipv6 [unicast | multicast | vpnv6] router configuration command enters address family configuration mode for configuring BGP routing sessions that use IPv6 address prefixes. The parameters in this command are described in Table 8-11.

Table 8-11. *address-family ipv6* Command Parameters

| Parameter | Description                                          |
|-----------|------------------------------------------------------|
| unicast   | (Optional) Specifies IPv6 unicast address prefixes   |
| multicast | (Optional) Specifies IPv6 multicast address prefixes |
| vpnv6     | (Optional) Specifies VPN version 6 address prefixes  |

The **neighbor ipv6-address activate** address family configuration (or router configuration) command enables the exchange of information with a BGP neighbor. The *ipv6-address* is the IPv6 address of the neighbor.

The same **network network-number** address family configuration (or router configuration) command is used to specify the networks to be advertised by the BGP and MBGP routing process in the *network-number* parameter.

To apply a route map to filter incoming or outgoing MBGP routes, use the similar **neighbor ipv6-address route-map name {in | out}** address family configuration (or router configuration) command. The routes that are permitted may have their attributes set or changed, using **set** commands in the route map. This is useful when trying to influence route selection.

#### MBGP Configuration and Verification Example

Example 8-61 illustrates an example configuration of a router with one MBGP neighbor. The first section of the router bgp configuration defines a 32-bit router ID (which must be configured in an IPv6-only network) and establishes the BGP peering session with the neighbor to send and receive advertisements. In this example, the peering session is established using IPv6, so IPv6 is the carrier protocol. The second section of the configuration starts with the address-family ipv6 command; it defines the passenger protocol, the protocol that is to be advertised using MBGP. In this example, the passenger protocol is IPv6. All passenger protocols have an address family section within the router bgp configuration. The address family section is where routing policies and specific features for the particular address family are defined. In this example, the neighbor is activated (as it must be for each address family) and redistribution is configured.

Example 8-61. Configuration of a Router with One MBGP Neighbor

```
router bgp 100
 bgp router-id 1.1.1.1
 neighbor 2001:100:2:4::1 remote-as 100
!
 address-family ipv6
 neighbor 2001:100:2:4::1 activate
 redistribute connected
```

#### Note

Further BGP configuration and verification examples are provided in the “RIPng, OSPFv3, and MBGP Redistribution Configuration and Verification Example” section, later in this chapter.

#### IPv6 Policy-Based Routing

As described in Chapter 4, “Manipulating Routing Updates,” policy-based routing (PBR) allows the network administrator to define a routing policy other than destination-based routing using the routing table. PBR is sometimes called traffic engineering and helps to provide a high degree of control over routing. PBR is available for both IPv4 and IPv6.

PBR can be used for manually configuring the path that packets will take. It can also be used to classify and mark packets. As is the case for IPv4, IPv6 PBR is based on **match** commands for identifying the traffic to be policy-based routed, and **set** commands for defining how that traffic will be routed.

#### IPv6 PBR Configuration and Verification Commands

Many of the commands used for IPv6 PBR are the same as or similar to their IPv4 counterparts, some with the **ip** keyword replaced by the **ipv6** keyword.

The route-map map-tag [permit | deny] [sequence-number] global configuration command is the same as the IPv4 command explained in Chapter 4. It creates a route map. The parameters of this command are explained again in detail in Table 8-12.

Table 8-12. *route-map* Command Parameters

| Parameter            | Description                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| map-tag              | Name of the route map.                                                                                              |
| <b>permit   deny</b> | (Optional) A parameter that specifies the action to be taken if the route map match conditions are met. The meaning |



| Parameter       | Description                                                                                                                                                                  |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                 | of <b>permit</b> or <b>deny</b> is dependent on how the route map is used. For PBR, <b>permit</b> means that the packet is policy routed.                                    |
| sequence-number | (Optional) A sequence number that indicates the position that a new route map statement will have in the list of route map statements already configured with the same name. |

The default for the **route-map** command is **permit**, with a *sequence-number* of 10.

Recall that if you leave out the sequence number when configuring all statements for the same route map name, the router will assume that you are editing (and possibly adding to) the first statement, which defaults to sequence number 10. Route map sequence numbers do not automatically increment!

A route map can refer to many **match** and **set** commands. This section provides only one of each, and in the next section provides an example that uses these two commands.

The **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} route-map configuration command specifies a prefix permitted by a prefix list to use in redistribution or an IPv6 access list to use to match packets for PBR for IPv6. The *prefix-list-name* specifies the name of the prefix list, and the *access-list-name* parameter specifies the name of the access list.

The **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*] route-map configuration command specifies where to forward IPv6 packets that pass a match clause of a route map for PBR. The *global-ipv6-address* is the IPv6 global address of the next hop to which packets are output. The next-hop router must be an adjacent router. Note that a global IPv6 address must be specified; an IPv6 link-local address cannot be used because an IPv6 link-local address requires the interface to be specified to determine which interface to use.

The **ipv6 policy route-map** *route-map-name* interface configuration command configures IPv6 PBR on an interface. The *route-map-name* parameter is the name of the route map to use for PBR and must match a *map-tag* value specified in a **route-map** command.

Packets originating on the router are not normally policy routed. *Local policy routing* enables packets originating on the router to take a route other than the obvious shortest path. To identify a route map to use for local policy routing, use the **ipv6 local policy route-map** *route-map-name* global configuration command. The *route-map-name* parameter is the name of the route map to use for PBR and must match a *map-tag* value specified in a route-map command. This command applies the specified route map to packets originating on the router.

The **ipv6 access-list** *access-list-name* global configuration command defines an IPv6 access list (ACL) and places the router in IPv6 access list configuration mode. The *access-list-name* parameter specifies the name of the access list. Note that IPv6 does not support numbered ACLs, and an IPv4 ACL and an IPv6 ACL cannot share the same name.

Within access list configuration mode, permit and deny statements create the access list. In IPv6, as in IPv4, many parameters are available for both the permit and deny statements. In this section we explore only a few of the parameters of the permit statement: the permit protocol {source-ipv6-prefix/prefix-length | any | host source-ipv6-address} {destination-ipv6-prefix/prefix-length | any | host destination-ipv6-address} IPv6 access list configuration command defines permit conditions for an IPv6 ACL. These parameters are explained in detail in Table 8-13.

Table 8-13. *permit* Command Parameters

| Parameter                              | Description                                                                                  |
|----------------------------------------|----------------------------------------------------------------------------------------------|
| protocol                               | Specifies the name or number of an IPv6 protocol. <b>ipv6</b> indicates any protocol.        |
| source-ipv6-prefix/prefix-length       | Specifies the source IPv6 network or class of networks about which to set permit conditions. |
| any                                    | An abbreviation for the IPv6 unspecified prefix ::/0.                                        |
| <b>host</b> <i>source-ipv6-address</i> | Specifies the source IPv6 host address about which to set permit conditions.                 |

| Parameter                                          | Description                                                                                       |
|----------------------------------------------------|---------------------------------------------------------------------------------------------------|
| <code>destination-ipv6-prefix/prefix-length</code> | Specifies the destination IPv6 network or class of networks about which to set permit conditions. |
| <b>host</b> <i>destination-ipv6-address</i>        | Specifies the destination IPv6 host address about which to set permit conditions.                 |

#### Note

As mentioned, many more parameters are available for the **permit** command. See the Cisco IOS IPv6 Command Reference Manual for details.

The **ping** command for IPv6 has been used many times in this chapter. As it is for IPv4, the default behavior is to use the address of the exiting interface as the source address of the packets. This can be modified by specifying the source interface to use in the **ping ipv6 ipv6-address source interface-name** command.

The **debug ipv6 policy** [*access-list-name*] displays IPv6 policy routing packet activity. The *access-list-name* parameter specifies the name of the access list.

The **show ipv6 access-list** [*access-list-name*] displays the contents of all or a specified IPv6 ACL.

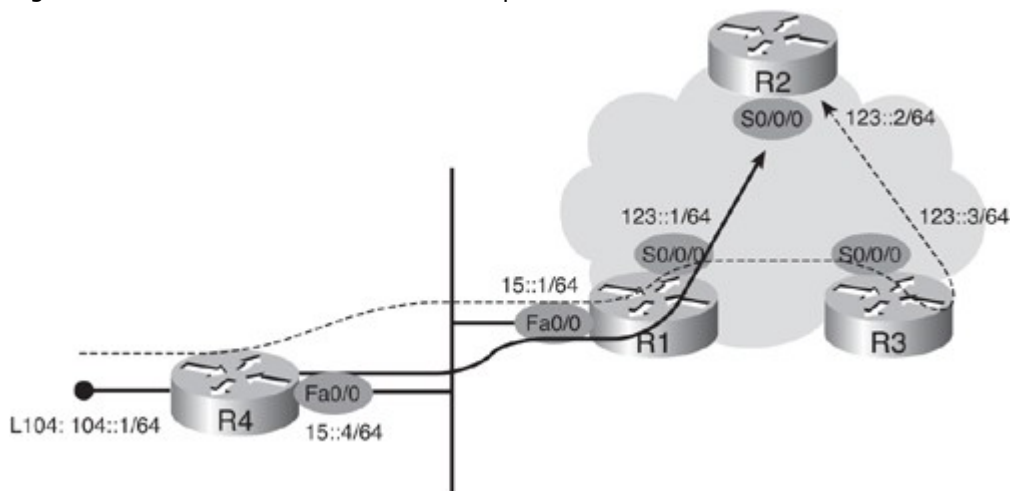
The *access-list-name* parameter specifies the name of the access list.

To display configured route maps, including IPv6 route maps, use the **show route-map** [*map-name*] EXEC command, where *map-name* is an optional name of a specific route map.

#### IPv6 PBR Configuration and Verification Example

Figure 8-34 illustrates the network used to illustrate PBR configuration and verification. All interfaces have IPv6 addresses, and RIPng is configured on all routers for full reachability. In this network, traffic from R4 to R2 normally takes the R4-R1-R2 path, which is the solid line in the figure. The objective of this example is to configure PBR such that traffic sourced from the loopback 104 interface on R4 takes the R4-R1-R3-R2 path, which is the dashed line in the figure.

Figure 8-34. Network for IPv6 PBR Example.



Recall that PBR is configured on the interface that receives the traffic that should be routed differently than the default path. In this example, packets received on the Fast Ethernet 0/0 interface on R1 will be examined. If they are from R4's loopback interface, they will be routed via R3; otherwise they will be routed normally, through R2. Example 8-62 shows the configuration of the route map **PBR\_SOURCE\_ADDRESS** on R1. Sequence 10 of the route map is a permit entry. It matches the IPv6 addresses permitted by the access list called **SOURCE\_104**. If the traffic matches, it is policy-routed and processed according to the **set** command, which specifies that the next-hop address is 123::3, Router R3's address. The **SOURCE\_104** ACL permits packets with a source address in the IPv6 subnet of R4's loopback address, 104::/64, and with any destination address. The route map is applied to the Fast Ethernet interface. Therefore, a packet received on the Fast Ethernet 0/0 interface, from a source address in the 104 network, will match this route map condition and its next-hop address will be set to 123::3.

#### Example 8-62. Configuring of Route Map on R1

```
R1(config)#route-map PBR_SOURCE_ADDRESS permit 10
R1(config-route-map)#match ipv6 address SOURCE_104
R1(config-route-map)#set ipv6 next-hop 123::3
R1(config-route-map)#exit
R1(config)#ipv6 access-list SOURCE_104
R1(config-ipv6-acl)#permit ipv6 104::/64 any
R1(config-ipv6-acl)#exit
R1(config)#interface fa0/0
R1(config-if)#ipv6 policy route-map PBR_SOURCE_ADDRESS
R1(config-if)#
```

As shown in Example 8-63, the debug of IPv6 policy-based routing is enabled using the debug ipv6 policy command on R1, to verify that the route map is working. On R4, the R2 router address is pinged, first using the default exit interface (R4's Fast Ethernet 0/0). In this case, there is no debug output on R1. Notice the round-trip delay of 56 milliseconds (ms). R2 is again pinged from R4, this time using the loopback 104 interface as the source. Notice the round-trip delay has increased (to 88 ms), indicating that the traffic is taking a longer path. The resulting debug output on R1 indicates that these packets were policy routed; traffic that matches the source address in the ACL is being sent to R3 as the next hop.

#### Example 8-63. Verification of Route Map on R1

Code View: Scroll / Show All

```
R1#debug ipv6 policy
```

```
R1#
```

```
R4#ping 123::2
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 123::2, timeout is 2 seconds:

```
!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms

```
R4#
```

```
R1#
```

```
R1#
```

```
R4#ping 123::2 source loopback 104
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 123::2, timeout is 2 seconds:

Packet sent with a source address of 104::1

```
!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 88/88/88 ms

```
R4#
```

```
R1#
```

```
*Aug 14 10:03:58.955: IPv6 PBR: FastEthernet0/0, matched src 104::1 dst 123::2
protocol 58
```

```

*Aug 14 10:03:58.955: IPv6 PBR: set nexthop 123::3, interface Serial0/0/0
*Aug 14 10:03:58.955: IPv6 PBR: policy route via Serial0/0/0/123::3
*Aug 14 10:03:59.043: IPv6 PBR: FastEthernet0/0, matched src 104::1 dst 123::2
protocol 58
*Aug 14 10:03:59.043: IPv6 PBR: set nexthop 123::3, interface Serial0/0/0
*Aug 14 10:03:59.043: IPv6 PBR: policy route via Serial0/0/0/123::3
*Aug 14 10:03:59.131: IPv6 PBR: FastEthernet0/0, matched src 104::1 dst 123::2
protocol 58
*Aug 14 10:03:59.131: IPv6 PBR: set nexthop 123::3, interface Serial0/0/0
*Aug 14 10:03:59.131: IPv6 PBR: policy route via Serial0/0/0/123::3
*Aug 14 10:03:59.219: IPv6 PBR: FastEthernet0/0, matched src 104::1 dst 123::2
protocol 58
*Aug 14 10:03:59.219: IPv6 PBR: set nexthop 123::3, interface Serial0/0/0
*Aug 14 10:03:59.219: IPv6 PBR: policy route via Serial0/0/0/123::3
*Aug 14 10:03:59.303: IPv6 PBR: FastEthernet0/0, matched src 104::1 dst 123::2
protocol 58
*Aug 14 10:03:59.307: IPv6 PBR: set nexthop 123::3, interface Serial0/0/0

```

Example 8-64 illustrates the show route-map and show ipv6 access-list command output on R1. In the first command output, notice that five packets (which is 500 bytes in this case) have been policy routed through the route map PBR\_SOURCE\_ADDRESS. The second command output verifies that these five packets are the five matches to the ACL SOURCE\_104.

Example 8-64. show Command Output for Route Map on R1

```

R1#show route-map
route-map PBR_SOURCE_ADDRESS, permit, sequence 10
 Match clauses:
 ipv6 address SOURCE_104
 Set clauses:
 ipv6 next-hop 123::3
 Policy routing matches: 5 packets, 500 bytes
R1#
R1#show ipv6 access-list
IPv6 access list SOURCE_104
 permit ipv6 104::/64 any (5 matches) sequence 10
R1#

```

## IPv6 Redistribution

Similar to IPv4, IPv6 routing information can be redistributed between routing protocols, and between instances of routing protocols. This section introduces and shows examples of redistribution.

## RIPng Redistribution

The “RIPng” section, earlier in this chapter, showed how to configure and verify a single instance of RIPng. With RIPng, multiple instances can be running on the same router, and on the same interface. By default, these instances use the same multicast group and the same port number and accept updates from each other. However, if the port number or the multicast group address are changed, the instances will not communicate by default; redistribution can be configured so that the instances share their routes. The seed metric used in redistributed routes defaults to one hop, and can be changed using route maps.

## RIPng Redistribution Configuration and Verification Commands

RIPng redistribution configuration and verification uses many of the commands explored so far in this chapter. Other commands used in the upcoming examples are noted in this section.

The **port port-number multicast-group multicast-address** router configuration command configures the specified UDP port and multicast group address for RIPng process.

There are many options for the **redistribute** router configuration command. The options explored in this section are the **redistribute connected** command, which redistributes connected routes into the routing process, and the **redistribute rip process-id**, which redistributes the specified RIPng process into the routing process.

Note

As for IPv4, routes that are included in the address space advertised by a **network** command under a routing process will be advertised by that process, and a **redistribute connected** command would not be required.

The **set metric metric-value** route-map configuration command specifies the routing protocol metric. For RIPng, the metric is hops.

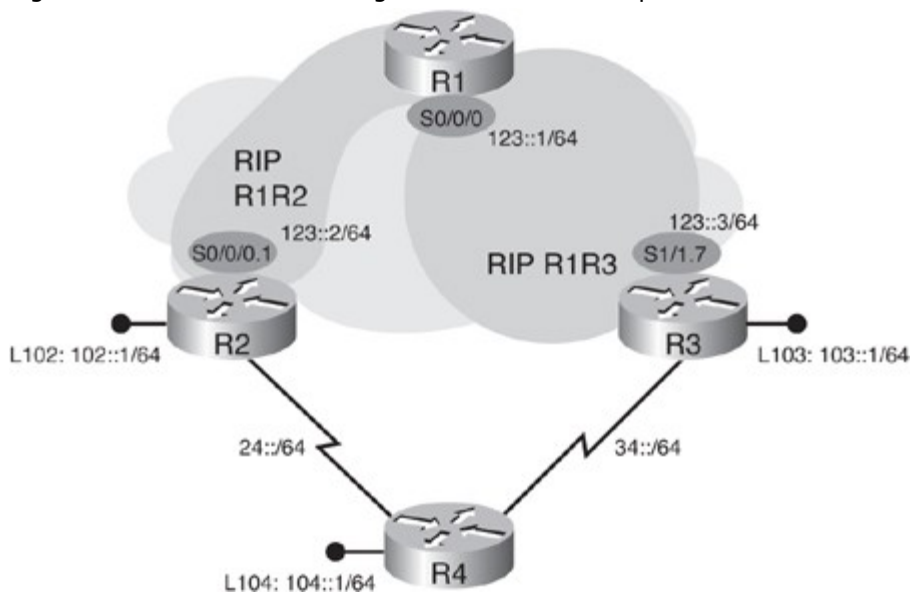
The **clear ipv6 route {ipv6-address | ipv6-prefix/prefix-length | \*}** privileged EXEC command deletes routes from the IPv6 routing table. The *ipv6-address* parameter causes only routes to the specified address to be cleared. The *ipv6-prefix/prefix-length* parameter causes only routes to the specified prefix to be cleared. The *\** parameter causes all routes to be cleared.

Similar to IPv4 prefix lists, IPv6 prefix lists have many parameters. In this section we explore only one option: the **ipv6 prefix-list list-name [seq seq-number] permit ipv6-prefix/prefix-length** global configuration command creates an IPv6 prefix list entry with the name *list-name* that permits networks that match the prefix specified by *ipv6-prefix/prefix-length*. The *seq-number* specifies the sequence number of a prefix list entry, which determines the order of the entries in the list.

RIPng Redistribution Configuration and Verification Example

Figure 8-35 illustrates a network used for a RIPng redistribution example. Routers R1, R2, and R3 will all be configured for RIPng, but in two different RIPng processes, R1R2 and R1R3, as shown in the figure. The networks on the R2 and R3 loopback and lower serial interfaces will be redistributed into the RIPng processes. (Router R4 is not used in this example, but the same diagram is used in later examples in which R4 is used.)

Figure 8-35. Network for RIPng Redistribution Example.



Before RIPng is configured, Example 8-65 illustrates the results of pings from R1 to R2, and from R1 to R3, confirming that there is communication over the 123::/64 network.

Example 8-65. Verification of Connectivity Between Routers

```
R1#ping 123::2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 123::2, timeout is 2 seconds:
```

```
!!!!
```

```

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
R1#
R1#
R1#ping 123::3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 123::3, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
R1#

```

Example 8-66 shows the configuration of the RIPng process on R1's Serial 0/0/0 interface and on R2's Serial 0/0/0.1 interface. Recall that configuring RIPng on the interface automatically creates the RIPng process if it were not already created. Notice that the configured processes have different names. This example also shows the results of the debug ipv6 rip command on R2. Because the processes are different, we may expect that R2 would only send, and not receive, updates. However, the debug output shows both sending and receiving happening. Notice that both processes by default use the FF02::9 multicast group address (for all RIPng routers) and use UDP port 521.

Example 8-66. RIPng Configuration and Verification on R1 and R2

```

Code View: Scroll / Show All
R1(config)#ipv6 unicast-routing
R1(config)#interface s0/0/0
R1(config-if)#ipv6 rip R1R3 enable
R1(config-if)#

R2(config)#ipv6 unicast-routing
R2(config)#interface s0/0/0.1
R2(config-subif)#ipv6 rip R1R2 enable
R2(config-subif)#do debug ipv6 rip
RIP Routing Protocol debugging is on
R2(config-subif)#
*Aug 14 11:43:03.970: RIPng: Sending multicast update on Serial0/0/0.1 for R1R2
*Aug 14 11:43:03.970: src=FE80::2
*Aug 14 11:43:03.970: dst=FF02::9 (Serial0/0/0.1)
*Aug 14 11:43:03.970: sport=521, dport=521, length=32
*Aug 14 11:43:03.970: command=2, version=1, mbz=0, #rte=1
*Aug 14 11:43:03.970: tag=0, metric=1, prefix=123::/64
*Aug 14 11:43:05.498: RIPng: response received from FE80::1 on Serial0/0/0.1 for
R1R2
*Aug 14 11:43:05.498: src=FE80::1 (Serial 0/0/0.1)
*Aug 14 11:43:05.498: dst=FF02::9
*Aug 14 11:43:05.498: sport=521, dport=521, length=32
*Aug 14 11:43:05.498: command=2, version=1, mbz=0, #rte=1
*Aug 14 11:43:05.498: tag=0, metric=1, prefix=123::/64

```

On R2, the redistribute connected command is configured under the R1R2 process, as shown in Example 8-67, so that R2 will advertise its connected routes to R1. As the debug output illustrates, the update is using the default FF02::9 multicast group address (for all RIPng routers) and UDP port 521. The connected routes

are being redistributed with a default metric of 1. The example also shows the RIPng routes in R1's routing table; it contains routes to R2's loopback and serial interface networks.

#### Example 8-67. Redistribute Connected Configuration and Verification

Code View: Scroll / Show All

```
R2(config)#ipv6 router rip R1R2
```

```
R2(config-rtr)#redistribute connected
```

```
*Aug 14 11:44:00.182: tag=0, metric=1, prefix=123::/64
```

```
R2(config-rtr)#
```

```
*Aug 14 11:44:05.182: RIPng: Sending multicast update on Serial0/0/0.1 for R1R2
```

```
*Aug 14 11:44:05.182: src=FE80::2
```

```
*Aug 14 11:44:05.182: dst=FF02::9 (Serial0/0/0.1)
```

```
*Aug 14 11:44:05.182: sport=521, dport=521, length=72
```

```
*Aug 14 11:44:05.182: command=2, version=1, mbz=0, #rte=3
```

```
*Aug 14 11:44:05.182: tag=0, metric=1, prefix=123::/64
```

```
*Aug 14 11:44:05.182: tag=0, metric=1, prefix=24::/64
```

```
*Aug 14 11:44:05.182: tag=0, metric=1, prefix=102::/64
```

```
R1#show ipv6 route rip
```

IPv6 Routing Table – 8 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 24::/64 [120/2]
```

```
via FE80::2, Serial0/0/0
```

```
R 102::/64 [120/2]
```

```
via FE80::2, Serial0/0/0
```

```
R1#
```

Example 8-68 shows a similar configuration on R3, enabling the R1R3 RIPng process. Viewing the RIPng routes in R3's routing table confirms that it is also learning R2's redistributed routes, network 24::/64 and 102::/64. All three of the RIPng routers are advertising to the same multicast address using the same port number, and are thus learning from each other.

#### Example 8-68. RIPng Configuration and Verification on R3

```
R3(config)#ipv6 unicast-routing
```

```
R3(config)#interface s1/1.7
```

```
R3(config-subif)#ipv6 rip R1R3 enable
```

```
R3(config-subif)#end
```

```
R3#show ipv6 route rip
```

IPv6 Routing Table – 10 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 24::/64 [120/2]
```



```
via FE80::2, Serial1/1.7
R 102::/64 [120/2]
via FE80::2, Serial1/1.7
R3#
```

To segregate the RIPng domains, the port number and or the multicast group address parameters must be changed. Example 8-69 illustrates changing the UDP port number to 1013 on R3 and R1 (for the R1R3 process). Notice that the multicast group address was left as the default FF02::9. After the process has been changed, the routing tables on R1 and R3 are cleared and then checked; there are no longer any RIPng routes on either router. Therefore, R1 and R3 are no longer listening to the updates from R2, because they are using different ports.

Example 8-69. Changing Port Number for RIPng Process

Code View: Scroll / Show All

```
R3(config)#ipv6 router rip R1R3
```

```
R3(config-rtr)#port ?
```

```
<1-65535> UDP port
```

```
R3(config-rtr)#port 1013 multicast-group ?
```

```
X:X:X:X:X multicast address
```

```
R3(config-rtr)#port 1013 multicast-group FF02::9
```

```
R3(config-rtr)#
```

```
R1(config)#ipv6 router rip R1R3
```

```
R1(config-rtr)#port 1013 multicast-group FF02::9
```

```
R1(config-rtr)#^Z
```

```
R1#
```

```
R1#clear ipv6 route *
```

```
R1#
```

```
R3#clear ipv6 route *
```

```
R3#
```

```
R3#show ipv6 route rip
```

```
IPv6 Routing Table - 8 entries
```

```
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
```

```
U - Per-user Static route
```

```
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
```

```
O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
```

```
ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
```

```
R3#
```

```
R1#show ipv6 route rip
```

```
IPv6 Routing Table - 6 entries
```

```
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
```

```
U - Per-user Static route
```

```
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
```

```
O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
```

```
ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
```

```
R1#
```

To allow R1 and R3 to see R2's connected routes again, R1 is first configured to be part of the R1R2 process; it remains in the R1R3 process, too. Example 8-70 illustrates this configuration. R1 is then configured to redistribute the R1R3 routes into the R1R2 process, and to redistribute the R1R2 routes into the R1R3 process. The resulting routing table on R3 is also shown in the example. R3 is again learning R2's redistributed routes, network 24::/64 and 102::/64.

Example 8-70. Configuring and Verifying Redistribution on R1

```
R1(config)#interface s0/0/0
R1(config-if)#ipv6 rip R1R2 enable
R1(config-if)#exit
R1(config)#ipv6 router rip R1R2
R1(config-rtr)#redistribute rip R1R3
R1(config-rtr)#exit
R1(config)#ipv6 router rip R1R3
R1(config-rtr)#redistribute rip R1R2
R1(config-rtr)#

R3#show ipv6 route rip
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R 24::/64 [120/3]
 via FE80::1, Serial1/1.7
R 102::/64 [120/3]
 via FE80::1, Serial1/1.7
R3#
```

The metrics of redistributed routes can be changed with route maps. In this network, R3 is configured to redistribute its connected routes; the loopback is advertised with a metric of 5 hops and the lower serial network is advertised with a metric of 15 hops. Therefore, when these metrics get to R1, they will be 6 hops and 16 hops, respectively. Recall that for RIPng, just like for IPv4 RIP, 15 hops is the maximum, so R1 will treat the 16-hop route as unreachable. The configuration on R3 is shown in Example 8-71. First, the redistribute connected command is entered, with the reference to the route-map CONN\_PREFIX. This route map is then created, with two statements. The first matches an IPv6 address that is permitted by the ACL PREFIX\_34, and if there is a match the metric is set to 15 (hops). The second matches an IPv6 address that is permitted by the prefix list PREFIX\_103, and if there is a match the metric is set to 5 (hops). The ACL PREFIX\_34 and the prefix list PREFIX\_103 are then created to permit the appropriate addresses.

Example 8-71. Configuring Redistribution with a Route Map on R3

```
R3(config)#ipv6 router rip R1R3
R3(config-rtr)#redistribute connected route-map ?
WORD Pointer to route-map entries
R3(config-rtr)#redistribute connected route-map
CONN_PREFIX
R3(config-rtr)#exit
R3(config)#route-map CONN_PREFIX permit 10
```

```

R3(config-route-map)#match ipv6 address PREFIX_34
R3(config-route-map)#set metric 15
R3(config-route-map)#exit
R3(config)#route-map CONN_PREFIX permit 20
R3(config-route-map)#match ipv6 address prefix-list
PREFIX_103
R3(config-route-map)#set metric 5
R3(config-route-map)#exit
R3(config)#ipv6 access-list PREFIX_34
R3(config-ipv6-acl)#permit ipv6 34::/64 any
R3(config-ipv6-acl)#exit
R3(config)#ipv6 prefix-list PREFIX_103 permit 103::/64 any
R3(config)#

```

Debug is enabled on R3, and the resulting output is shown in Example 8-72. The debug output confirms that prefix 34 is being tagged with a metric of 15, and prefix 103 is being tagged with a metric of 5. The example also shows the RIPng routes in R1's routing table. R1 has learned about network 103 with a metric of 6. However, R1 has no knowledge of network 34 because it had reached a metric of 16, which is interpreted as unreachable. This example shows the importance of the seed metric value chosen when redistributing.

Example 8-72. Verifying Redistribution with a Route Map on R3

Code View: Scroll / Show All

```

R3(config)#do debug ipv6 rip
RIP Routing Protocol debugging is on
R3(config)#
*Mar 1 18:30:07: RIPng: Sending multicast update on Serial1/1.7 for R1R3
*Mar 1 18:30:07: src=FE80::3
*Mar 1 18:30:07: dst=FF02::9 (Serial1/1.7)
*Mar 1 18:30:07: sport=1013, dport=1013, length=72
*Mar 1 18:30:07: command=2, version=1, mbz=0, #rte=3
*Mar 1 18:30:07: tag=0, metric=15, prefix=34::/64
*Mar 1 18:30:07: tag=0, metric=5, prefix=103::/64
*Mar 1 18:30:07: tag=0, metric=1, prefix=123::/64
R3(config)#

```

**R1#show ipv6 route rip**

IPv6 Routing Table – 9 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```

R 24::/64 [120/2]
 via FE80::2, Serial0/0/0
R 102::/64 [120/2]
 via FE80::2, Serial0/0/0
R 103::/64 [120/6]
 via FE80::3, Serial0/0/0

```

R1#



From this example, we can summarize the issues and solutions observed as follows:

- Multiple RIPvng instances can run on the same router and on the same link.
- Multiple instances of RIPvng by default will send and receive advertisements between each other.
- Changing the port and or multicast group addresses separates the processes. Redistribution must be configured to share information between these separate processes.
- Seed metrics may need to be configured to control paths.

RIPvng and OSPFv3 Redistribution

The previous section examined redistribution between two RIPvng instances. This section explores redistribution between RIPvng and OSPFv3.

RIPvng and OSPFv3 Redistribution Configuration and Verification Commands

The redistribution configuration and verification in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

As noted earlier, there are many options for the redistribute router configuration command. The options explored in this section are the redistribute connected [metric metric-value] [metric-type type-value] [tag tag-value] [route-map map-tag] command, which redistributes connected routes into the OSPFv3 process, and the redistribute ospf [process-id] [metric metric-value} [metric-type type-value] [tag tag-value] [route-map map-tag], which redistributes the specified OSPF process into another routing process. The parameters in these commands are described in Table 8-14.

Table 8-14. *redistribute* Command for OSPFv3 Parameters

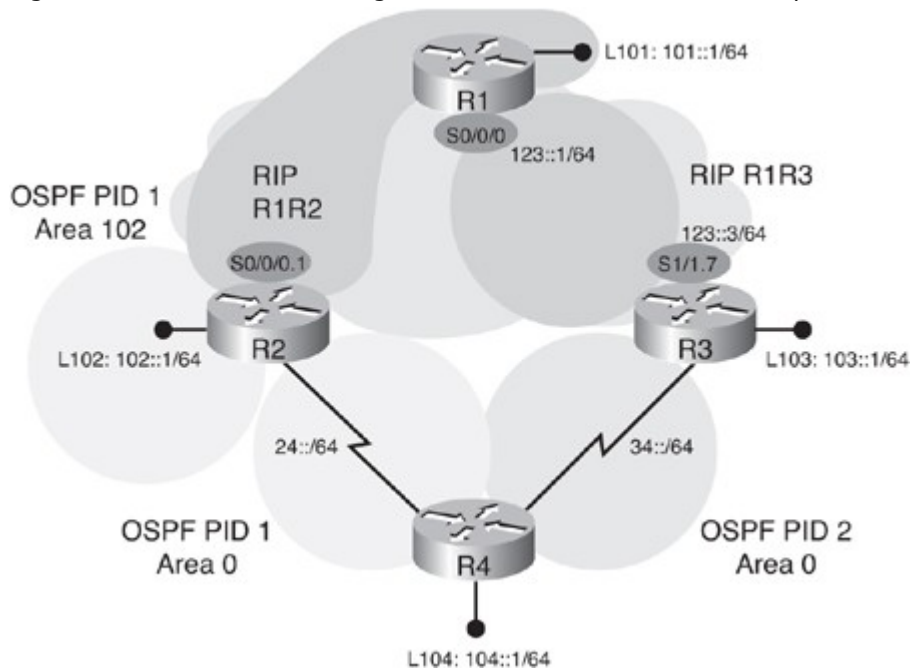
| Parameter                            | Description                                                                                                                                                                                                                               |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>metric</b> <i>metric-value</i>    | (Optional) Specifies the metric value for redistributed routes.                                                                                                                                                                           |
| <b>metric-type</b> <i>type-value</i> | (Optional) Specifies the external link type associated with the redistributed route advertised into the OSPF routing domain. It can be one of two values:<br>1—Type 1 external route<br>2—Type 2 external route<br>Type 2 is the default. |
| <b>tag</b> <i>tag-value</i>          | (Optional) Specifies the 32-bit decimal value attached to each external route.                                                                                                                                                            |
| <b>route-map</b> <i>map-tag</i>      | (Optional) Specifies the route map to be interrogated to filter the importation of routes from this source routing protocol to the current routing protocol.                                                                              |

The **distance** *distance* router configuration command configures the administrative distance of IPv6 routes.

RIPvng and OSPFv3 Redistribution Configuration and Verification Example

Figure 8-36 illustrates the network used for a RIPvng and OSPFv3 redistribution example. Routers R1, R2, and R3 are all configured for RIPvng the same as in the previous example, except that now the loopback interface on R1 is also included in the R1R2 process. R2, R3, and R4 are configured for OSPF as shown in the figure. Notice that there are two OSPF processes. Process 1 has two areas, area 0 and area 102. Process 2 has one area, area 0. Having two OSPF processes on R4 is not a problem, because OSPFv3 for IPv6, just like OSPFv2 for IPv4, maintains two different topology databases for the different processes. By default the two processes do not communicate or exchange routing updates.

Figure 8-36. Network for RIPng and OSPFv3 Redistribution Example.



In this example, redistributing connected interfaces into OSPFv3, redistributing between different OSPFv3 processes, and redistributing between OSPFv3 and RIPng, are demonstrated. Issues including metrics used in distribution, suboptimal routing, and routing loops are encountered and resolved.

To confirm that the mutual redistribution is configured and working between the two RIPng processes, the `show ipv6 protocols` and `show ipv6 ripcommands` are used. The results of these commands on R1 are displayed in Example 8-73. Notice that the multicast groups are the same but the ports are different.

Example 8-73. Verifying Redistribution on RIPng Routers

Code View: Scroll / Show All

**R1#show ipv6 protocols**

IPv6 Routing Protocol is "connected"

IPv6 Routing Protocol is "static"

IPv6 Routing Protocol is "rip R1R3"

Interfaces:

Serial0/0/0

Redistribution:

Redistributing protocol connected

Redistributing protocol rip R1R2

IPv6 Routing Protocol is "rip R1R2"

Interfaces:

Loopback101

Serial0/0/0

Redistribution:

Redistributing protocol rip R1R3

IPv6 Routing Protocol is "rip R1R2"

R1#

**R1#show ipv6 rip R1R2**

RIP process "R1R2", port 521, multicast-group FF02::9, pid 192

Administrative distance is 120. Maximum paths is 16

Updates every 30 seconds, expire after 180

Holddown lasts 0 seconds, garbage collect after 120

```

Split horizon is on; poison reverse is off
Default routes are not generated
Periodic updates 4819, trigger updates 56
Interfaces:
 Loopback101
 Serial0/0/0
Redistribution:
 Redistributing protocol rip R1R3
R1#
R1#show ipv6 rip R1R3
RIP process "R1R3", port 1013, multicast-group FF02::9, pid 39
 Administrative distance is 120. Maximum paths is 16
 Updates every 30 seconds, expire after 180
 Holddown lasts 0 seconds, garbage collect after 120
 Split horizon is on; poison reverse is off
 Default routes are not generated
 Periodic updates 4816, trigger updates 52
Interfaces:
 Serial0/0/0
Redistribution:
 Redistributing protocol connected
 Redistributing protocol rip R1R2
R1#

```

The first step in establishing end-to-end connectivity is to redistribute the connected interfaces into OSPFv3 on R3. This configuration is shown in Example 8-74. This configuration redistributes the connected routes with metric type 1, a tag of 3333, and using a route map called LOOPBACKS. Recall from Chapter 3 that routes with metric type 1 (displayed in the routing table as OE1) calculate the cost by adding the external cost to the internal cost of each link the packet crosses. Routes with metric type 2 (displayed in the routing table as OE2) are the default; the cost of these routes is always the external cost only. The route map LOOPBACKS matches routes permitted by the ACL L103, which permits only the loopback interface network to be redistributed into OSPFv3, 103::/64.

Example 8-74. Configuring Redistribution of Connected Routes on R3

```

Code View: Scroll / Show All
R3(config)#ipv6 router ospf 2
R3(config-rtr)#redistribute connected metric-type 1 tag 3333 route-map
LOOPBACKS
R3(config-rtr)#route-map LOOPBACKS permit 10
R3(config-route-map)#match ipv6 address L103
R3(config-route-map)#exit
R3(config)#ipv6 access-list L103
R3(config-ipv6-acl)#permit ipv6 103::/64 any
R3(config-ipv6-acl)#

```

Example 8-75 shows the resulting OSPFv3 routes in R4's routing table. The loopback network 103 has the 3333 tag and metric type 1 (OE1).

#### Example 8-75. Verifying Redistribution of Connected Routes on R4

```
R4#show ipv6 route ospf
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
 D - EIGRP, EX - EIGRP external
OI 102::1/128 [110/64]
 via FE80::2, Serial0/0/0.2
OE1 103::/64 [110/84], tag 3333
 via FE80::3, Serial0/0/0.3
R4#
```

R2 and R3 do not see any OSPFv3 networks yet, because R4 has not been configured to redistribute between its two OSPFv3 processes. Example 8-76 illustrates the configuration of the redistribution between the two OSPFv3 processes on R4. Notice that each process is redistributed into the other process, and connected routes are also redistributed into both processes.

#### Example 8-76. OSPFv3 Redistribution Configuration on R4

```
R4(config)#ipv6 router ospf 1
R4(config-rtr)#redistribute connected
R4(config-rtr)#redistribute ospf 2
R4(config-rtr)#ipv6 router ospf 2
R4(config-rtr)#redistribute connected
R4(config-rtr)#redistribute ospf 1
R4(config-rtr)#
```

Example 8-77 shows the resulting OSPFv3 routes in the routing tables on R2 and R3. Notice that all the routes appear on both routers. R2 sees the 103 network with a tag of 3333, but with a metric type 2; the redistribution process on R4 reset the metric type to the default type 2. R2 sees all other routes as type 2, as does R3. Because type 2 routes do not include the internal network cost, suboptimal routing could result.

#### Example 8-77. Verifying OSPFv3 Redistribution on R2 and R3

```
Code View: Scroll / Show All
R2#show ipv6 route ospf
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
 D - EIGRP, EX - EIGRP external
OE2 34::/64 [110/20]
 via FE80::4, Serial0/0/0.4
OE2 103::/64 [110/84], tag 3333
 via FE80::4, Serial0/0/0.4
OE2 104::/64 [110/20]
 via FE80::4, Serial0/0/0.4
R2#
```



```
R3(config-rtr)#do show ipv6 route ospf
```

IPv6 Routing Table – 12 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

D – EIGRP, EX – EIGRP external

```
OE2 24::/64 [110/20]
```

```
via FE80::4, Serial0/0/0.4
```

```
OE2 102::1/128 [110/64]
```

```
via FE80::4, Serial0/0/0.4
```

```
OE2 104::/64 [110/20]
```

```
via FE80::4, Serial0/0/0.4
```

```
R3(config-rtr)#
```

The next step is to configure mutual redistribution between RIPng and OSPFv3, starting on R2 as shown in Example 8-78. The OSPFv3 and connected routes are redistributed into RIPng. The RIPng and connected routes are redistributed into OSPFv3. The example also shows the same configuration on the other edge router, R3.

Example 8-78. Configuring RIPng and OSPFv3 Redistribution on R2 and R3

```
R2(config)#ipv6 router rip R1R2
```

```
R2(config-rtr)#redistribute ospf 1
```

```
R2(config-rtr)#redistribute connected
```

```
R2(config-rtr)#ipv6 router ospf 1
```

```
R2(config-rtr)#redistribute rip R1R2
```

```
R2(config-rtr)#redistribute connected
```

```
R3(config)#ipv6 router ospf 2
```

```
R3(config-rtr)#redistribute connected
```

```
R3(config-rtr)#redistribute rip R1R3
```

```
R3(config-rtr)#ipv6 router rip R1R3
```

```
R3(config-rtr)#redistribute connected
```

```
R3(config-rtr)#redistribute ospf 2
```

The show ipv6 route rip command is used on R1 to examine the RIPng routes and to verify the configuration; the output is displayed in Example 8-79. In this output the redistributed connected networks (the loopbacks on R2 and R3) appear, but the redistributed OSPFv3 network (the loopback on R4) is not there. The example also shows the result of enabling the debug ipv6 rip on R1 to see what is happening. This debug output shows that all the routes being redistributed from OSPFv3 into RIPng have a metric of 16, which in RIPng is unreachable. These routes are therefore being discarded by RIPng.

Example 8-79. Verifying RIPng and OSPFv3 Redistribution on R1

Code View: Scroll / Show All

```
R1#show ipv6 route rip
```

IPv6 Routing Table – 10 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

```

U - Per-user Static route
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D - EIGRP, EX - EIGRP external
R 24::/64 [120/2]
 via FE80::2, Serial0/0/0
R 34::/64 [120/2]
 via FE80::3, Serial0/0/0
R 102::/64 [120/2]
 via FE80::2, Serial0/0/0
R 103::/64 [120/2]
 via FE80::3, Serial0/0/0
R1#
R1#debug ipv6 rip
RIP Routing Protocol debugging is on
R1#
RIPng: Next RIB walk in 171112
RIPng: response received from FE80::3 on Serial0/0/0 for R1R3
 src=FE80::3 (Serial0/0/0)
 dst=FF02::9
 sport=1013, dport=1013, length=152
 command=2, version=1, mbz=0, #rte=7
 tag=0, metric=1, prefix=123::/64
 tag=0, metric=1, prefix=34::/64
 tag=0, metric=1, prefix=103::/64
 tag=0, metric=16, prefix=24::/64
 tag=0, metric=16, prefix=104::/64
 tag=0, metric=16, prefix=102::1/128
 tag=0, metric=16, prefix=101::/64

```

To fix this issue, the metric needs to be specified when OSPFv3 routes are redistributed into RIPng on both R2 and R3; these configurations with a metric of 1 are shown in Example 8-80. The example also shows the R1 routing table again, confirming that R1 now sees the redistributed OSPFv3 routes, including to the 104 network. A ping from the loopback of R4 to the loopback of R1 verifies end-to-end connectivity.

Example 8-80. Configuring and Verifying Redistribution Metric

```

Code View: Scroll / Show All
R2(config)#ipv6 router rip R1R2
R2(config-rtr)#redistribute ospf 1 metric 1
R2(config-rtr)#

R3(config)#ipv6 router rip R1R3
R3(config-rtr)#redistribute ospf 2 metric 1
R3(config-rtr)#

R1# show ipv6 route rip

```

IPv6 Routing Table – 12 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

D – EIGRP, EX – EIGRP external

```
R 24::/64 [120/2]
 via FE80::2, Serial0/0/0
R 34::/64 [120/2]
 via FE80::3, Serial0/0/0
R 102::/64 [120/2]
 via FE80::2, Serial0/0/0
R 102::1/128 [120/2]
 via FE80::3, Serial0/0/0
R 103::/64 [120/2]
 via FE80::3, Serial0/0/0
R 104::/64 [120/2]
 via FE80::2, Serial0/0/0
```

R1#

```
R4#ping 101::1 source 104::1
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 101::1, timeout is 2 seconds:

Packet sent with a source address of 104::1

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

R4#

The trace command is used on R4 to examine the path that packets to R1's loopback are taking; the results are shown in Example 8-81. The path goes through R2 (24::2) and then to R1, which is the optimal path. The example also shows part of R3's routing table (using the command and regular expression section 10[14]::64, which means that only networks 101 or 104 are displayed). As this output reveals, R3 is learning about both 101 and 104 via R4 with OSPFv3, even though it would be better to get to 101 via R1 with RIPng. This is suboptimal routing. A trace from R3 to the 101 loopback confirms that it goes via R4, to R2, and then to R1.

Example 8-81. Examining Resulting Paths

```
R4#trace 101::1
```

Type escape sequence to abort.

Tracing the route to 101::1

```
 1 24::2 20 msec 20 msec 24msec
 2 101::1 44 msec 44 msec 44 msec
```

R4#

```
R3#show ipv6 route | section
10[14]::/64
```

```
OE2 101::/64 [110/20]
```

```

 via FE80::4, Serial0/0/0.4
OE2 104::/64 [110/20]
 via FE80::4, Serial0/0/0.4
R3#
R3#trace 101::1
Type escape sequence to abort.
Tracing the route to 101::1
 1 34::4 20 msec 28 msec 24msec
 2 24::2 44 msec 44 msec 44 msec
 3 101::1 40 msec 36 msec 40 msec
R3#

```

Although R3 is learning about 101 from both RIPng and OSPFv3, the OSPFv3 route is in the routing table because it has a lower administrative distance of 110 compared with the administrative distance of 120 for RIPng. This problem is easily fixed and will be done in an upcoming example.

However, there are other problems caused by the feedback loop created by redistributing from OSPFv3 into RIPng and from RIPng into OSPFv3. One issue is that the two edge routers are redistributing OSPFv3 routes into RIPng, and those same routes are being redistributed from RIPng into OSPFv3. Similarly, RIPng routes are being redistributed into OSPFv3 and then being redistributed back into RIPng network, with a better administrative distance. The solution to this issue is the following:

- Redistribute OSPFv3 routes into RIPng, but prevent those same routes from going back into OSPFv3, and
- Redistribute RIPng routes into OSPFv3, but prevent those same routes from going back into RIPng.

Route filtering can be configured to implement this solution.

There is another problem in this network. As seen earlier, when the network is stable, there is suboptimal routing. However, consider what happens when the network is not stable, such as when an interface goes down. Example 8-82 illustrates this scenario. First, debugging is enabled on R3, and then the loopback interface on R4 is shutdown. The resulting debug output on R3 illustrates that the OSPFv3 route to 104 is deleted from the routing table. However, the entry is then re-entered into R3's routing table as a RIPng route, via R1. Even though the 104 network is down, R3 has a route to it.

Example 8-82. Examining Behavior When a Network Goes Down

```

Code View: Scroll / Show All
R3#debug ipv6 routing
IPv6 routing table events debugging is on
R3#

R4(config)#interface loopback 104
R4(config-if)#shutdown
R4(config-if)#

R3#
IPV6RTO: ospf 2, Deleting all next-hops for 104::/64
IPV6RTO: ospf 2, Delete 104::/64 from table
IPV6RTO: rip R1R3, Backup call for 104::/64
IPV6RTO: rip R1R3, Route add 104::/64 [new]
IPV6RTO: rip R1R3, Add 104::/64 to table
IPV6RTO: rip R1R3, Adding next-hop FE80::1 over Serial 0/0/0.123 for 104::/64,
[120/3]
IPV6RTO: Event: 104::/64, Del, owner ospf, previous None

```

IPV6RTO: Event: 104::/64, Add, owner rip, previous None

To determine where this route is coming from, debugs are disabled on R3, and part of the R3 routing table is displayed; the results are shown in Example 8-83. The 104 network is in the routing table, as a RIPng route via R1. So, the R1 routing table is displayed next. R1 has the 104 network as a RIPng route via R2. Therefore, the R2 routing table is displayed next; R2 has the 104 network as an OSPFv3 external route, via R4. Finally, R4's routing table reveals that it sees its own loopback network 104 as an OSPFv3 external route via R3. Therefore, there is a routing loop.

Example 8-83. Following the Routing Loop

Code View: Scroll / Show All

R3#**undebug all**

All possible debugging has been turned off

R3#**show ipv6 route | section 10[14]::/64**

OE2 101::/64 [110/20]

via FE80::4, Serial0/0/0.4

R 104::/64 [120/3]

via FE80::1, Serial0/0/0.123

R3#

R1#**show ipv6 route | section 10[14]::/64**

C 101::/64 [0/0]

via ::, Loopback101

R 104::/64 [120/2]

via FE80::2, Serial0/0/0

R1#

R2#**show ipv6 route | section 10[14]::/64**

R 101::/64 [120/2]

via FE80::1, Serial0/0/0.123

OE2 104::/64 [110/20]

via FE80::4, Serial0/0/0.4

R2#

R4#**show ipv6 route | section 10[14]::/64**

OE2 101::/64 [110/20]

via FE80::2, Serial0/0/0.2

OE2 104::/64 [110/20]

via FE80::3, Serial0/0/0.3

R4#

The loop is verified with a trace from R4, as shown in Example 8-84.

Example 8-84. Confirming the Routing Loop

Code View: Scroll / Show All

R4#**trace 104::1**

Type escape sequence to abort.

Tracing the route to 104::1

```
 1 34::3 24 msec 24 msec 20 msec
 2 123::1 44 msec 44 msec 44 msec
 3 123::2 36 msec 40 msec 40 msec
 4 24::4 32 msec 32 msec 32 msec
 5 34::3 56 msec 56 msec 52 msec
 6 123::1 76 msec 76 msec 76 msec
 7 123::2 68 msec 72 msec 72 msec
 8 24::4 64 msec 64 msec 64 msec
 9 34::3 88 msec 88 msec 84 msec
10 123::1 108 msec 108 msec 108 msec
11 123::2 104 msec 104 msec 100 msec
12 24::4 100 msec 96 msec 96 msec
13 34::3 120 msec 120 msec 116 msec
14 123::1 140 msec 140 msec 160 msec
15 123::2 136 msec 132 msec 136 msec
16 24::4 140 msec 128 msec 132 msec
17 34::3 148 msec 152 msec 152 msec
18 123::1 172 msec 172 msec 172 msec
19 123::2 168 msec 164 msec 164 msec
20 24::4 164 msec 160 msec 160 msec
21 34::3 184 msec 184 msec 180 msec
22 123::1 204 msec 204 msec 204 msec
23 123::2 200 msec 200 msec 200 msec
24 24::4 192 msec 196 msec 192 msec
25 34::3 216 msec 212 msec 212 msec
26 123::1 240 msec 236 msec 236 msec
27 123::2 232 msec 228 msec 232 msec
28 24::4 224 msec 224 msec 224 msec
29 34::3 248 msec 248 msec 244 msec
30 123::1 268 msec 268 msec 272 msec
```

**Destination not found inside max hopcount diameter.**

R4#

We start by first fixing the routing loop by configuring route filtering on both R2 and R3. As shown in Example 8-85, the R4 loopback interface is reenabled, and then R2 and R3 are configured. In the RIPng process, the redistribute command on these routers is modified to refer to a route map OSPF\_TO\_RIP. The route map denies routes matched by the RIP\_NATIVE prefix list from being redistributed, and permits all other routes to be redistributed. The prefix list identifies (permits) the R1 loopback network 101. Therefore, network 101 will be denied from being redistributed into RIPng, because it should be a RIPng learned route. All other routes will be redistributed.

In this example, R1 is also configured, to filter routes going between the two RIPng processes. In each process, the **redistribute** command is modified to refer to a route map RIP\_NOT\_OSPF (because it is allowing only the RIPng prefixes, not any of the OSPF-learned prefixes). The route map permits only routes that match the prefix list RIP\_NATIVE to be redistributed; only the 101 network matches. Therefore, only the RIPng network 101 is redistributed between the two RIPng processes, no other routes (which are the ones that came from OSPFv3) will be redistributed.

#### Example 8-85. Configuring Filtering to Prevent Routing Loop

Code View: Scroll / Show All

```
R4(config)#interface loopback 104
```

```
R4(config-if)#no shutdown
```

```
R4(config-if)#
```

```
R2(config)#ipv6 router rip R1R2
```

```
R2(config-rtr)#redistribute ospf 1 metric 1 route-map OSPF_TO_RIP
```

```
R2(config-rtr)#route-map OSPF_TO_RIP deny 10
```

```
R2(config-route-map)#match ipv6 address prefix-list RIP_NATIVE
```

```
R2(config-route-map)#route-map OSPF_TO_RIP permit 20
```

```
R2(config-route-map)#ipv6 prefix-list RIP_NATIVE seq 5 permit
101::/64
```

```
R2(config)#
```

```
R3(config)#ipv6 router rip R1R3
```

```
R3(config-rtr)#redistribute connected
```

```
R3(config-rtr)#redistribute ospf 2 metric 1 route-map OSPF_TO_RIP
```

```
R3(config-rtr)#route-map OSPF_TO_RIP deny 10
```

```
R3(config-route-map)#match ipv6 address prefix-list RIP_NATIVE
```

```
R3(config-route-map)#route-map OSPF_TO_RIP permit 20
```

```
R3(config-route-map)#ipv6 prefix-list RIP_NATIVE seq 5 permit
101::/64
```

```
R3(config)#
```

```
R1(config)#ipv6 router rip R1R3
```

```
R1(config-rtr)#redistribute rip R1R2 route-map RIP_NOT_OSPF
```

```
R1(config-rtr)#ipv6 router rip R1R2
```

```
R1(config-rtr)#redistribute rip R1R3 route-map RIP_NOT_OSPF
```

```
R1(config-rtr)#route-map RIP_NOT_OSPF permit 10
```

```
R1(config-route-map)#match ipv6 address prefix-list RIP_NATIVE
```

```
R1(config-route-map)#route-map RIP_NOT_OSPF deny 20
```

```
R1(config-route-map)#ipv6 prefix-list RIP_NATIVE seq 5 permit
101::/64
```

```
R1(config)#
```

To verify that the routing loop is fixed, debugging is enabled on R3 and the R4 loopback interface is shut down again. These commands and the resulting debug output on R3 are shown in Example 8-86. This time the OSPFv3 route is deleted, and no route is added to the routing table. The routing tables on R3, R1, and R2 are checked and confirm that the routing loop problem was solved with the route filtering. The R4 loopback interface is re-enabled, and the debug output on R3 confirms the route is added.

#### Example 8-86. Verifying That Filtering Did Prevent Routing Loop

Code View: Scroll / Show All

```
R3#debug ipv6 routing
```

```
IPv6 routing table events debugging is on
```

```
R3#
```



```

R4(config)#interface loopback 104
R4(config-if)#shutdown
R4(config-if)#

R3#
IPV6RTO: ospf 2, Deleting all next-hops for 104::/64
IPV6RTO: ospf 2, Delete 104::/64 from table
IPV6RTO: Event: 104::/64. Del, owner ospf, previous None
R3#
R3#show ipv6 route | section 10[14]::/64
OE2 101::/64 [110/20]
 via FE80::4, Serial0/0/0.4
R3#

R1# show ipv6 route | section 10[14]::/64
%SYS-5-CONFIG_I: Configured from console by console
C 101::/64 [0/0]
 via ::, Loopback101
R1#

R2# show ipv6 route | section 10[14]::/64
R 101::/64 [120/2]
 via FE80::1, Serial0/0/0.123
R2#

R4(config)#interface loopback 104
R4(config-if)#noshutdown
R4(config-if)#

R3#
IPV6RTO: ospf 2, Route add 104::/64 [new]
IPV6RTO: ospf 2, Add 104::/64 to table
IPV6RTO: ospf 2, Adding next-hop FE80::4 over Serial0/0/0.4 for 104::/64, [110/20]
IPV6RTO: Event: 104::/64. Add, owner ospf, previous None
R3#

```

The suboptimal routing problem is the next to fix. As shown in Example 8-87, debugging is disabled on R3, and part of its routing table is displayed. The route from R3 to R1's loopback 101 network is going via R4. The solution is to give RIPng preference, with a lower administrative distance, for those networks that should be learned via RIPng. This is accomplished by configuring the distance command under the RIPng process. The example shows this configuration on R3 and R2; the administrative distance is set to 109, to make it lower than the OSPFv3 administrative distance 110. The R2 and R3 routing tables confirm that R2 and R3

now reach the 101 network via RIPng. A ping from R4's loopback to network 101 on R1 confirms that the network has full end-to-end connectivity.

#### Example 8-87. Fixing and Verifying Suboptimal Routing Issue

Code View: Scroll / Show All

R3#**undebg all**

All possible debugging has been turned off

R3#**show ipv6 route | section 10[14]::/64**

OE2 101::/64 [110/20]

via FE80::4, Serial0/0/0.4

R 104::/64 [120/3]

via FE80::1, Serial0/0/0.123

R3#

R3#**conf t**

R3(config)#**ipv6 router rip R1R3**

R3(config-rtr)#**distance 109**

R3(config-rtr)#

R2(config)#**ipv6 router rip R1R2**

R2(config-rtr)#**distance 109**

R2(config-rtr)#**end**

R2#**show ipv6 route | section 10[14]::/64**

R 101::/64 [109/2]

via FE80::1, Serial0/0/0.123

OE2 104::/64 [110/20]

via FE80::4, Serial0/0/0.4

R2#

R3#**show ipv6 route | section 10[14]::/64**

R 101::/64 [109/2]

via FE80::1, Serial0/0/0.123

OE2 104::/64 [110/20]

via FE80::4, Serial0/0/0.4

R3#

R4#**ping 101::1 source 104::1**

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 101::1, timeout is 2 seconds:

Packet sent with a source address of 104::1

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

R4#

To summarize, the issues and solutions observed in this example are as follows:

- OSPFv3 does not redistribute connected networks by default.
- The metric and metric type may be reset when redistributing. The solution is to explicitly configure the seed metric and metric type.

- Redistribution may cause suboptimal routing and routing loops. Solutions include changing administrative distance and route filtering.

#### RIPng, OSPFv3, and MBGP Redistribution

The previous section examined redistribution between RIPng and OSPFv3. This section explores redistribution between RIPng, OSPFv3, and MBGP.

#### RIPng, OSPFv3, and MBGP Redistribution Configuration and Verification Commands

The redistribution configuration and verification in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

As noted earlier, there are many options for the **redistribute** router configuration command. The option explored in this section is the **redistribute bgp** *autonomous-system* address family (or router) configuration command, which redistributes BGP into the routing process. The *autonomous-system* identifies the local autonomous system.

The **bgp redistribute-internal** address family (or router) configuration command configures IBGP redistribution into an IGP.

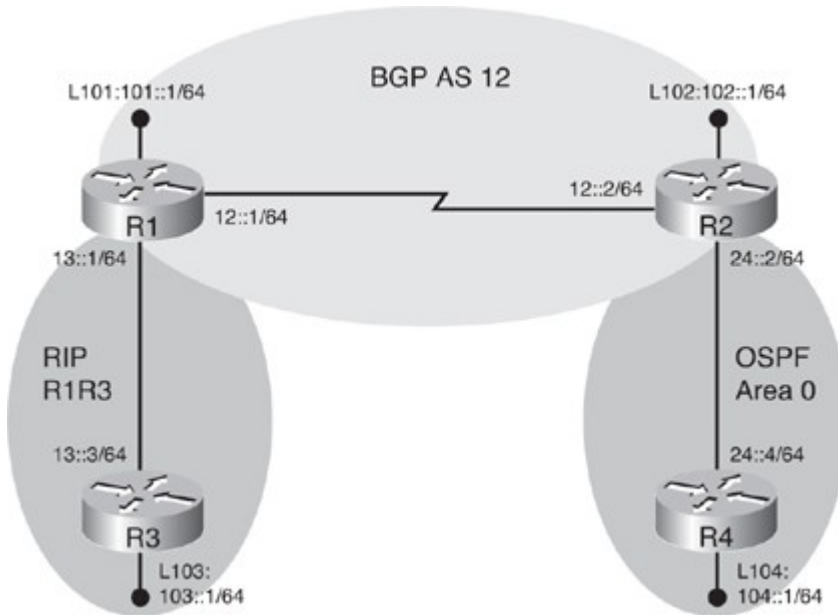
The **show ip bgp summary** command displays entries in the BGP table.

The **show bgp ipv6 unicast summary** command displays a summary of unicast entries in the IPv6 BGP routing table.

#### RIPng, OSPFv3, and MBGP Redistribution Configuration and Verification Example

Figure 8-37 illustrates the network used for a RIPng, OSPFv3, and MBGP redistribution example. All IPv6 addresses shown are already configured on the routers. RIPng R1R3 will run between routers R1 and R3. MBGP AS 12 will run between routers R1 and R2. OSPFv3 will run between routers R2 and R4. Each router's loopback interface will be included in the appropriate routing protocol configuration.

Figure 8-37. Network for RIPng, OSPFv3, and MBGP Redistribution Example.



The goal is to have end-to-end reachability. Example 8-88 illustrates the configuration of R1; IPv6 unicast routing is enabled and RIPng R1R3 is configured on the Fast Ethernet 0/0 interface. The example also shows the R3 configuration; both the Fast Ethernet 0/0 and Loopback 103 interfaces are included in the RIPng process. The configuration is verified by viewing the RIPng routes in R1's routing table. R1 does have R3's loopback network 103 in its routing table.

#### Example 8-88. Configuring RIPng on R1 and R3

```
R1(config)#ipv6 unicast-routing
R1(config)#interface fa0/0
R1(config-if)#ipv6 rip R1R3 enable
R1(config-if)#
```

```

R3(config)#ipv6 unicast-routing
R3(config)#interface fa0/0
R3(config-if)#ipv6 rip R1R3 enable
R3(config-subif)#exit
R3(config)#interface loopback 103
R3(config-if)#ipv6 rip R1R3 enable
R3(config-if)#

```

```

R1(config-if)#do show ipv6 route rip
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R 103::/64 [120/2]
 via FE80::219:55FF:FEDF:AD22, FastEthernet0/0
R1(config-if)#

```

Example 8-89 illustrates the OSPFv3 configuration on R4. IPv6 unicast routing is enabled, the router ID is configured, and OSPF process 1 for area 0 is configured on the Fast Ethernet 0/0 and loopback 104 interfaces. A similar configuration is shown for R2. The OSPFv3 neighbor relationship goes to the FULL state. The OSPFv3 routes in R2's routing table and R2's OSPFv3 neighbor table are also shown in the example, confirming that the routers are neighbors and that R2 is learning R4's loopback address.

Example 8-89. Configuring OSPFv3 on R4 and R2

Code View: Scroll / Show All

```

R4(config)#ipv6 unicast-routing
R4(config)#ipv6 router ospf 1
R4(config-rtr)#router-id 4.4.4.4
R4(config-rtr)#exit
R4(config)#interface fa0/0
R4(config-if)#ipv6 ospf 1 area 0
R4(config-if)#exit
R4(config)#interface loopback 104
R4(config-if)#ipv6 ospf 1 area 0
R4(config-if)#exit

```

```

R2(config)#ipv6 unicast-routing
R2(config)#ipv6 router ospf 1
R2(config-rtr)#
*Aug 16 03:12:47.369: %OSPFv3-4-NORTRID: OSPFv3 process 1 could not pick a
router-id, please configure manually
R2(config-rtr)#router-id 2.2.2.2
R2(config-rtr)#exit
R2(config)#interface fa0/0
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#^Z

```

```

R2#
*Aug 16 03:13:08.757: %OSPFv3-5-ADJCHG: Process 1, Nbr 4.4.4.4 on
FastEthernet0/0 from LOADING to FULL, Loading Done
*Aug 16 03:13:10.053: %SYS-5-CONFIG_I: Configured from console by console
R2#
R2#show ipv6 route ospf
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O 104::1/128 [110/1]
 via FE80::4, FastEthernet0/0
R2#
R2#show ipv6 ospf neighbor

```

Example 8-90 shows the configuration of MBGP AS 12 on R1. IPv6 is both the carrier and passenger protocol for this network. The BGP router ID is configured first. The neighbor remote-as command is then configured to establish a neighbor relationship with R2, using its global unicast address 12::2; R2 is in the same AS. The passenger protocol (IPv6) is configured next, within the address-family ipv6 unicast configuration. The neighbor 12::2 (R2) is activated to carry IPv6 as the passenger protocol, and the network command is configured to advertise R1's loopback 101 network.

Note

Remember that the **network** command for BGP is used only to advertise BGP networks from the autonomous system.

Example 8-90. Configuring BGP on R1 and R2

```

Code View: Scroll / Show All
R1(config)#router bgp 12
R1(config-router)#bgp router-id 1.1.1.1
R1(config-router)#neighbor 12::2 remote-as 12
R1(config-router)#address-family ipv6 unicast
R1(config-router-af)#neighbor 12::2 activate
R1(config-router-af)#network 101::/64
R1(config-router-af)#exit
R1(config-router)#exit
R1(config)#

R2(config)#router bgp 12
R2(config-router)#bgp router-id 2.2.2.2
R2(config-router)#neigh 12::1 remote-as 12
R2(config-router)#address-family ipv6 unicast
R2(config-router-af)#
*Aug 16 03:41:50.584: %BGP-5-ADJCHANGE: neighbor 12::1 Up
R2(config-router-af)#neighbor 12::1 activate
*Aug 16 03:42:18.692: %BGP-5-ADJCHANGE: neighbor 12::1 Down Address family
activ

```

```
*Aug 16 03:42:20.728: %BGP-5-ADJCHANGE: neighbor 12::1
R2(config-router-af)#network 102::/64
R2(config-router-af)#
```

Example 8-90 also shows R2's similar BGP configuration. Notice the first message indicating that the BGP adjacency is up; this is the carrier protocol adjacency that is up. After the neighbor is activated and the network advertised, the passenger protocol adjacency also comes up.

#### Caution

Be careful when configuring MBGP commands. Because most of the commands are available in both router configuration mode and address family configuration mode, it is quite easy to be in the wrong mode when configuring. For example, if a route map filter is configured in the main BGP mode rather than in the address family mode, the filter would apply to the carrier protocol instead of to the passenger protocol.

Verifying this BGP configuration again requires understanding of the carrier and passenger protocols. In Example 8-91, the `show ip bgp summary` command output is displayed. It confirms that R2 has a neighbor R1, but it does not show any prefixes being advertised (the State/PfxRcd column has a 0). (Notice that this is the same command as used in IPv4 BGP.) This command is showing the carrier protocol; R2 does have a neighbor R1, but the two routers do not advertise their carrier protocol prefixes, they advertise their passenger protocol prefixes. The example also shows other commands that can be used to view BGP passenger protocol information with the `show bgp ?` output (for example, the `show bgp vpnv4` command is for MPLS VPNs). To see the advertised IPv6 prefixes, we need to see the passenger protocol; this is observed with the `show bgp ipv6 unicast summary` command, as shown in the example. In this command output, the same neighbor relationship is confirmed, but this time the prefix count is 1; R2 is learning one prefix from R1.

#### Example 8-91. Verifying BGP on R2

```
Code View: Scroll / Show All
R2#show ip bgp summary
BGP router identifier 2.2.2.2, local AS number 12
BGP table version is 1, main routing table version 1
Neighbor V AS MsgRcvd MsgSent Tblver Inq OutQ Up/Down
State/PfxRcd
12::1 4 12 10 10 1 0 0 00:01:10 0
R2#
R2#show bgp ?
all All address families
ipv4 Address family
ipv6 Address family
nsap Address family
vpnv4 Address family
R2#show bgp ipv6 unicast ?
X:X:X:X:X/<0-128> IPv6 prefix <network>/<length>
community Display routes matching the communities
community-list Display routes matching the community-list
dampening Display detailed information about dampening
extcommunity-list Display routes matching the extcommunity-list
filter-list Display routes confirming to the filter-list
inconsistent-as Display only routes with inconsistent origin ASs
labels Display BGP labels for prefixes
neighbors Detailed information on TCP and BGP neighbor connections
paths Path information
```

```

peer-group Display information on peer-groups
pending-prefixes Display prefixes pending deletion
prefix-list Display routes matching the prefix-list
quote-regexp Display routes matching the AS path "regular expression"
regexp Display routes matching the AS path regular expression
replication Display replication status of update-group(s)
rib-failure Display bgp routes that failed to install in the routing
 table (RIB)
route-map Display routes matching the route-map
summary Summary of BGP neighbor status
update-group Display information on update-groups
| Output modifiers

```

#### **R2#show bgp ipv6 unicast summary**

```

BGP router identifier 2.2.2.2, local AS number 12
BGP table version is 3, main routing table version 3
2 network entries using 298 bytes of memory
2 path entries using 152 bytes of memory
3/2 BGP path/bestpath attribute entries using 372 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 822 total bytes of memory
BGP activity 2/0 prefixes, 2/0 paths, scan interval 60 seconds
Neighbor V AS MsgRcvd MsgSent Tblver Inq OutQ Up/Down State/PfxRcd
12::1 4 12 11 11 3 0 0 00:02:30 1
R2#

```

Notice that when IPv6 is used as both the carrier and passenger protocol, the **show ip bgp summary** command displays the carrier protocol neighbor relationship information, while the **show bgp ipv6 unicast summary** command displays the passenger protocol neighbor relationship information.

Of course, the routing table is also a good place to verify configuration. Example 8-92 shows the BGP routes in R2's routing table, confirming that it is learning R1's loopback network.

Example 8-92. BGP Routes in R2's Routing Table

#### **R2#show ipv6 route bgp**

```

IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
B 101::/64 [200/0]
 via 12::1
R2#

```

With all three routing protocols configured and working, redistribution is configured next. In this example, R1 will redistribute between RIPng and MBGP, and R2 will redistribute between OSPFv3 and MBGP. Example 8-93 shows the R1 configuration to redistribute the RIPng R1R3 process into BGP. A new keyword, include-connected, is used in this redistribute command. This keyword causes both RIPng and connected routes to be redistributed, and saves adding the redistribute connected command. (For example R1 is not learning network 101 or 13 via RIPng so they would not be redistributed by default.) The configuration is verified by



viewing the BGP routes in R2's routing table; R2 has R1's loopback, R3's loopback and the network connecting R1 and R3 as BGP routes.

Note

The administrative distance of these BGP routes is 200, indicating that they were learned via IBGP.

Notice that the configuration for redistribution into BGP is done in the address family configuration mode, not in the main BGP configuration mode, since the redistribution is into the passenger protocol.

Example 8-93. Configuring and Verifying RIPng Redistribution into BGP on R1

```
R1(config-router)#address-family ipv6 unicast
R1(config-router-af)#redistribute rip R1R3 include-connected
R1(config-router-af)#

R2#show ipv6 route bgp
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
B 13::/64 [200/0]
 via 12::1
B 101::/64 [200/0]
 via 12::1
B 103::/64 [200/2]
 via 12::1
R2#
```

Example 8-94 illustrates R2's configuration to redistribute OSPFv3 into MBGP. Again the redistribution is performed in address family configuration mode, and again the include-connected keyword is used to automatically advertise connected routes along with the OSPFv3 learned routes. (For example, R2 does not learn network 24 and 102 by OSPFv3 and would not redistribute them by default.) The example also shows the BGP routes in R1's routing table, confirming that it has R2's loopback, R4's loopback and the network connecting R2 and R4 as BGP routes.

Example 8-94. Configuring and Verifying OSPFv3 Redistribution into BGP on R2

```
R2(config)#router bgp 12
R2(config-router)#address-family ipv6 unicast
R2(config-router-af)#redistribute ospf 1 include-connected
R2(config-router-af)#

R1(config-router-af)#do show ipv6 route bgp
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
B 24::/64 [200/0]
 via 12::2
B 102::/64 [200/0]
```

```
via 12::2
B 104::1/128 [200/1]
via 12::2
```

```
R1(config-router-af)#
```

Redistribution also needs to be configured in the other direction, from MBGP into RIPng and OSPFv3. Example 8-95 illustrates the configuration on R1, redistributing MBGP into RIPng; the include-connected keyword is used again. The bgp redistribute-internal command is needed in the address family configuration because IBGP learned networks are not redistributed into an IGP, such as RIPng, by default. This command forces the IBGP routes to be redistributed. The example also shows the RIPng routes in R3's routing table, confirming that R3 has R2's loopback, R4's loopback and the network connecting R2 and R4 as RIPng routes. However, notice that R3 does not see R1's directly connected networks.

Example 8-95. Configuring and Verifying BGP Redistribution into RIPng on R1

Code View: Scroll / Show All

```
R1(config)#ipv6 router rip R1R3
R1(config-rtr)#redistribute bgp 12 include-connected
R1(config-rtr)#exit
R1(config)#
R1(config)#router bgp 12
R1(config-router)#address-family ipv6 unicast
R1(config-router-af)#bgp redistribute-internal
R1(config-router-af)#
```

```
R3(config-if)#do show ipv6 route rip
```

IPv6 Routing Table - 9 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP

U - Per-user Static route

I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary

O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2

ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2

```
R 24::/64 [120/2]
via FE80::1, FastEthernet0/0
R 102::/64 [120/2]
via FE80::1, FastEthernet0/0
R 104::1/128 [120/2]
via FE80::1, FastEthernet0/0
R3(config-if)#
```

R3 does not see R1's directly connected networks because it turns out that the **include-connected** keyword does *not* actually work when redistributing from BGP into another protocol (even though it is an option on the command).

Therefore, the redistribute connected command must be configured on R1, as shown in Example 8-96. The example also shows the RIPng routes in R3's routing table, confirming that R3 is learning all the routes in the network.

Example 8-96. Correcting and Verifying BGP Redistribution into RIPng on R1

```
R1(config)#ipv6 router rip R1R3
```

```
R1(config-rtr)#redistribute connected
```

```
R1(config-rtr)#
```

```
R3(config-if)#do show ipv6 route rip
```

IPv6 Routing Table – 11 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 12::/64 [120/2]
```

via FE80::1, FastEthernet0/0

```
R 24::/64 [120/2]
```

via FE80::1, FastEthernet0/0

```
R 101::/64 [120/2]
```

via FE80::1, FastEthernet0/0

```
R 102::/64 [120/2]
```

via FE80::1, FastEthernet0/0

```
R 104::1/128 [120/2]
```

via FE80::1, FastEthernet0/0

```
R3(config-if)#
```

The final step is to configure MBGP redistribution into OSPFv3 on R2, as shown in Example 8-97. The BGP autonomous system is redistributed into the OSPFv3 process (without the include-connected keyword) and the redistributed connected command is added. The bgp redistribute-internal command is configured in the address family configuration. The example also shows the OSPFv3 routes in R4's routing table, confirming that R4 has all the routes. Pings from R4 to R3's loopback and from R4's loopback to R3's loopback confirm that the network has full connectivity.

Example 8-97. Configuring and Verifying BGP Redistribution into OSPFv3 on R2

Code View: Scroll / Show All

```
R2(config)#ipv6 router ospf 1
```

```
R2(config-rtr)#redistribute bgp 12
```

```
R2(config-rtr)#redistribute connected
```

```
R2(config-rtr)#exit
```

```
R2(config)#router bgp 12
```

```
R2(config-router)#address-family ipv6 unicast
```

```
R2(config-router-af)#bgp redistribute-internal
```

```
R2(config-router-af)#
```

```
R4(config-if)#do show ipv6 route ospf
```

IPv6 Routing Table – 11 entries

Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
OE2 12::/64 [110/20]
```

via FE80::2, FastEthernet0/0

```

OE2 13::/64 [110/1]
 via FE80::2, FastEthernet0/0
OE2 101::/64 [110/1]
 via FE80::2, FastEthernet0/0
OE2 102::/64 [110/20]
 via FE80::2, FastEthernet0/0
OE2 103::/64 [110/2]
 via FE80::2, FastEthernet0/0
R4(config-if)#end

R4#ping 103::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 103::1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/15/16 ms
R4#
R4#ping 103::1 source loopback 104
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 103::1, timeout is 2 seconds:
Packet sent with a source address of 104::1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/14/16 ms
R4#

```

### Transitioning IPv4 to IPv6

The successful market adoption of any new technology depends on its easy integration with the existing infrastructure without significant disruption of services. The Internet consists of hundreds of thousands of IPv4 networks and millions of IPv4 nodes. The challenge for IPv6 lies in making the integration of IPv4 and IPv6 nodes and the transition to IPv6 as transparent as possible to end users. From renumbering, to router upgrades, to protocol conversion, to client support, many different techniques are needed to ensure a smooth transition.

Fortunately, the transition from IPv4 to IPv6 does not require upgrades on all nodes at the same time; IPv4 and IPv6 will coexist for some time. Many RFCs relate to this transition, including those listed in Table 8-15.

Table 8-15. RFCs Relating to IPv4 to IPv6 Transition

| RFC Number | RFC Title                                                                |
|------------|--------------------------------------------------------------------------|
| RFC 2071   | Network Renumbering Overview: Why would I want it and what is it anyway? |
| RFC 2072   | <i>Router Renumbering Guide</i> (Updated by RFC 4192)                    |
| RFC 2185   | Routing Aspects of IPv6 Transition                                       |
| RFC 2529   | Transmission of IPv6 over IPv4 Domains Without Explicit Tunnels          |
| RFC 2767   | Dual Stack Hosts Using the Bump-In-the-Stack Technique (BIS)             |
| RFC 3056   | Connection of IPv6 Domains via IPv4 Clouds                               |
| RFC 3142   | An IPv6-to-IPv4 Transport Relay Translator                               |

| RFC Number | RFC Title                                                                     |
|------------|-------------------------------------------------------------------------------|
| RFC 4192   | Procedures for Renumbering an IPv6 Network Without a Flag Day                 |
| RFC 4213   | Basic Transition Mechanisms for IPv6 Hosts and Routers (Obsoletes RFC 2893)   |
| RFC 4380   | Teredo: Tunneling IPv6 over UDP Through Network Address Translations (NATs)   |
| RFC 4554   | Use of VLANs for IPv4-IPv6 Coexistence in Enterprise Networks                 |
| RFC 4779   | ISP IPv6 Deployment Scenarios in Broadband Access Networks                    |
| RFC 4798   | Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE) |
| RFC 4942   | IPv6 Transition/Co-existence Security Considerations                          |

A wide range of techniques are available for the period of transition between IPv4 and IPv6. These techniques can be grouped into the following three categories:

- **Dual-stack techniques**— Hosts and network devices run both IPv4 and IPv6 at the same time. This technique is useful as a temporary transition, but it adds overhead and uses many resources.
- **Tunneling techniques**— Isolated IPv6 networks are connected over an IPv4 infrastructure using tunnels. The edge devices are the only ones that need to be dual-stacked. Scalability may be an issue if many tunnels need to be created.
- **Translation techniques**— A translation device converts IPv6 packets into IPv4 packets and vice versa, allowing IPv6-only devices to communicate with IPv4-only devices. Scalability may again be an issue because of the resources required on the translator device.

The following sections provide a brief overview of each of these techniques. Later sections in this chapter detail some of the many tunneling and translation techniques.

#### Dual Stack

Dual stack is one of the primary technologies that makes the transition to IPv6 possible. It is an integration method in which a node has connectivity to both an IPv4 and IPv6 network; the node has two protocol stacks. The two stacks can be on the same interface or on multiple interfaces.

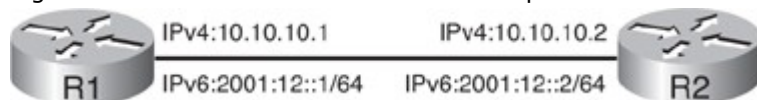
A dual-stack node chooses which stack to use based on the destination address; the node should prefer IPv6 when available. The dual-stack approach to IPv6 integration will be one of the most commonly used methods. Old IPv4-only applications will continue to work as before, while new and modified applications take advantage of both IP layers.

Application authors can add a new application programming interface (API) that supports both IPv4 and IPv6 addresses and Domain Name System (DNS) requests. Converted applications will be able to make use of both IPv4 and IPv6. An application can be converted to the new API while still using only IPv4.

Cisco IOS Software is IPv6 ready: As soon as IPv4 and IPv6 configurations are complete on an interface, the interface is dual stacked and it forwards both IPv4 and IPv6 traffic.

Consider the simple network in Figure 8-38. When R1 is configured with the commands shown in Example 8-98, it is dual stacked. Notice that the Fast Ethernet 0/0 interface has two addresses on it, an IPv4 and an IPv6 address. Also notice that for each protocol, the addresses on R1 and R2 are on the same network. The example also shows commands used to verify the configuration. The show ip interface command verifies the IPv4 configuration on the interface, and the show ipv6 interface command verifies the IPv6 configuration on the interface.

Figure 8-38. Network for Dual-Stack Example.



Example 8-98. Configuring and Verifying Dual Stack

Code View: Scroll / Show All  
R1(config)#**interface fa0/0**

```

R1(config-if)#ip address 10.10.10.1 255.255.255.0
R1(config-if)#ipv6 address 2001:12::1/64
R1(config-if)#^Z
R1#show ip interface fa0/0
FastEthernet0/0 is up, line protocol is up
 Internet address is 10.10.10.1/24
 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
 Outgoing access list is not set
 Inbound access list is not set
 Proxy ARP is enabled
 Local Proxy ARP is disabled
 Security level is default
 Split horizon is enabled
 ICMP redirects are always sent
 ICMP unreachable are always present
<output omitted>

R1#show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
 IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60
 Global unicast address(es):
 2001:12::1, subnet is 2001:12::/64
 Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF00:1
 FF02::1:FF2C:9F60
 MTU is 1500 bytes
 ICMP error messages limited to one every 100 milliseconds
 ICMP redirects are enabled
 ND DAD is enabled, number of DAD attempts: 1
 ND reachable time is 30000 milliseconds

```

A drawback of dual stacking is the resources required within each device configured with both protocols. The device must keep dual routing tables, routing protocol topology tables, and so forth, and must process each protocol independently. There is also a higher administrative overhead, and troubleshooting, monitoring, and so on is more complex.

#### Tunneling

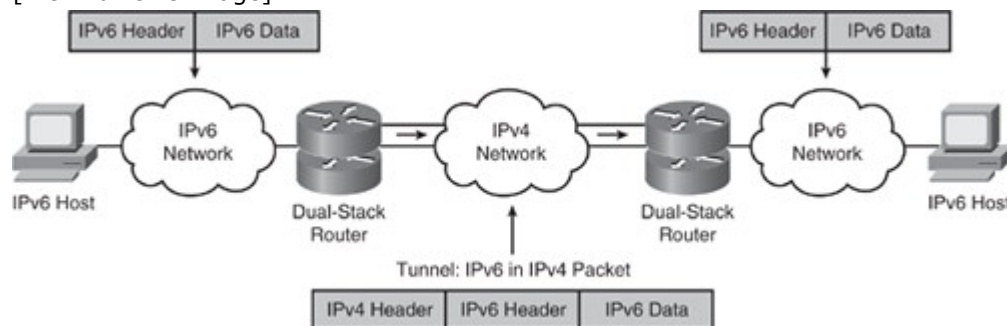
Tunnels are often used in networking to overlay incompatible functions over an existing network. For IPv6, tunneling is an integration method in which an IPv6 packet is encapsulated within IPv4.

When tunneling IPv6 traffic over an IPv4 network, an edge device (such as a router) encapsulates the IPv6 packet inside an IPv4 packet and the device at the other edge decapsulates it, and vice versa. This enables the connection of IPv6 islands without the need to convert the intermediary network to IPv6.

An example is illustrated in Figure 8-39. The routers involved in this example are dual stacking.

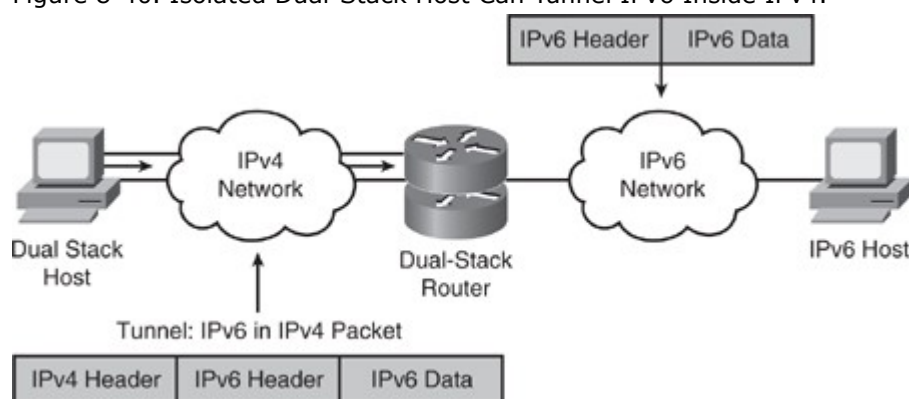
Figure 8-39. Tunneling IPv6 Inside IPv4 Packets.

[View full size image]



Tunneling can also be done between a host and a router, as shown in Figure 8-40, where an isolated dual-stack host uses an encapsulated tunnel to connect to the edge router of the IPv6 network.

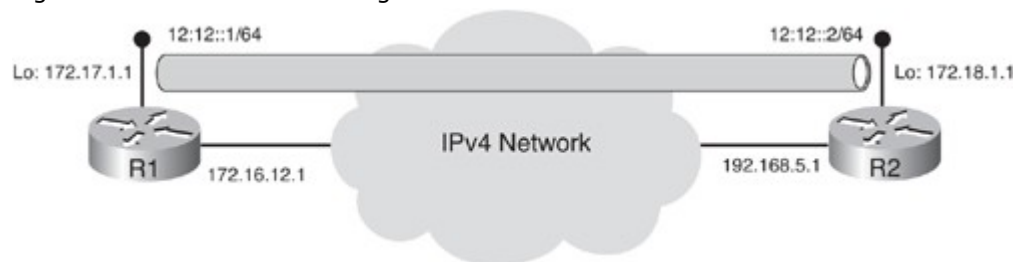
Figure 8-40. Isolated Dual-Stack Host Can Tunnel IPv6 Inside IPv4.



Tunnels can be either manually or automatically configured, depending on the scale required and administrative overhead tolerated.

Figure 8-41 illustrates another tunnel example, with addressing shown. Tunnels are typically created between loopback addresses on the edge routers, for stability. Notice that the physical interfaces connected to the IPv4 network have IPv4 addresses, as do the loopback interfaces. Also notice that the R1 and R2 addresses on the IPv4 network are on different subnets because they are not directly connected. However, the IPv6 addresses on the tunnel are on the same IPv6 network because the tunnel is a virtual point-to-point IPv6 connection between the two routers. The tunnel is an IPv6 link that is not concerned with the complexity of the IPv4 network over which it runs.

Figure 8-41. Tunnel Connecting IPv6 Networks over an IPv4 Network.



Some tunneling terminology can be explained using this example:

- IPv4 is the *transport protocol*, the protocol over which the tunnel is created.
- IPv6 is the *passenger protocol*, the protocol encapsulated in the tunnel and carried through the tunnel.
- Another protocol is used to create the tunnel, and is known as the *tunneling protocol*. It is also called the *encapsulation protocol* or the *carrier protocol*. An example of such a protocol is the Cisco generic routing encapsulation (GRE) protocol. It encapsulates the passenger protocol.



Tunneling is described further in the upcoming “Tunneling IPv6 Traffic” section.

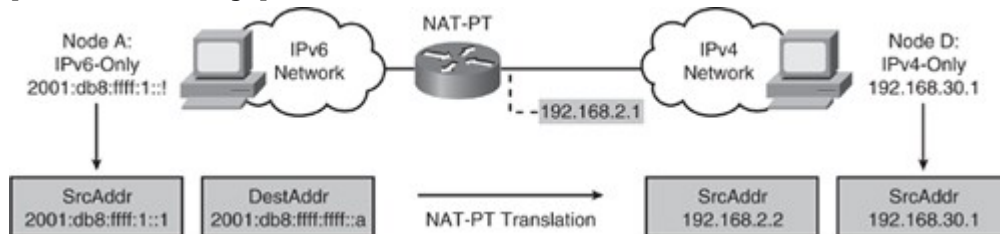
#### Translation

The dual-stack and tunneling techniques manage the interconnection of IPv6 domains. For legacy equipment that will not be upgraded to IPv6 and for some deployment scenarios, techniques are available for connecting IPv4-only nodes to IPv6-only nodes, using translation, an extension of NAT techniques.

As shown in Figure 8-42, NAT-PT is a translation mechanism that sits between an IPv6 network and an IPv4 network. The job of the translator (which, of course, can be a Cisco IOS router) is to translate IPv6 packets into IPv4 packets and vice versa. It is more than an address translator, it is really a protocol translator.

Figure 8-42. IPv4 - IPv6 Translation Mechanism.

[View full size image]



#### Note

RFC 4966, *Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status*, deprecates NAT-PT. However, this feature is still functional in the Cisco IOS.

The example in Figure 8-42 illustrates the translation of an IPv6 datagram sent from node A to node D. From the perspective of node A, it is establishing a communication to another IPv6 node. One advantage of NAT-PT is that no modifications are required on IPv6 node A; all it needs to know is the IPv6 address mapping of the IPv4 address of node D. This mapping can be obtained dynamically from the DNS server. IPv4 node D can also send a datagram to node A by using the IPv4 address mapped to the IPv6 address of node A. Again, from the perspective of node D, it is establishing IPv4 communication with node A. Node D does not require modification.

The NAT-PT device maintains a pool of globally routable IPv4 addresses that are assigned to IPv6 nodes dynamically as sessions are initiated across the IPv6/IPv4 boundary. The NAT-PT device also performs header translations, and translates any IP addresses in the payload of the packets. NAT-PT is therefore called a stateful technique.

Translation is described further in the upcoming “Translation Using NAT-PT” section.

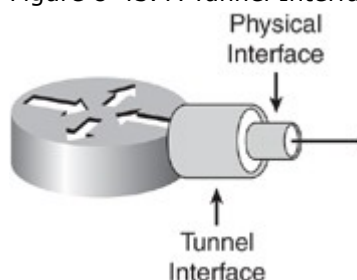
#### Tunneling IPv6 Traffic

This section details various techniques for tunneling IPv6 traffic. Manual tunnels are described first, followed by GRE tunnels. 6to4 tunnels, IPv4-compatible IPv6 tunnels, and Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) tunnels are also explored.

##### Manual IPv6 Tunnels

A manual tunnel simulates a permanent link between two IPv6 domains over an IPv4 backbone. Tunnel interfaces are created at each end, and IPv6 addresses are manually configured on each tunnel interface, as illustrated earlier in Figure 8-41. The tunnel interfaces do not have IPv4 addresses. The tunnel uses physical interfaces to carry the traffic, as illustrated in Figure 8-43. These physical interfaces have IPv4 addresses. A best practice is to use loopback interfaces as tunnel source and destination interfaces for stability, because loopback interfaces only go down if the router goes down. In this case, the loopback interfaces have IPv4 addresses and must be reachable across the IPv4 network. This practice mitigates the risk of tying the tunneling process to real physical interfaces, which could go down because of physical or data link layer problems. Alternatively, physical interfaces could be used as the tunnel source and destination interfaces.

Figure 8-43. A Tunnel Interface Runs over a Physical Interface.



The end routers implementing a manual tunnel must be dual stacked (because both IPv4 and IPv6 addresses must be configured on them). IPv4 routing must be set up properly to forward a packet between the two IPv6 networks. Note that the configuration of the tunnel will not change dynamically as network and routing needs change.

Manually tunneling IPv6 inside of IPv4 uses IPv4 protocol 41 and adds a 20-byte IPv4 header (if there are not any options in the header) before the IPv6 header and payload (data). Therefore, tunneling effectively decreases the MTU by 20 octets (or more if the IPv4 header contains any optional fields). Also note that tunneling will not work if an intermediary node between the two end points of the tunnel, such as a firewall, filters out IPv4 protocol 41, the IPv6 in IPv4 encapsulation protocol.

Manual tunnels are typically created between routers, but can be created between routers and hosts. Manual tunnels are used when a permanent connection is needed between two routers, between a host and router, or between remote IPv6 networks. The communication can be made secure with the use of cryptographic technologies such as IPsec, to provide confidentiality, integrity, and authentication services for the tunneled IPv6 traffic.

#### Manual IPv6 Tunnel Configuration and Verification Commands

The configuration and verification of manual tunnels in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

The **tunnel source** *interface-type interface-number* interface configuration command sets the source address for a tunnel interface as the address of the specified interface.

The **tunnel destination** *ip-address* interface configuration command specifies the destination address for a tunnel interface. In this case the *ip-address* is an IPv4 address.

The **tunnel mode ipv6ip** interface configuration command sets the encapsulation mode for the tunnel interface to use IPv6 as the passenger protocol, and IPv4 as both the encapsulation and transport protocol.

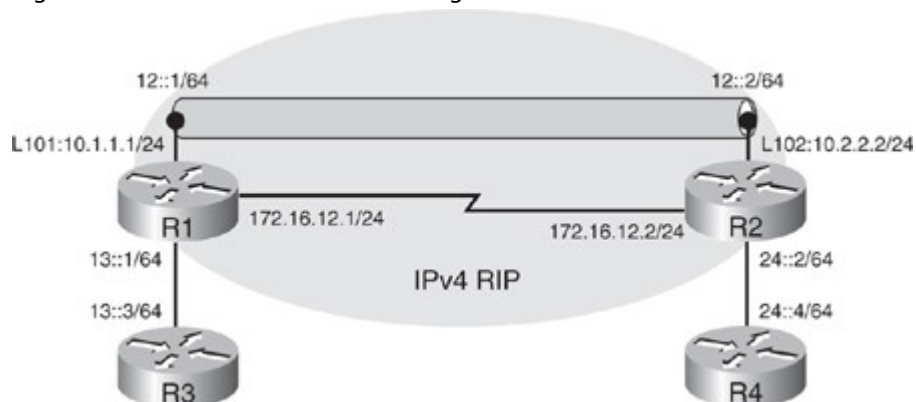
The **debug tunnel** EXEC command enables the display of a tunnel encapsulation and decapsulation process.

The **debug ip packet detail** EXEC command enables the display of details about IP packets traversing the router.

#### Manual IPv6 Tunnel Configuration and Verification Example

Figure 8-44 illustrates a network used to demonstrate the configuration and verification of a manual tunnel. There are two IPv6 networks, 13::/64 and 24::/64, separated by an IPv4-only network. (In this example the IPv4-only network is a single link, but it could be a more complex network.) IPv4 RIP is running between R1 and R2 to provide connectivity between the loopback interface IPv4 networks; Example 8-99 confirms this connectivity with the successful ping and a display of R1's IPv4 routing table.

Figure 8-44. Network Used for Configuration and Verification of a Manual Tunnel.



#### Example 8-99. Confirming IPv4 Connectivity Between R1 and R2

Code View: Scroll / Show All

**R1#ping 10.2.2.2 source 10.1.1.1**

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.2.2.2, timeout is 2 seconds:

Packet sent with a source address of 10.1.1.1

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 12/14/16 ms

R1#

**R1#show ip route**

Codes: 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

Gateway of last resort is not set

172.16.0.0/24 is subnetted, 1 subnets

C 172.16.12.0 is directly connected, Serial0/1/0

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks

C 10.1.1.0/24 is directly connected, Loopback101

R 10.0.0.0/8 [120/1] via 172.16.12.2, 00:00:05, Serial0/1/0

R1#

The objective of this example is to provide full connectivity between the IPv6 islands over the IPv4-only infrastructure. The first step is to create a manual tunnel between Routers R1 and R2, using the loopback interfaces on routers R1 and R2 as the tunnel source and destination. Example 8-100 illustrates the R1 and R2 configurations. On both routers, tunnel 12, representing the tunnel between routers R1 and R2, is created, although any interface number could have been chosen on each router. As soon as the tunnel interface is created, a message indicates that it is down, because it has not been completely defined yet. Because the tunnel does not require an IPv4 address, the no ip address command is used. The appropriate IPv6 address is configured. The appropriate loopback address is used as the tunnel source; its IPv4 address will be the source address for the tunnel. IPv4 is functioning here as the encapsulation protocol and as the transport protocol. The tunnel destination is the IPv4 address of the other router. This router looks up the destination address in its routing table and uses the associated outgoing interface as its physical interface for the tunnel. As soon as the tunnel has been created, the tunnel interface comes up. The tunnel mode command defines the encapsulation. In this case the tunnel mode ipv6ip command specifies a manual IPv6 tunnel with IPv6 as the passenger protocol, and IPv4 as both the encapsulation and transport protocol.

#### Example 8-100. Manual Tunnel Configuration on R1 and R2

Code View: Scroll / Show All

**R1(config)#interface tunnel 12**

R1(config-if)#

\*Aug 16 09:34:46.643: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down

R1(config-if)#**no ip address**

R1(config-if)#**ipv6 address 12::1/64**

R1(config-if)#**tunnel source loopback 101**

R1(config-if)#**tunnel destination 10.2.2.2**

```

R1(config-if)#
*Aug 16 09:36:52.051: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tun-
nel12, changed state to up
R1(config-if)#tunnel mode ?
aurp AURP TunnelTalk AppleTalk encapsulation
cayman Cayman TunnelTalk AppleTalk encapsulation
dvmrp DVMRP multicast tunnel
eon EON compatible CLNS tunnel
gre generic route encapsulation protocol
ipip IP over IP encapsulation
ipsec IPsec tunnel encapsulation
iptalk Apple IPTalk encapsulation
ipv6 Generic packet tunneling in IPv6
ipv6ip IP over IP encapsulation (KA9Q/NOS compatible)
rbscp RBSCP in IP tunnel
R1(config-if)#tunnel mode ipv6ip
R1(config-if)#

R2(config)#interface tunnel 12
R2(config-if)#
*Aug 16 09:38:47.532: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tun-
nel12, changed state to down
R2(config-if)#no ip address
R2(config-if)#ipv6 address 12::2/64
R2(config-if)#tunnel source loopback 102
R2(config-if)#tunnel destination 10.1.1.1
R2(config-if)#
*Aug 16 09:39:24.056: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tun-
nel12, changed state to up
R2(config-if)#tunnel mode ipv6ip
R2(config-if)#

```

To verify the tunnel operation, the debug tunnel command is enabled on R1, and one ping is sent to the R2 end of the tunnel. The output is provided in Example 8-101 and shows the tunnel interface encapsulating the outgoing IPv6 traffic and decapsulating the return traffic. The debug output also indicates that an additional 20 bytes are being added to the packet; these bytes are the IPv4 packet header. This example also shows the output from the debug ip packet detail command when the ping is repeated. This debug output shows the tunnel source and destination addresses, and the outgoing interface that the router selected for the tunnel (Serial 0/1/0 in this case), based on the tunnel destination address. The protocol of 41 is also displayed, indicating that IPv6 is encapsulated in the IPv4 packet.

Example 8-101. Verifying Manual Tunnel Operation

```

Code View: Scroll / Show All
R1#debug tunnel
Tunnel Interface debugging is on
R1#
R1#ping 12::2 repeat 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 12::2, timeout is 2 seconds:

```

```

!
Success rate is 100 percent (1/1), round-trip min/avg/max = 20/20/20 ms
R1#
*Aug 16 09:56:17.231: Tunnel12: IPv6/IP encapsulated 10.1.1.1->10.2.2.2
(linktype=79, len=120)
*Aug 16 09:56:17.231: Tunnel12 count tx, adding 20 encaps bytes
*Aug 16 09:56:17.247: Tunnel12: IPv6/IP to classify 10.2.2.2->10.1.1.1
(tbl=0, "default" len=120
ttl=254 tos=0x0)
*Aug 16 09:56:17.247: Tunnel12: IPv6/IP (PS) to decaps 10.2.2.2->10.1.1.1
(tbl=0, "default",
len=120,ttl=254)
*Aug 16 09:56:17.247: Tunnel12: decapsulated IPv6/IP packet (len 120)

R1#undebug all
All possible debugging has been turned off
R1#debug ip packet detail
IP packet debugging is on (detailed)
R1#
R1#ping 12::2 repeat 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 12::2, timeout is 2 seconds:
!
Success rate is 100 percent (1/1), round-trip min/avg/max = 16/16/16 ms
R1#
*Aug 16 09:57:04.627: IP: s=10.1.1.1 (Tunnel12), d=10.2.2.2 (Serial0/1/0), len
120, sending,
proto=41
*Aug 16 09:57:04.643: IP: tableid=0, s=10.2.2.2 (Serial0/1/0), d=10.1.1.1
(Loopback101), routed
via RIB
*Aug 16 09:57:04.643: IP: s=10.2.2.2 (Serial0/1/0), d=10.1.1.1, len 120, rcvd 4,
proto=41
*Aug 16 09:57:08.071: IP: s=172.16.12.2 (Serial0/1/0), d=224.0.0.9, len 52, rcvd 2
*Aug 16 09:57:08.071: UDP src=520, dst=520
*Aug 16 09:57:09.203: IP: s=10.1.1.1 (local), d=224.0.0.9 (Loopback101), len 72,
sending
broad/multicast
*Aug 16 09:57:09.203: UDP src=520, dst=520
*Aug 16 09:57:09.203: IP: s=10.1.1.1 (Loopback101), d=224.0.0.9, len 72, rcvd 2
*Aug 16 09:57:09.203: UDP src=520, dst=520

```

The show interface tunnel command also provides useful information; its output is shown in Example 8-102. The encapsulation, source address, and destination address can all be verified with this command. The tunnel mode is indicated in the "Tunnel protocol/transport IPv6/IP" line.

Example 8-102. Observing the Manual Tunnel

```

R1#show interface tunnel 12

```

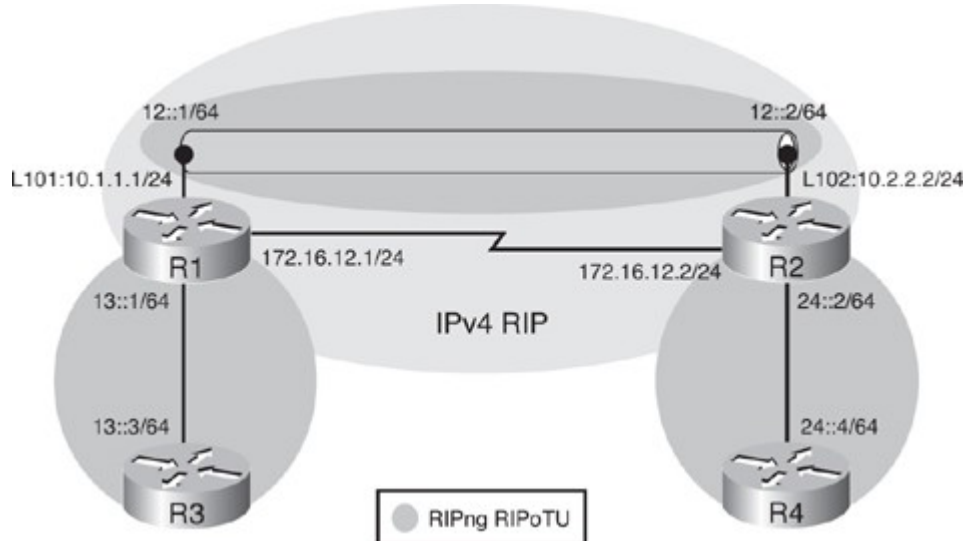
```

Tunnel12 is up, line protocol is up
Hardware is Tunnel
MTU 1514 bytes,BW 9 Kbit/sec, DLY 500000 usec,
 reliability 255/255, txload 1/255, rxload 1/255
Encapsulation TUNNEL, loopback not set
Keepalive not set
Tunnel source 10.1.1.1 (Loopback101), destination 10.2.2.2
Tunnel protocol/transport IPv6/IP
Tunnel TTL 255
Fast tunneling enabled
Tunnel transmit bandwidth 8000 (kbps)
Tunnel receive bandwidth 8000 (kbps)

```

To achieve full connectivity, RIPng is configured as shown in Figure 8-45. The RIPng process RIPv6 will be enabled between R3 and R1, between R2 and R4, and across the IPv6 tunnel between R1 and R2; the tunnel interface can participate in routing just like any other IPv6 link. Notice that RIPng will run across the tunnel, while IPv4 RIP is running across the physical interfaces to provide connectivity between the IPv4 addresses on the loopback interfaces.

Figure 8-45. Network Used for End-to-End Connectivity Through a Manual Tunnel.



Example 8-103 provides the RIPng configuration for all four routers. On R1 and R2, RIPng is enabled on the tunnel interface and on the Fast Ethernet interface. On R3 and R4, RIPng is only enabled on the Fast Ethernet interface.

Example 8-103. Configuring RIPng, Including Across the Manual Tunnel

```

R1(config)#ipv6 unicast-routing
R1(config)#interface tunnel 12
R1(config-if)#ipv6 rip RIPv6TU enable
R1(config-if)#interface fa0/0
R1(config-if)#ipv6 rip RIPv6TU enable
R1(config-if)#

```

```

R2(config)#ipv6 unicast-routing
R2(config)#interface tunnel 12
R2(config-if)#ipv6 rip RIPv6TU enable

```

```
R2(config-if)#interface fa0/0
R2(config-if)#ipv6 rip RIPv6 enable
R2(config-if)#
```

```
R3(config)#ipv6 unicast-routing
R3(config)#interface fa0/0
R3(config-if)#ipv6 rip RIPv6 enable
R3(config-if)#
```

```
R4(config)#ipv6 unicast-routing
R4(config)#interface fa0/0
R4(config-if)#ipv6 rip RIPv6 enable
R4(config-if)#^Z
R4#
```

The RIPv6 routes in R4's IPv6 routing table are displayed, to verify the RIPv6 configuration, as shown in Example 8-104. This output confirms that R4 is learning the 12::/64 and 13::/64 networks. The RIPv6 routes in R2's routing table are also displayed in this example. Notice that R2 is learning the 13::/64 network from R1 via the tunnel interface. To verify full connectivity across the tunnel, a ping from R3 to R4 is performed. As shown in the example, it is successful.

Example 8-104. Verifying RIPv6 Across the Manual Tunnel

Code View: Scroll / Show All

```
R4#show ipv6 route rip
```

IPv6 Routing Table – 6 entries

Codes: C – Connected, L – Local, S – Static, R – RIPv6, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 12::/64 [120/2]
 via FE80::2, FastEthernet0/0
R 13::/64 [120/3]
 via FE80::2, FastEthernet0/0
```

R4#

```
R2#show ipv6 route rip
```

IPv6 Routing Table – 7 entries

Codes: C – Connected, L – Local, S – Static, R – RIPv6, B – BGP

U – Per-user Static route

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary

O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2

ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2

```
R 13::/64 [120/2]
 via FE80::A01:101, Tunnel12
```

R2#



```
R3#ping 24::4
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms
```

```
R3#
```

This example demonstrates that manual tunnels are simple to configure, and are therefore useful for a small number of sites. However, for large networks manual tunnels are not scalable, from both a configuration and management perspective (for example, the number of interfaces to plan and configure) and from a resources-used perspective. The edge routers on which the tunnels terminate need to be dual stacked, and therefore must be capable of running both protocols and have the capacity to do so.

#### GRE IPv6 Tunnels

GRE IPv6 tunnels are very similar to manual tunnels, and their configuration is also very similar to manual tunnel configuration. GRE tunnels were developed by Cisco, and GRE encapsulation is the default tunneling protocol (configured with the **tunnel mode** command) on Cisco routers. GRE tunnels are more flexible in the protocols that they support. GRE can be deployed on top of multiple transport protocols and carry multiple passenger protocols.

Similar to the truck analogy used earlier for MBGP, in GRE tunnels

- The road is the transport protocol. For GRE tunnels, this can be IPv4 or IPv6.
- The truck is the tunneling (or encapsulation or carrier) protocol. This is the GRE encapsulation.
- The payload is the passenger protocol. For GRE tunnels this can be IPv6 or other protocols such as IS-IS.

In the upcoming "GRE IPv6 Tunnel Configuration and Verification Examples" section, two examples are configured: an IPv6 GRE tunnel over IPv4 as the transport protocol, and an IPv6 GRE tunnel over IPv6 as the transport protocol. An example use of this latter scenario is for IPv6 VPN connections.

Similar to manual tunnels, GRE tunnels are typically created between routers, but can also be created between routers and hosts. GRE tunnels are used when a permanent connection is needed between two routers, between a host and router, or between remote IPv6 networks. The communication can be made secure with the use of cryptographic technologies such as IPsec, to provide confidentiality, integrity, and authentication services for the tunneled IPv6 traffic. Note that GRE itself does not provide these security features; it is only an encapsulation protocol.

#### GRE IPv6 Tunnel Configuration and Verification Commands

The configuration and verification of manual tunnels in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

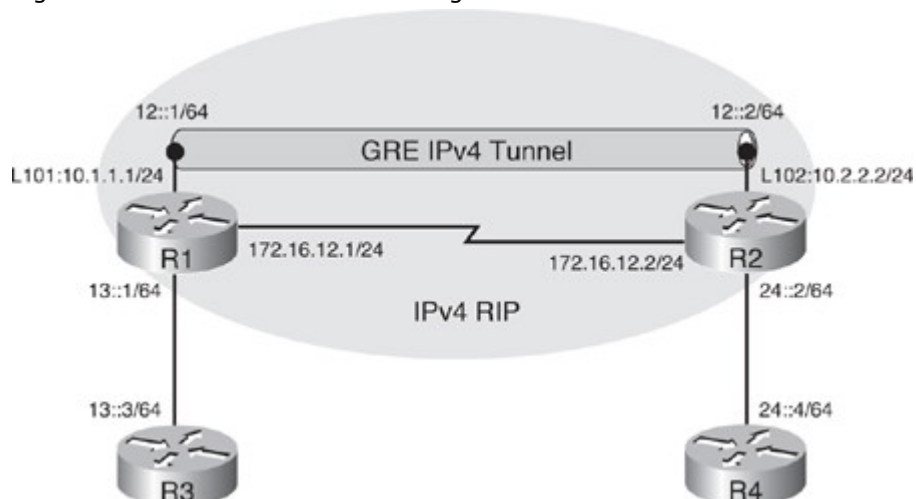
In this case, only two new commands are used:

- The **tunnel mode gre** interface configuration command sets the encapsulation mode for the tunnel interface to use GRE. This is the default mode, so this command is not required unless changing the mode from another mode.
- The **tunnel mode gre ipv6** interface configuration command sets the encapsulation mode for the tunnel interface to use GRE tunneling, using IPv6 as the transport protocol.

#### GRE IPv6 Tunnel Configuration and Verification Examples

Figure 8-46 illustrates a network used to demonstrate the configuration and verification of a GRE tunnel. This is the same network as in Figure 8-44, with the same addressing. The only difference is that now a GRE tunnel will be configured over the IPv4 network. Recall that there are two IPv6 networks, 13::/64 and 24::/64, separated by an IPv4-only network, and IPv4 RIP is running between R1 and R2 to provide connectivity between the loopback interface networks.

Figure 8-46. Network Used for Configuration and Verification of a GRE Tunnel.



The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure. The first step is to create a GRE tunnel between routers R1 and R2, using the loopback interfaces on routers R1 and R2 as the tunnel source and destination. Example 8-105 illustrates the R1 and R2 configurations. Notice that the configuration is identical to the manual tunnel configuration, with one exception: The tunnel mode command is not required because GRE is the default encapsulation (mode). As before, as soon as the tunnel has been created, the tunnel interface comes up.

Example 8-105. GRE Tunnel Configuration on R1 and R2

Code View: Scroll / Show All

**R1(config)#interface tunnel 12**

R1(config-if)#

%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down

R1(config-if)# **no ip address**

R1(config-if)# **ipv6 address 12::1/64**

R1(config-if)# **tunnel source loopback 101**

R1(config-if)# **tunnel destination 10.2.2.2**

R1(config-if)#

%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up

R1(config-if)#

**R2(config)#interface tunnel 12**

R2(config-if)#

%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down

R2(config-if)# **no ip address**

R2(config-if)# **ipv6 address 12::2/64**

R2(config-if)# **tunnel source loopback 102**

R2(config-if)# **tunnel destination 10.1.1.1**

R2(config-if)#

%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up

R2(config-if)#

The show interface tunnel command again provides useful information; its output is shown in Example 8-106. The encapsulation, source address, and destination address can all be verified with this command. The

tunnel mode is indicated in the "Tunnel protocol/transport GRE/IP" line, confirming that GRE is the default encapsulation.

#### Example 8-106. Observing the GRE Tunnel

```
Code View: Scroll / Show All
R2(config-if)#do show interface tunnel 12
Tunnel12 is up, line protocol is up
 Hardware is Tunnel
 MTU 1514 bytes,BW 9 Kbit/sec, DLY 500000 usec
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation TUNNEL, loopback not set
 Keepalive not set
 Tunnel source 10.2.2.2 (Loopback102), destination 10.1.1.1
 Tunnel protocol/transport GRE/IP
 Key disabled, sequencing disabled
 Checksumming of packets disabled
 Tunnel TTL 255
 Fast tunneling enabled
 Tunnel transmit bandwidth 8000 (kbps)
 Tunnel receive bandwidth 8000 (kbps)
 Last input never, output 00:00:11, 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: 0/0 (size/max)
 5 minute input rate 0 bits/sec, 0 packets/sec
 5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
 Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
—More—
```

To verify the tunnel operation, the debug ip packet detail command is enabled on R2, and R1's IPv6 address on the tunnel is pinged from R2. The output is provided in Example 8-107 and shows the tunnel source and destination addresses, and the outgoing interface that the router selected for the tunnel (Serial 0/1/0 in this case), based on the tunnel destination address. The protocol of 47 is also displayed, which is the protocol number for GRE.

#### Example 8-107. Verifying the GRE Tunnel Operation

```
Code View: Scroll / Show All
R2#debug ip packet detail
IP packet debugging is on (detailed)
R2#
IP: s=172.16.12.2 (local), d=224.0.0.9 (Serial0/1/0), len 52, sending broad/multicast
 UDP src=520, dst=520
IP: s=172.16.12.1 (Serial0/1/0), d=224.0.0.9, len 52, rcvd 2
 UDP src=520, dst=520
R2#ping 12::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12::1, timeout is 2 seconds:
!!!!
```

Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms

R2#

IP: s=10.2.2.2 (Tunnel12), d=10.1.1.1 (Serial0/1/0), len 124, sending, proto=47

IP: s=10.2.2.2 (Tunnel12), d=10.1.1.1 (Serial0/1/0), len 124, sending, proto=47

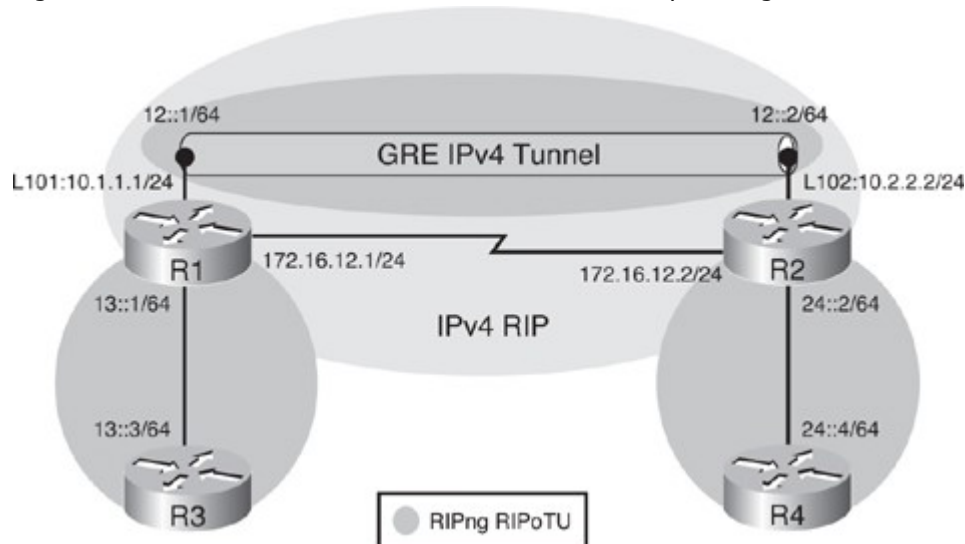
IP: s=10.2.2.2 (Tunnel12), d=10.1.1.1 (Serial0/1/0), len 124, sending, proto=47

IP: s=10.2.2.2 (Tunnel12), d=10.1.1.1 (Serial0/1/0), len 124, sending, proto=47

IP: s=10.2.2.2 (Tunnel12), d=10.1.1.1 (Serial0/1/0), len 124, sending, proto=47

To achieve full connectivity, RIPng is again configured as shown in Figure 8-47. The RIPng process RIPoTU will be enabled between R3 and R1, between R2 and R4, and across the IPv6 tunnel between R1 and R2. The tunnel interface can again participate in routing just like any other IPv6 link. Just as in the previous example, RIPng will run across the tunnel, while IPv4 RIP is running across the physical interfaces to provide connectivity between the IPv4 addresses on the loopback interfaces.

Figure 8-47. Network Used for End-to-End Connectivity Through a GRE Tunnel.



Example 8-108 provides the RIPng configuration for all four routers. This is the same configuration used for the manual tunnel example.

Example 8-108. Configuring RIPng, Including Across the GRE Tunnel

```
R1(config)#ipv6 unicast-routing
R1(config)#interface tunnel 12
R1(config-if)#ipv6 rip RIPoTU enable
R1(config-if)#interface fa0/0
R1(config-if)#ipv6 rip RIPoTU enable
R1(config-if)#
```

```
R2(config)#ipv6 unicast-routing
R2(config)#interface tunnel 12
R2(config-if)#ipv6 rip RIPoTU enable
R2(config-if)#interface fa0/0
R2(config-if)#ipv6 rip RIPoTU enable
R2(config-if)#
```

```
R3(config)#ipv6 unicast-routing
R3(config)#interface fa0/0
R3(config-if)#ipv6 rip RIPoTU enable
R3(config-if)#
```

```
R4(config)#ipv6 unicast-routing
R4(config)#interface fa0/0
R4(config-if)#ipv6 rip RIPoTU enable
R4(config-if)#
```

To verify full connectivity across the tunnel, a ping and a trace from R4 to R3 are performed. As shown in Example 8-109, they are successful. The trace confirms the path is via the IPv6 tunnel network 12::/64.

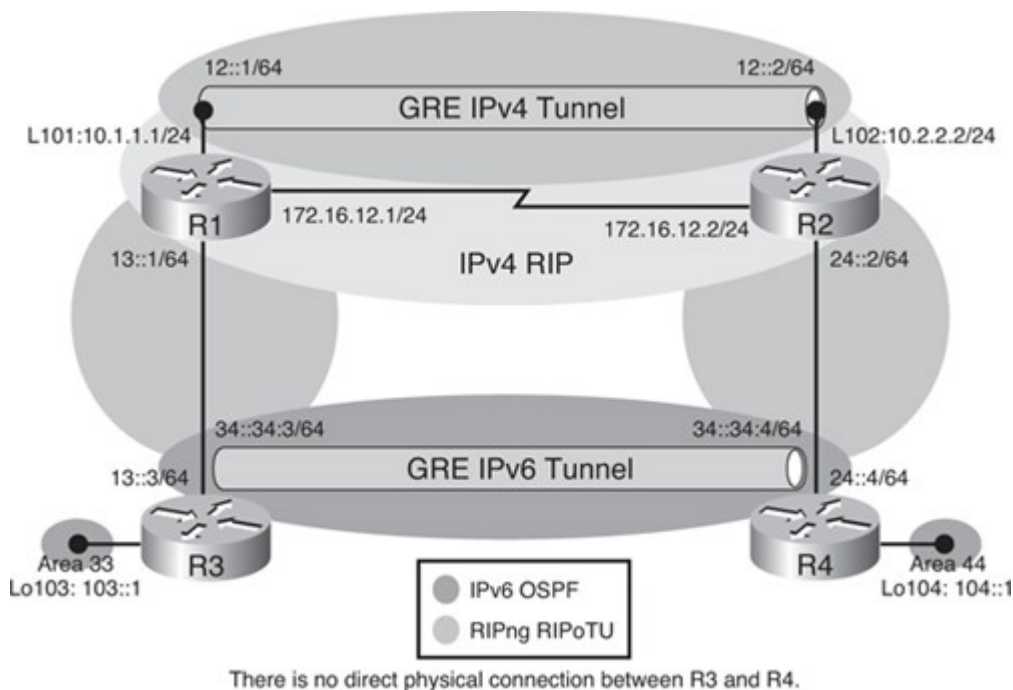
Example 8-109. Verifying RIPng Across the GRE Tunnel

```
R4#ping 13::3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 13::3, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms
R4#trace 13::3
Type escape sequence to abort.
Tracing the route to 13::3
 0 24::2 0 msec 0 msec 4 msec
 1 12::1 12 msec 16 msec 16 msec
 2 13::3 16 msec 16 msec 12 msec
R4#
```

To continue this example, another tunnel is configured, but this time it is an IPv6 GRE tunnel over IPv6. In other words, IPv6 is both the transport protocol and the passenger protocol; GRE is still the carrier (tunneling) protocol. In this case, IPv6 packets are encapsulated in IPv6 packets. This new tunnel is created between the physical interfaces on R3 and R4, as shown in Figure 8-48. As also shown in this figure, OSPFv3 is configured as the routing protocol over the tunnel, and the R3 and R4 (new) loopback interfaces are in separate OSPFv3 areas. It is important to notice that there is no direct physical connection between R3 and R4. The GRE IPv6 tunnel is configured between R3 and R4, but the physical path between these routers is still via R1 and R2 (and in the IPv6 world, this path includes the GRE IPv4 tunnel).

Figure 8-48. Network Used for End-to-End Connectivity Through a GRE IPv6 Tunnel.

[View full size image]



First, the tunnel is created between R3 and R4, as shown in Example 8-110. This configuration is very similar to the previous tunnel configuration. One difference is that instead of using the loopback interfaces as tunnel source and destination, the physical Fast Ethernet 0/0 interfaces are used (although using loopback interfaces is a best practice as discussed earlier, this was done to demonstrate that it can be done). Another difference is that the tunnel mode gre ipv6 command is added, indicating that the GRE tunnel is over IPv6 as the transport protocol.

Example 8-110. GRE IPv6 Tunnel Configuration on R3 and R4

Code View: Scroll / Show All

```
R3(config)#interface tunnel 34
R3(config-if)#no ip address
R3(config-if)#ipv6 address 34::34:3/64
R3(config-if)#tunnel source fa0/0
R3(config-if)#tunnel destination 24::4
R3(config-if)#tunnel mode gre ?
ip over IP
ipv6 over IPv6
multipoint over IP (multipoint)
R3(config-if)#tunnel mode gre ipv6
R3(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel34, changed state to up
R3(config-if)#

R4(config)#interface tunnel 34
R4(config-if)#no ip address
R4(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel34, changed state to down
R4(config-if)#ipv6 address 34::34:4/64
R4(config-if)#tunnel source fa0/0
R4(config-if)#tunnel destination 13::3
R4(config-if)#tunnel mode gre ipv6
```

```
R4(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel34, changed state to up
R4(config-if)#
```

The final step is to configure OSPFv3 on R3 and R4. This configuration is shown in Example 8-111. Notice that area 0 is between the routers on the tunnel interface, and the loopback interfaces are in different areas. (R3's loopback is in area 33 and R4's loopback is in area 44.) When the configuration is complete, the adjacency between the two routers goes to full state.

Example 8-111. Configuring OSPFv3 Across the GRE IPv6 Tunnel

```
R4(config)#ipv6 router ospf 1
R4(config-rtr)#router-id 4.4.4.4
R4(config-rtr)#exit
R4(config)#interface loopback 104
R4(config-if)#ipv6 ospf 1 area 44
R4(config-if)#exit
R4(config)#interface tunnel 34
R4(config-if)#ipv6 ospf 1 area 0
R4(config-if)#

R3(config)#ipv6 router ospf 1
R3(config-rtr)#router-id 3.3.3.3
R3(config-rtr)#exit
R3(config)#interface loopback 103
R3(config-if)#ipv6 ospf 1 area 33
R3(config-if)#exit
R3(config)#interface tunnel 34
R3(config-if)#ipv6 ospf 1 area 0
R3(config-if)#
%OSPFv3-5-ADJCHG: Process 1, Nbr 4.4.4.4 on Tunnel34 from LOADING to FULL,
Loading Done
R3(config-if)#
```

To verify end-to-end connectivity, a ping to R3's loopback interface address, sourced from R4's loopback interface, is performed, as shown in Example 8-112. The ping is successful.

Example 8-112. Configuring OSPFv3 Across the GRE IPv6 Tunnel

```
R4#ping 103::1 source loopback 104
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 103::1, timeout is 2 seconds:
Packet sent with a source address of 104::1
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/24/28 ms
R4#
```

## 6to4 Tunnels

6to4 tunnels, also known as 6-to-4 tunnels, are the first of three automatic tunneling methods explored in this chapter. 6to4 tunnels are again used to connect IPv6 domains over an IPv4 network, but they are point-to-multipoint, rather than the point-to-point tunnels discussed so far. The 6to4 tunnels are built automatically



by the edge routers, based on embedded IPv4 address within the IPv6 addresses of the tunnel interfaces on the edge routers.

Each 6to4 edge router, which is a dual-stacked device, has an IPv6 address with a /48 prefix, which is the concatenation of 2002::/16 and the hexadecimal representation of the IPv4 address of that edge router. 2002::/16 is a specially assigned address range for the purpose of 6to4 tunneling, as defined in RFC 3056, Connection of IPv6 Domains via IPv4 Clouds. The edge routers automatically build the tunnel using the IPv4 addresses that are embedded in the IPv6 addresses. For example, if the IPv4 address of an edge router is 192.168.99.1, the prefix of its IPv6 address is 2002:c0a8:6301::/48, because c0a86301 is the hexadecimal representation of 192.168.99.1. This calculation is shown in Figure 8-49.

Figure 8-49. IPv4 Address Example Decimal to Hexadecimal Calculation.

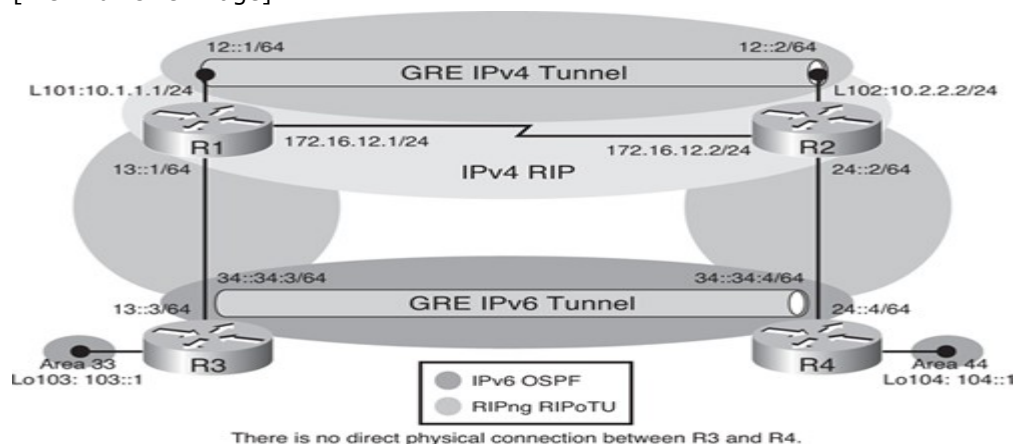
|             |          |          |            |          |
|-------------|----------|----------|------------|----------|
| Decimal     | 192      | 168      | 99         | 1        |
| Binary      | 11000000 | 10101000 | 0111000011 | 00000001 |
| Hexadecimal | C0       | A8       | 63         | 01       |

6to4 tunnels enable the fast deployment of IPv6 in a corporate network without the need for public IPv6 addresses from ISPs or registries.

Figure 8-50 illustrates an example. When the edge 6to4 Router A receives an IPv6 packet with a destination address in the range of 2002::/16 (the address 2002:c0a8:1e01::/48 in the example), it determines from its routing table that the packet must traverse the tunnel. The router extracts the IPv4 address embedded in the third to sixth octets, inclusively, in the IPv6 next-hop address. These octets are c0a8:1e01 in this example. In decimal the address is therefore 192.168.30.1. This IPv4 address is the IPv4 address of the 6to4 router at the destination site—the router at the other end of the tunnel, Router B in this figure. Router A encapsulates the IPv6 packet in an IPv4 packet with Router B's extracted IPv4 address as the destination address. The packet passes through the IPv4 network. The destination edge router, Router B, decapsulates the IPv6 packet from the received IPv4 packet and forwards the IPv6 packet to its final destination. The lower portion of Figure 8-50 illustrates the format of the IPv6 tunnel address.

Figure 8-50. Automatic 6to4 Tunnel Formation and Address Format.

[View full size image]



The biggest limitation of 6to4 tunnels is that only static routes or BGP can be used across them. This is because the other routing protocols use link-local addresses to form adjacencies and exchange updates. These link-local addresses do not conform to the address requirements for 6to4 tunnels (starting with 2002 and embedding the IPv4 address), so they cannot be used for 6to4 tunnels. Another limitation is that NAT cannot be used along the IPv4 path of the tunnel, again because of the 6to4 address requirements. Therefore, 6to4 tunnels are a good approach to use during migration to IPv6, but as a temporary rather than permanent solution. After migration, the IPv6 network will likely be renumbered, to remove the restrictive addressing scheme required by these tunnels.

#### 6to4 Tunnel Configuration and Verification Commands

The configuration and verification of 6to4 tunnels in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

In this case, only two new commands are used:

- The **tunnel mode ipv6ip 6to4** interface configuration command specifies IPv6 automatic tunneling mode using a 6to4 address.
- The **debug ipv6 packet detail** EXEC command enables the display of details about IPv6 packets traversing the router.

#### 6to4 Tunnel Configuration and Verification Example

Figure 8-51 illustrates a network used to demonstrate the configuration and verification of a 6to4 tunnel. This is the same network as in Figure 8-44 and Figure 8-46. The only difference in the addressing is the IPv4 loopback addresses on R1 and R2; they are now 172.16.101.1 and 172.16.102.1 respectively. In this example a 6to4 tunnel will be created over the IPv4 network. Recall that there are two IPv6 networks, 13::/64 and 24::/64, separated by an IPv4-only network, and IPv4 RIP is running between R1 and R2 to provide connectivity between the loopback interface networks. RIPv6 is running between R1 and R3, and between R2 and R4.

Figure 8-51. Network Used for Configuration and Verification of a 6to4 Tunnel.

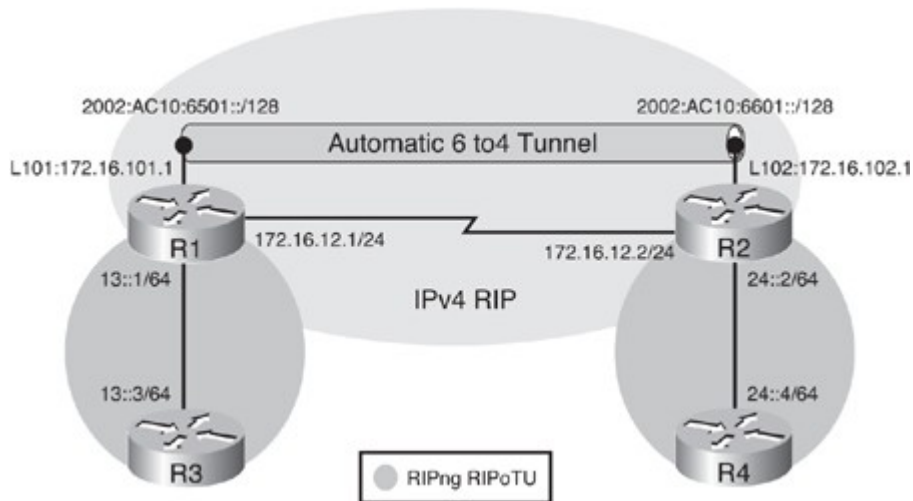


Figure 8-52 illustrates how to convert the IPv4 loopback addresses to hexadecimal, for use in the 6to4 tunnel addresses. The tunnel addresses, as shown in Figure 8-52, are the concatenation of 2002 with the converted IPv4 address. A /128 prefix length was chosen in this example network. These addresses will be configured as the IPv6 tunnel interface addresses. They embed the IPv4 addresses needed to establish the tunnel.

Figure 8-52. Conversion of IPv4 Loopback Addresses to Hexadecimal.

|             |          |          |          |          |
|-------------|----------|----------|----------|----------|
| Decimal     | 172      | 16       | 101      | 1        |
| Binary      | 10101100 | 00010000 | 01100101 | 00000001 |
| Hexadecimal | AC       | 10       | 65       | 01       |
|             |          |          |          |          |
| Decimal     | 172      | 16       | 172      | 1        |
| Binary      | 10101100 | 00010000 | 01100110 | 00000001 |
| Hexadecimal | AC       | 10       | 66       | 01       |

The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure. The first step is to configure routers R1 and R2 so that they can establish the 6to4 tunnel between them. Example 8-113 illustrates the R1 and R2 configurations. Notice that the configuration is similar to the manual and GRE tunnel configurations. One difference is that the tunnel destination is not specified, because the destination IPv4 address is embedded in the IPv6 address. Another difference is the tunnel mode ipv6ip 6to4 command is specified on each tunnel interface. As before, as soon as the tunnel has been created, the tunnel interface comes up.

#### Example 8-113. 6to4 Tunnel Configuration on R1 and R2

Code View: Scroll / Show All

```
R1(config)#interface tunnel 12
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R1(config-if)#no ip address
R1(config-if)#ipv6 address 2002:AC10:6501::/128
R1(config-if)#tunnel source loopback 101
R1(config-if)#tunnel mode ipv6ip 6to4
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R1(config-if)#
```

```
R2(config)#interface tunnel 12
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R2(config-if)#no ip address
R2(config-if)#ipv6 address 2002:AC10:6601::/128
R2(config-if)#tunnel source loopback 102
R2(config-if)#tunnel mode ipv6ip 6to4
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
```

To verify the tunnel operation, the debug ipv6 packet detail and debug tunnel commands are enabled on R2, and R1's IPv6 address on the tunnel is pinged from R2. The output is provided in Example 8-114 and shows that the route is not found. To investigate why this is so, R2's IPv6 routing table is also shown in the example. Notice that R2's own tunnel address, 2002:AC10:6601::/128 is in the routing table, but R1's address is not. This is because the addresses assigned to the each end of the tunnel are on different subnets (recall that a /128 prefix length was used).

#### Example 8-114. Verifying the 6to4 Tunnel Operation

Code View: Scroll / Show All

```
R2#debug ipv6 packet detail
IPv6 unicast packet debugging is on (detailed)
R2#debug tunnel
Tunnel interface debugging is on
R2#
R2#ping 2002:AC10:6501::
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:AC10:6501::, timeout is 2 seconds:
IPv6: SAS picked source 24::2 for 2002:AC10:6501:: (FastEthernet0/0)
IPv6: source 24::2 (local)
 dest 2002:AC10:6501::
 traffic class 0, flow 0x0, len, 100+0, prot 58, hops 64, Route not found.
IPv6: SAS picked source 24::2 for 2002:AC10:6501:: (FastEthernet0/0)
IPv6: source 24::2 (local)
 dest 2002:AC10:6501::
 traffic class 0, flow 0x0, len, 100+0, prot 58, hops 64, Route not found.
```

```
IPv6: SAS picked source 24::2 for 2002:AC10:6501:: (FastEthernet0/0)
IPv6: source 24::2 (local)
 dest 2002:AC10:6501::
 traffic class 0, flow 0x0, len 100+0, prot 58, hops 64, Route not found.
<output omitted>
```

R2#

**R2#show ipv6 route**

IPv6 Routing Table - 5 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP

U - Per-user Static route

I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary

O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2

ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2

C 24::/64 [0/0]

via ::, FastEthernet0/0

L 24::2/128 [0/0]

via ::, FastEthernet0/0

**LC 2002:AC10:6601::/128 [0/0]**

via ::, Tunnel12

L FE80::/10 [0/0]

via ::, Null0

L FF00::/8 [0/0]

via ::, Null0

R2#

To resolve this issue, a static route is configured on R2 to reach R1, and on R1 to reach R2. These configurations are shown in Example 8-115. Notice that in this case, because there is only one tunnel, the prefix length used on the static route is /16. This results in any packets with a 2002 prefix being accessible via the tunnel. The ping is tried again, and it is successful; the ping results and debug ipv6 packet detail and debug tunnel output are also shown in the example. This time the route is found, via the tunnel interfaces. The debug output also shows the IPv4 addresses used for tunnel creation, extracted from the IPv6 addresses.

Example 8-115. Configuring Static Routes Between the 6to4 Tunnel Endpoints

Code View: Scroll / Show All

**R2(config)#ipv6 route 2002::/16 tunnel 12**

R2(config)#

**R1(config)#ipv6 route 2002::/16 tunnel 12**

R1(config)#

**R2#debug ipv6 packet detail**

**R2#debug tunnel**

**R2#ping 2002:AC10:6501::**

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 2002:AC10:6501::, timeout is 2 seconds:

```

!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms
R2#
IPv6: SAS picked source 2002:AC10:6601:: for 2002:AC10:6501:: (Tunnel12)
IPv6: source 2002:AC10:6601:: (local)
 dest 2002:AC10:6501:: (Tunnel12)
 traffic class 0, flow 0x0, len 100+0, prot 58, hops 64, originating
IPv6: Sending on Tunnel12
Tunnel12 count tx, adding 20 encap bytes
Tunnel12: IPv6/IP to classify 172.16.101.1->172.16.102.1 (tbl=0, "default"
len=120 ttl=254
tos=0x0)
Tunnel12: IPv6/IP (PS) to decaps 172.16.101.1->172.16.102.1 (tbl=0, "default",
len=120,ttl=254)
Tunnel12: decapsulated IPv6/IP packet (len 120)
IPv6: source 2002:AC10:6501:: (Tunnel12)
 dest 2002:AC10:6601::

```

To reach destinations beyond the tunnel, more static routes must be added. Example 8-116 illustrates the use of the ipv6 route 24::/64 tunnel 12 command; however, as the example also shows, this does not work. The problem is that tunneling is not being triggered. R1 does not have enough information to encapsulate the IPv6 packets because it does not know the IPv4 address to use for the tunnel destination. The example shows the solution is to use the ipv6 route 24::/64 2002:AC10:6601:: command, which points to R2's tunnel address. Because this address has the IPv4 address embedded within it, the tunnel is created, and the ping works. The static routes in R1's routing table are also displayed in the example. Notice that R1 gets to the 24 network via 2002:AC10:6601::, which is R2's address. As the routing table shows, it gets to anything that starts with 2002 via the Tunnel 12 interface. Therefore, R1 can reach network 24 via R2, which it reaches via the tunnel.

#### Example 8-116. Configuring Static Routes Across a 6to4 Tunnel

```

Code View: Scroll / Show All
R1(config)#ipv6 route 24::/64 tunnel 12
R1(config)#do ping 24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
R1(config)#
R1(config)#no ipv6 route 24::/64 tunnel 12
R1(config)#ipv6 route 24::/64 2002:AC10:6601::
R1(config)#do ping 24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms
R1(config)#
R1(config)#do show ipv6 route static
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route

```

```

I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary
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 24::/64 [1/0]
 via 2002:AC10:6601::
S 2002::/16 [1/0]
 via ::, Tunnel12
R1(config)#

```

A static default route can also be configured, to route for all destinations. The configuration and confirmation ping are shown in Example 8-117. Recall that ::/0 is used for a default route.

Example 8-117. Configuring a Default Static Route Across a 6to4 Tunnel

```

R1(config)#no ipv6 route 24::/64 2002:AC10:6601::
R1(config)#ipv6 route ::/0 2002:AC10:6601::
R1(config)#do ping 24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms
R1(config)#

```

#### IPv4-Compatible IPv6 Tunnels

Although they are simpler to configure, IPv4-compatible IPv6 tunnels have in fact been deprecated. These tunnels are described here for completeness in case they are encountered in existing configurations.

IPv4-compatible IPv6 tunnels are very similar to 6to4 tunnels: they both are used to connect IPv6 domains over an IPv4 network, they both are point-to-multipoint tunnels, and both embed an IPv4 address within the IPv6 address so that the tunnel destination IPv4 address is easily obtained by the router and it can therefore automatically create the tunnel. The difference is how the IPv4 address is embedded. The IPv4-compatible IPv6 address is simply an IPv6 address with 96 bits filled with 0s followed by the 32-bit IPv4 address in the lower 32 bits; the 32-bit portion is even written in dotted-decimal format. For example, if the IPv4 address is 172.16.101.1, the IPv4-compatible IPv6 address is ::172.16.101.1.

The limitations of IPv4-compatible IPv6 tunnels are the same as those for 6to4 tunnels, for the same reasons, including that only static routes or BGP can be used across them, and the limitation on the IPv6 addressing plan.

#### IPv4-Compatible IPv6 Tunnel Configuration and Verification Commands

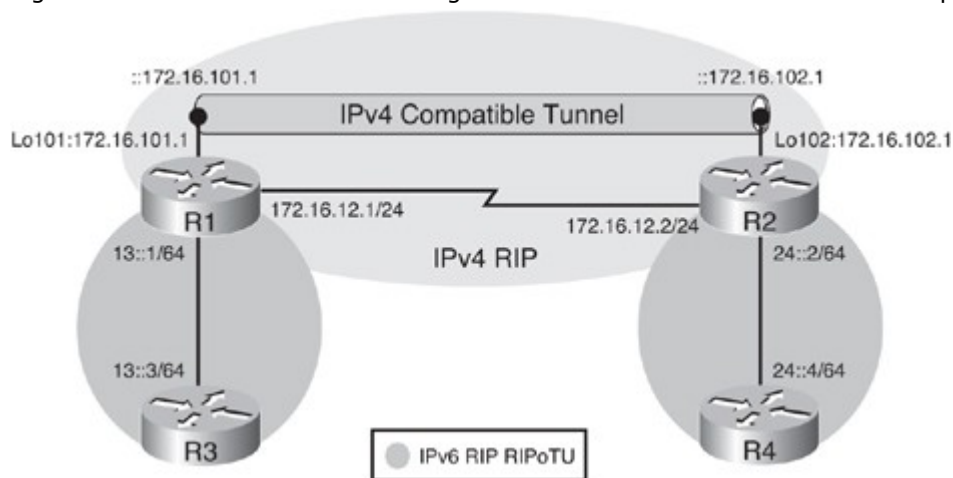
The configuration and verification of IPv4-compatible IPv6 tunnels in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

In this case, only one new command is used: the **tunnel mode ipv6ip auto-tunnel** interface configuration command specifies IPv6 automatic tunneling mode using an IPv4-compatible IPv6 address.

#### IPv4-Compatible IPv6 Tunnel Configuration and Verification Example

Figure 8-53 illustrates a network used to demonstrate the configuration and verification of an IPv4-compatible IPv6 tunnel. This is the same network in Figure 8-51, with the same IPv4 addresses on the loopback interfaces. Of course, the difference is that now an IPv4-compatible IPv6 tunnel will be created over the IPv4 network so the tunnel IPv6 addresses will be created automatically. Recall that there are two IPv6 networks, 13::/64 and 24::/64, separated by an IPv4-only network, and IPv4 RIP is running between R1 and R2 to provide connectivity between the loopback interface networks. RIPng is running between R1 and R3, and between R2 and R4.

Figure 8-53. Network Used for Configuration and Verification of an IPv4-Compatible IPv6 Tunnel.



The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure. The first step is to configure routers R1 and R2 so that they can establish the IPv4-compatible IPv6 tunnel between them. Example 8-118 illustrates the R1 and R2 configurations. Notice that the configuration is similar to the 6to4 configurations. Again, the tunnel destination is not specified, because the destination IPv4 address is embedded in the IPv6 address. One difference is that no IPv6 address is assigned to the tunnel interface this time. The IPv6 address of the tunnel is automatically created; it is the IPv4-compatible IPv6 address of the specified tunnel source interface. For example, on R1 the tunnel source is the loopback 101 interface, which has an IPv4 address 172.16.101.1. Therefore, the IPv6 address of R1's tunnel interface will be ::172.16.101.1. Another difference is that the tunnel mode `ipv6ip auto-tunnel` command is specified on each tunnel interface. As before, as soon as the tunnel has been created, the tunnel interface comes up.

Example 8-118. IPv4-Compatible IPv6 Tunnel Configuration on R1 and R2

```
Code View: Scroll / Show All
R1(config)#interface tunnel 12
R1(config-if)#no ip address
R1(config-if)#tunnel source loopback 101
R1(config-if)#tunnel mode ipv6ip auto-tunnel
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R1(config-if)#

R2(config)#interface tunnel 12
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R2(config-if)#no ip address
R2(config-if)#tunnel source loopback 102
R2(config-if)#tunnel mode ipv6ip auto-tunnel
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R2(config-if)#
```

To examine the tunnel interface status, the `show interface tunnel` command is used, as shown in Example 8-119. This output confirms that the encapsulation is set to tunnel, the tunnel source is the IPv4 address of the



loopback interface, and the tunnel protocol/transport is IPv6 auto-tunnel. Notice that the tunnel destination is unknown; it is determined when the tunnel is used.

#### Example 8-119. Observing IPv4-Compatible IPv6 Tunnel Interface

```
Code View: Scroll / Show All
R2(config-if)#do show interface tunnel 12
Tunnel12 is up, line protocol is up
 Hardware is Tunnel
 MTU 1514 bytes,BW 9 Kbit/sec, DLY 500000 usec
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation TUNNEL, loopback not set
 Keepalive not set
 Tunnel source 172.16.102.1 (Loopback102), destination UNKNOWN
 Tunnel protocol/transport IPv6 auto-tunnel

Fast tunneling enabled
Tunnel transmit bandwidth 8000 (kbps)
Tunnel receive bandwidth 8000 (kbps)
Last input never, output 00:00:15, 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: 0/0 (size/max)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
 Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
 0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
 4 packets output, 384 bytes, 0 underruns
—More—
```

To reach destinations beyond the tunnel, a static route must be added. Example 8-120 illustrates the use of the ipv6 route 24::/64 ::172.16.102.1 command on R1, which points to R2's tunnel IPv6 address as the way to get to the 24::/64 network between R2 and R4. Because this tunnel IPv6 address has the IPv4 address embedded within it, the tunnel is created, and a ping from R1 to R4 works, as also shown in the example. The ping is repeated after the debug tunnel command is entered. The debug output verifies that the destination IPv4 address of the tunnel is selected (from the IPv6 address in the static route). The example also shows the results of tracing to the same address; notice that R2's tunnel address is the first hop.

#### Example 8-120. Configuring a Static Route Across an IPv4-Compatible IPv6 Tunnel

```
Code View: Scroll / Show All
R1(config)#ipv6 route 24::/64 ::172.16.102.1
R1(config)#do ping 24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms
R1(config)#
R1(config)#do debug tunnel
Tunnel Interface debugging is on
R1(config)#do ping 24::4
```

```

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/19/20 ms
R1(config)#
Tunnel12 count tx, adding 20 encap bytes
Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"
len=120 ttl=254
tos=0x0)
Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",
len=120,ttl=254)
Tunnel12: decapsulated IPv6/IP packet (len 120)
Tunnel12 count tx, adding 20 encap bytes
<output omitted>

R1(config)#do undebug all
All possible debugging has been turned off
R1(config)#do trace 24::4
Type escape sequence to abort.
Tracing the route to 24::4
 1 ::172.16.102.1 12 msec 12 msec 16 msec
 2 24::4 16 msec 12 msec 16 msec
R1(config)#

```

#### Note

As mentioned, BGP is also supported across both 6to4 and IPv4-compatible IPv6 tunnels. For BGP, neighbor relationships would have to be configured to use the appropriate 6to4 or IPv4-compatible IPv6 address.

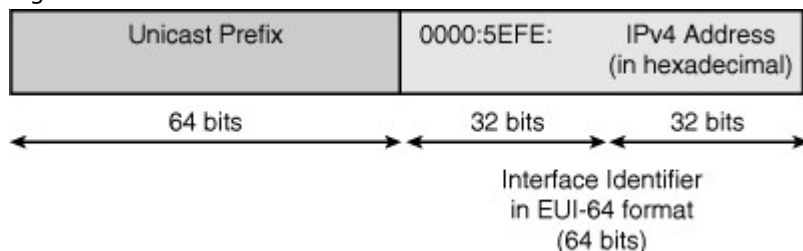
#### ISATAP Tunnels

ISATAP tunnels are very similar to 6to4 and IPv4-compatible IPv6 tunnels: They all are used to connect IPv6 domains over an IPv4 network, and all embed an IPv4 address within the IPv6 address so that the tunnel destination IPv4 address is easily obtained by the devices at the end of the tunnel and it can therefore automatically create the tunnel.

The goal of ISATAP is to provide connectivity for IPv6 hosts to a centralized IPv6-capable router, over an IPv4-only access network. ISATAP was designed to transport IPv6 packets within a site (hence the "intra-site" part of its name). It can still be used between sites, but its purpose is within sites.

ISATAP tunnels use IPv6 addresses in the format shown in Figure 8-54. A 64-bit prefix is concatenated to a 64-bit interface ID in EUI-64 format.

Figure 8-54. ISATAP IPv6 Address Format.



The 64-bit IPv6 prefix can be any valid unicast prefix, including a global routable prefix, a link-local prefix, or even a 6to4 prefix. The prefix should be selected according to the address plan for the network.

The upper 32 bits of the interface ID are 0000:5EFE, a reserved OUI value indicating an IPv6 ISATAP address. The lower (least significant) 32 bits of the interface ID contain the IPv4 address of the interface (written in hexadecimal). This embedded IPv4 address is used to create the tunnel, similar to other mechanisms. For example, consider the IPv4 address 172.16.101.1. From the earlier Figure 8-52, the hexadecimal equivalent of this address is AC10:6501. Therefore the 64-bit interface ID would be 0000:5EFE:AC10:6501.

As a transition strategy, ISATAP is an ideal, scalable solution for enterprise campus and branch sites because IPv6 connectivity can be automatically activated over an existing IPv4 network while the organization gradually migrates to an all-IPv6 network. Only the ISATAP router requires configuration. (Note that a best practice is to deploy more ISATAP routers for high availability.)

ISATAP is supported in popular operating systems including Windows XP, Vista, Windows 7, and Linux. An ISATAP tunnel is typically created from a host, such as a PC, to an ISATAP router. The tunnel is established using the IPv4 addresses of both the host and the router, across the IPv4 campus. The host establishes the tunnel only when it needs to. Hosts can discover the tunnel destination address (the IPv4 address of the router) dynamically through DNS or via a local statically configured definition (on the host itself). Because both IPv4 and tunneled IPv6 packets are transported over a single common IPv4 infrastructure, IPv4-dependent applications can continue to use IPv4 while newer IPv6-capable applications can be deployed immediately.

The main limitation of ISATAP is that it does not support IPv6 multicast. This is not an issue for static routing or BGP (which uses TCP unicasts to establish the peering sessions). However, because dynamic IGP's such as RIPng and OSPFv3 use multicast, they are not inherently supported in ISATAP. There is a workaround for OSPFv3, though: A list of neighbors can be specified, and unicast hello packets rather than multicast packets are sent to the neighbors in the list.

Another ISATAP issue is that it cannot operate across a NAT device. This might not be a significant issue because ISATAP is intended to be used inside an organization,

#### ISATAP Tunnel Configuration and Verification Commands

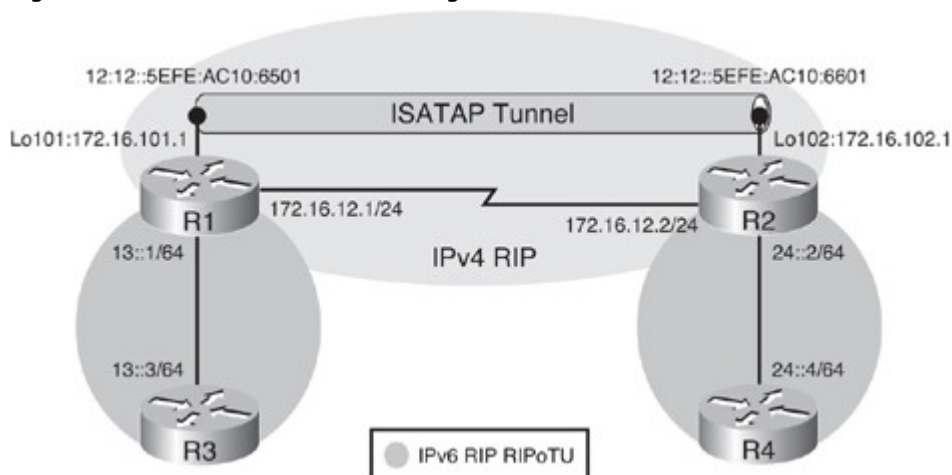
The configuration and verification of ISATAP tunnels in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

In this case, only one new command is used: The **tunnel mode ipv6ip isatap** interface configuration command specifies IPv6 automatic tunneling mode using an ISATAP address.

#### ISATAP Tunnel Configuration and Verification Example

Figure 8-55 illustrates a network used to demonstrate the configuration and verification of an ISATAP tunnel. Instead of a host-to-router scenario, this is a router-to-router scenario, using the same network as in Figure 8-53, with the same IPv4 addresses on the loopback interfaces. Of course, the difference is that now an ISATAP tunnel will be created over the IPv4 network, so the tunnel IPv6 addresses will be created automatically. Recall that there are two IPv6 networks, 13::/64 and 24::/64, separated by an IPv4-only network, and IPv4 RIP is running between R1 and R2 to provide connectivity between the loopback interface networks. RIPng is running between R1 and R3, and between R2 and R4.

Figure 8-55. Network Used for Configuration and Verification of an ISATAP Tunnel.



Notice that the IPv6 tunnel addresses, on network 12:12::/64, are derived from the loopback IPv4 addresses. R1's loopback address is 172.16.101.1. As discussed earlier, the hexadecimal equivalent of this address is AC10:6501, and therefore its 64-bit interface ID is 0000:5EFE:AC10:6501. Its IPv6 address is

12:12::0000:5EFE:AC10:6501, which can also be written as 12:12::5EFE:AC10:6501. Similarly, R2's loopback address is 172.16.102.1. The hexadecimal equivalent of this address is AC10:6601, and therefore its 64-bit interface ID is 0000:5EFE:AC10:6601. Its IPv6 address is 12:12::0000:5EFE:AC10:6601, which can also be written as 12:12::5EFE:AC10:6601.

The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure. The first step is to configure routers R1 and R2 so that they can establish the ISATAP tunnel between them. Example 8-121 illustrates the R1 and R2 configurations. Notice that the configuration is similar to the previous automatic tunnel configurations. Again, the tunnel destination is not specified, because the destination IPv4 address is embedded in the IPv6 address. This time an IPv6 address is assigned to the tunnel interface, but only the 64-bit prefix is specified. The eui-64 keyword is entered, so that the router selects the correct 64-bit interface ID. The tunnel mode ipv6ip isatap command is specified on each tunnel interface. It is at this point that the router calculates the IPv6 ISATAP address, using the specified 64-bit prefix, the ISATAP OUI, and the IPv4 address of the tunnel source (in hexadecimal format). As before, as soon as the tunnel has been created, the tunnel interface comes up.

Example 8-121. ISATAP Tunnel Configuration on R1 and R2

Code View: Scroll / Show All

```
R1(config)#interface tunnel 12
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R1(config-if)#no ip address
R1(config-if)#ipv6 address 12:12::/64 eui-64
R1(config-if)#tunnel source loopback 101
R1(config-if)#tunnel mode ipv6ip isatap
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R1(config-if)#

R2(config)#interface tunnel 12
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R2(config-if)#no ip address
R2(config-if)#ipv6 address 12:12::/64 eui-64
R2(config-if)#tunnel source loopback 102
R2(config-if)#tunnel mode ipv6ip isatap
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R2(config-if)#^Z
R2#
```

To examine the tunnel interface status, the show interface tunnel command is used, as shown in Example 8-122. This output confirms that the encapsulation is set to tunnel, the tunnel source is the IPv4 address of the loopback interface, and the tunnel protocol/transport is IPv6 ISATAP. Notice that the tunnel destination is unknown; it is determined when the tunnel is used. The show ipv6 interface tunnel command is used next in the example, to view the IPv6 properties of the interface. This output confirms that the router concatenated the prefix 12:12::/64 with the ISATAP OUI 0000:5EFE and the IPv4 address AC10:6501, to create the IPv6 address of the tunnel 12:12::5EFE:AC10:6501. Notice that the link-local address of the tunnel interface has the same 64-bit interface ID; it of course has a different prefix, the FE80 link-local prefix. Therefore, for ISATAP tunnels, the global unicast address and the link-local address are created on the interface.

Example 8-122. Observing ISATAP Tunnel Interface

Code View: Scroll / Show All

```

R1(config-if)#do show interface tunnel 12
Tunnel12 is up, line protocol is up
 Hardware is Tunnel
 MTU 1514 bytes,BW 9 Kbit/sec, DLY 500000 usec
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation TUNNEL, loopback not set
 Keepalive not set
 Tunnel source 172.16.101.1 (Loopback101), destination UNKNOWN
 Tunnel protocol/transport IPv6 ISATAP

Fast tunneling enabled
Tunnel transmit bandwidth 8000 (kbps)
Tunnel receive bandwidth 8000 (kbps)
Last input never, output 00:00:23, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output
R1(config-if)#do show ipv6 interface tunnel 12
Tunnel12 is up, line protocol is up
 IPv6 is enabled, link-local address is FE80::5EFE:AC10:6501
 Global unicast address(es):
 12::12::5EFE:AC10:6501, subnet is 12:12::/64 [EUI]
 Joined group address(es):
 FF02::1
 FF02::2
 FF02::1:FF10:6501
 MTU is 1480 bytes
 ICMP error messages limited to one every 100 milliseconds
 ICMP redirects are enabled
 ND DAD is not supported
 ND reachable time is 30000 milliseconds
 Hosts use stateless autoconfig for addresses.
R1(config-if)#

```

To verify connectivity, debugging is enabled and R2's tunnel IPv6 address is pinged from R1, as shown in Example 8-123. The ping is successful. The debug output confirms that the IPv4 addresses are being extracted from the ISATAP addresses to establish the tunnel across the network when IPv6 packets need to be sent. The ping is repeated, this time using R2's tunnel IPv6 link-local address; recall that the interface must be specified when a link-local address is used. The ping is successful. Unlike the previous tunneling mechanisms we discussed, ISATAP tunnels have predictable link-local addresses that are automatically created and use the tunnel mechanism. ISATAP is designed for traffic within a site, so that for hosts the tunnel appears as a connection to a normal Ethernet interface.

Example 8-123. Verifying ISATAP Tunnel Operation

```

Code View: Scroll / Show All
R1#debug tunnel
Tunnel Interface debugging is on
R1#
R1#ping 12:12::5EFE:AC10:6601
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12:12::5EFE:AC10:6601, timeout is 2 seconds:

```

```

!!!!
Success rate is 100 percent (5/5), round-trip min/avg = 16/17/20 ms
Tunnel12 count tx, adding 20 encap bytes
Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"
len=120 ttl=254
tos=0x0)
Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",
len=120,ttl=254)
Tunnel12: decapsulated IPv6/IP packet (len 120)
Tunnel12 count tx, adding 20 encap bytes
Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"
len=120 ttl=254
tos=0x0)
Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",
len=120,ttl=254)
Tunnel12: decapsulated IPv6/IP packet (len 120)
<output omitted>
R1#debug tunnel
Tunnel Interface debugging is on
R1#
R1#ping FE80::5EFE:AC10:6601
Output Interface: tunnel12
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::5EFE:AC10:6601, timeout is 2 seconds:
Packet sent with a source address of FE80::5EFE:AC10:6501
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/16/20 ms
R1#
Tunnel12 count tx, adding 20 encap bytes
Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"
len=120 ttl=254 tos=0x0)
Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",
len=120,
<output omitted>

```

To reach destinations beyond the tunnel, a static route must be added. Example 8-124 illustrates the use of a static route on R1, using R2's link-local address. Note that the outgoing interface must again be specified in the static route command when using a link-local address. This static route points to R2's link-local address as the way to get to the 24::/64 network between R2 and R4. Because this IPv6 address has the IPv4 address embedded within it, the tunnel is created, and a ping from R1 to R4 works, as also shown in the example. The debug output verifies that the destination IPv4 address of the tunnel is selected (from the IPv6 address in the static route).

Example 8-124. Configuring a Static Route Across an ISATAP Tunnel

```

Code View: Scroll / Show All
R1(config)#ipv6 route 24::/64 FE80::5EFE:AC10:6601
% Interface has to be specified for a link-local nexthop
R1(config)#ipv6 route 24::/64 tunnel12 FE80::5EFE:AC10:6601
R1(config)#do ping 24::4

```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:

!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms

R1(config)#

Tunnel12 count tx, adding 20 encap bytes

Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"

len=120 ttl=254

tos=0x0)

Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",

len=120,ttl=254)

Tunnel12: decapsulated IPv6/IP packet (len 120)

Tunnel12 count tx, adding 20 encap bytes

Tunnel12: IPv6/IP to classify 172.16.102.1->172.16.101.1 (tbl=0, "default"

len=120 ttl=254

tos=0x0)

Tunnel12: IPv6/IP (PS) to decaps 172.16.102.1->172.16.101.1 (tbl=0, "default",

len=120,ttl=254)

Tunnel12: decapsulated IPv6/IP packet (len 120)

### Translation Using NAT-PT

This section explores the use of both static and dynamic NAT-PT for translation between IPv6 and IPv4.

NAT-PT is another powerful transition technique, but is not a replacement for the other techniques, such as dual stack and tunneling, discussed so far in this chapter. Rather, it can be used in situations where direct communication between IPv6-only and IPv4-only networks is desired. It would not be appropriate in situations where connectivity between two IPv6 networks is required, because two points of translation would be necessary, which would not be efficient or effective.

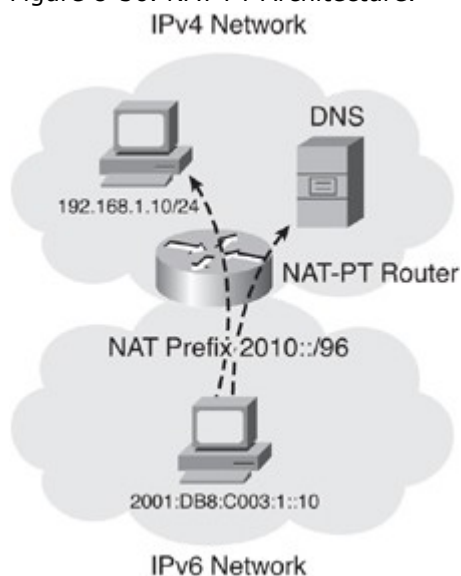
With NAT-PT, all configuration and translation is performed on the NAT-PT router. The other devices in the network are not aware of the existence of the other protocol's network, nor that translations are occurring.

The NAT-PT router translates source and destination addresses and other packet header fields in both directions: from the IPv4 network to the IPv6 network, and from the IPv6 network to the IPv4 network. Therefore, this router is dual stacked and must have two sets of translation entries for this bidirectional translation.

Figure 8-56 illustrates the NAT-PT architecture. DNS is crucial in real-life NAT-PT architectures, because applications initiate traffic from hosts, and DNS translates domain names to IP addresses. Because DNS requests may cross the NAT-PT router, a DNS application layer gateway (ALG) is typically implemented in NAT-PT routers to facilitate the name-to-address mapping. The DNS-ALG translates IPv6 addresses in DNS queries and responses into their IPv4 address bindings, and vice versa, as DNS packets traverse between IPv6 and IPv4 domains.



Figure 8-56. NAT-PT Architecture.



NAT-PT uses a 96-bit IPv6 network prefix to direct all IPv6 traffic that needs to be translated to the NAT-PT router. This prefix can be any routable prefix within the IPv6 domain. IPv6 routing must be configured such that all IPv6 packets addressed to this prefix are routed to the NAT-PT device. When the NAT-PT router receives an IPv6 packet destined for the NAT-PT prefix, it translates the packet according to the configured mapping rules. This prefix is also used in the translation of IPv4 address into IPv6 addresses.

Within the IPv6 domain, external IPv4 addresses are mapped to IPv6 addresses. This mapping is done statically (by means of predefined mapping between IPv4 and IPv6 addresses using the NAT-PT IPv6 prefix) or dynamically (by appending the IPv4 address to the NAT-PT IPv6 prefix). Similarly, static and dynamic mapping can be configured for translating internal IPv6 addresses to external IPv4 addresses.

Therefore, the NAT-PT router performs several bidirectional translations, including DNS, addressing, packet headers, and so forth.

The next section examines static NAT-PT. The following section examines dynamic NAT-PT.

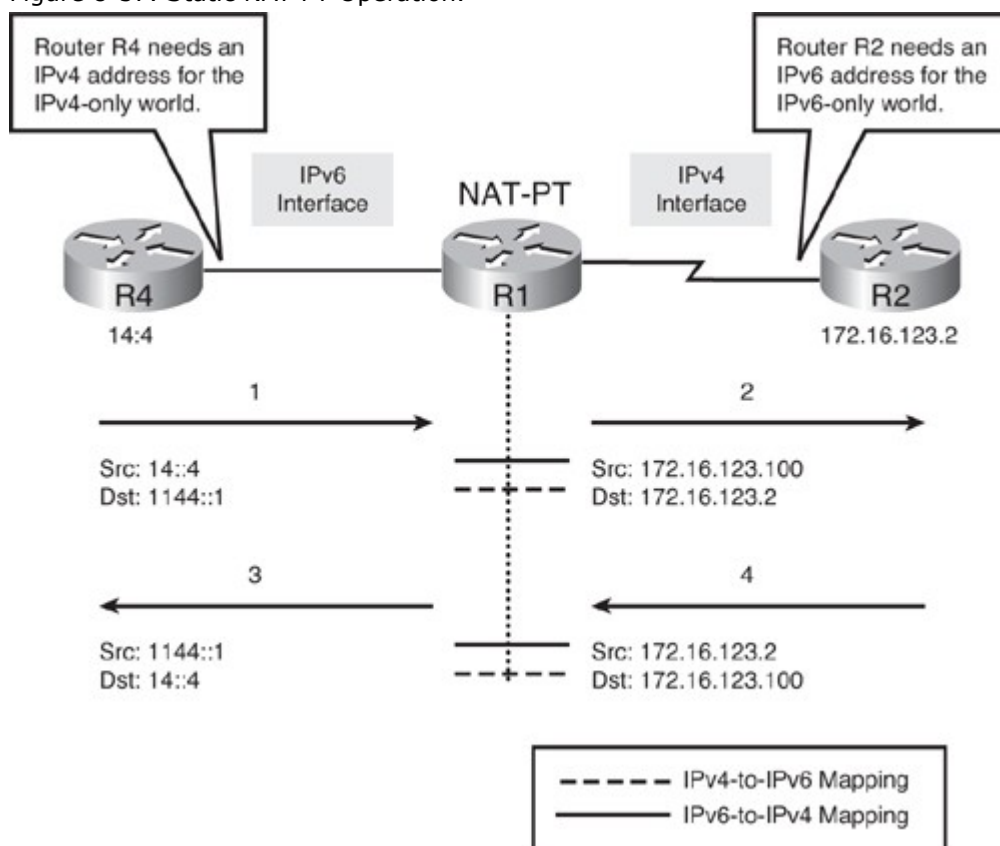
#### Static NAT-PT for IPv6

This section explores static NAT-PT.

#### Static NAT-PT Operation

Figure 8-57 illustrates how static NAT-PT operates. In this scenario, R4 and R2 need to communicate; R4 only has an IPv6 address and R2 only has an IPv4 address. Two static NAT-PT translations are configured on router R1 to allow bidirectional traffic between the two devices. Both the source and destination addresses in both directions will be translated.

Figure 8-57. Static NAT-PT Operation.



When R4 wants to communicate with R2, it sends an IPv6 packet (the only type it knows) with its own source address (14::4) and a destination address (1144::1) within the NAT-PT prefix. This prefix guides packets to the NAT-PT router, R1. The NAT-PT prefix is configured on R1 and typically advertised by R1 in an IGP such as RIPng or OSPFv3. The destination IPv6 address (1144::1) is the representation of the IPv4-only devices in the IPv6 world. When R1 receives the IPv6 packet from R4, it translates this destination IPv6 address to R2's IPv4 address. This is one of the static translation entries configured on R1; it is an IPv4-to-IPv6 mapping that gives the IPv4-only device R2 an address in the IPv6 world.

When R1 receives the IPv6 packet from R4, it also translates R4's IPv6 source address (14::4) to an IPv4 source address (172.16.123.100). This is the other static translation entry configured on R1; it is an IPv6-to-IPv4 mapping that gives the IPv6-only device R4 an address in the IPv4 world. The translated IPv4 address is typically within the destination IPv4 subnet; otherwise, it must be advertised to the IPv4 routers via static or dynamic routing.

When R2 replies to R4, traffic travels in the other direction and R1 translates the IPv4 source address (172.16.123.2) to IPv6 (1144::1) the IPv4 destination address (172.16.123.100) to IPv6 (14::4) by using the configured translation entries.

#### Static NAT-PT Configuration and Verification Commands

The configuration and verification of static NAT-PT in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

The **ipv6 nat** commands have similar syntax to the **ip nat** commands you are probably familiar with. However, as discussed, NAT-PT does more than translate addresses because it is translating between two different protocols.

The **ipv6 nat** interface configuration command designates that traffic originating from or destined for the interface is subject to NAT-PT.

The **ipv6 nat prefix** *ipv6-prefix/prefix-length* global or interface configuration command assigns an IPv6 prefix where matching IPv6 packets will be translated using NAT-PT. The *ipv6-prefix/prefix-length* specifies the IPv6 network used as the NAT-PT prefix; it is important to note that the prefix-length must be 96.

The **ipv6 nat v4v6 source** *ipv4-address ipv6-address* global configuration command configures IPv4-to-IPv6 static address translation using NAT-PT. The *ipv4-address ipv6-address* parameters specify a single static translation. The IPv4 address is translated to the IPv6 address.

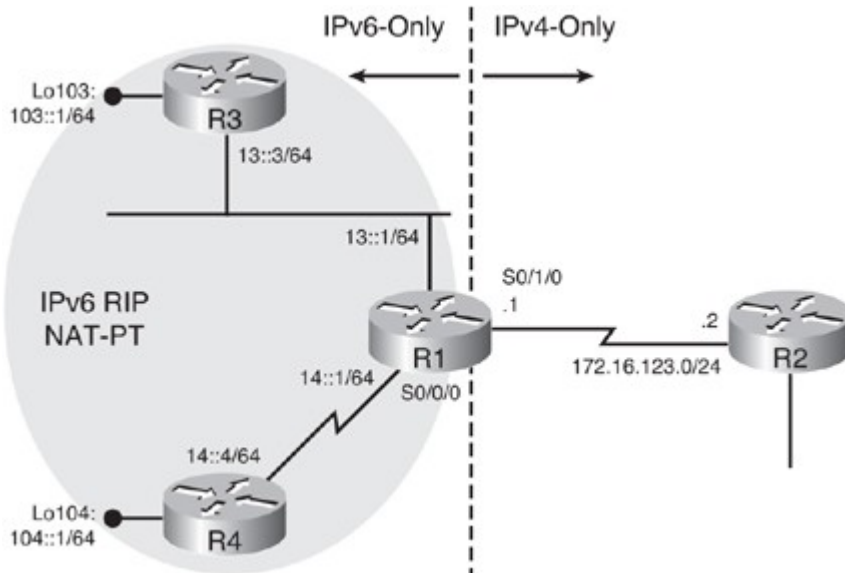
The **ipv6 nat v6v4 source** *ipv6-address ipv4-address* global configuration command configures IPv6-to-IPv4 static address translation using NAT-PT. The *ipv6-address ipv4-address* parameters specify a single static translation. The IPv6 address is translated to the IPv4 address.

The **show ipv6 nat translations** EXEC command displays active NAT-PT translations.

#### Static NAT-PT Configuration and Verification Example

Figure 8-58 illustrates a network used to demonstrate the configuration and verification of static NAT-PT. R3 and R4 are IPv6-only devices, and R2 is an IPv4-only device. R1 is the NAT-PT router. In this example, we require R4 and R2 to communicate. As described in the previous section, two static translation entries are required in R1 to allow this bidirectional communication.

Figure 8-58. Network Used for Configuration and Verification of Static NAT-PT.



All NAT-PT configuration is done on one device, the NAT-PT router, which is R1 in this example. The other routers are used for testing and verification, but one of the benefits of this transition technique is that only a single device is configured.

Example 8-125 illustrates the NAT-PT configuration on R1. First, NAT-PT is enabled on the two interfaces on which it is required, the interfaces pointing to R2 and R4, with the **ipv6 nat** command. Notice that, unlike IPv4 NAT configurations, no inside or outside keywords are required, because having an IPv6-only side and an IPv4-only side resolves the issue of boundaries.

Example 8-125. Configuring Static NAT-PT

Code View: Scroll / Show All

```
R1(config)#interface s0/0/0
```

```
R1(config-if)#ipv6 nat
```

```
R1(config-if)#
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface NV10, changed state to up
```

```
R1(config-if)#interface s0/1/0
```

```
R1(config-if)#ipv6 nat
```

```
R1(config-if)#exit
```

```
R1(config)#ipv6 nat v6v4 ?
```

```
pool Configure a IPv4 address pool
```

```
source Source Address Translation
```

```
R1(config)#ipv6 nat v6v4 source ?
```

```
X:X:X::X IPv6 address to translate
```

```
list Specify access list
```

```

route-map Configure a route-map
R1(config)#ipv6 nat v6v4 source 14::4 172.16.123.100
R1(config)#ipv6 nat v4v6 source 172.16.123.2 1144::1
R1(config)#exit
R1#
R1#debug ipv6 routing
 IPv6 routing table events debugging is on
R1#config t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#ipv6 nat prefix 1144::/64
% Invalid prefix length
R1(config)#ipv6 nat prefix 1144::/32
% Invalid prefix length
R1(config)#ipv6 nat prefix 1144::/96
R1(config)#
IPv6RT[Default]: connected, Route add 1144::/96 [new]
IPv6RT[Default]: connected, Add 1144::/96 to table
IPv6RT[Default]: connected, Adding next-hop :: over NV10 for 1144::/96, [0/0]
IPv6RT[Default]: Event: 1144::/96, Add, owner connected, previous None
R1(config)#

```

Next in Example 8-125, the `ipv6 nat v6v4 source` command is used to configure the mapping between R4's IPv6 source address (14::4) and the IPv4 address that R4 appears as in the IPv4 world (172.16.123.100). Notice that 172.16.123.100 is a valid address on the subnet between R1 and R2. It is an unused IP address on the destination subnet, so R1 does not need to advertise a new subnet to R2. Traffic coming from R4 will therefore look like it is coming from this R1-R2 subnet.

Example 8-125 also shows the `ipv6 nat v4v6 source` command, used to configure the mapping for return traffic—between R2's IPv4 source address (172.16.123.2) and the IPv6 address that R2 appears as in the IPv6 world (1144::1). This IPv6 address does not exist in the IPv6 world; it is an unused address selected to represent IPv4 devices in the IPv6 world. This IPv6 address is on the NAT-PT prefix, which is configured next. Before configuring the NAT-PT prefix in Example 8-125, the `debug ipv6 routing` command is entered so that the behavior of the router can be observed during the configuration. The `ipv6 nat prefix` command is entered, to define the NAT-PT prefix. Traffic destined to this prefix received on R1 will be translated. In this example, 1144::/64 is the NAT-PT prefix selected; it identifies all destinations on the IPv4-only network. As the example shows, you must configure a 96-bit prefix length. This is because 32-bit IPv4 addresses are translated into 128-bit IPv6 addresses; the difference is  $128 - 32 = 96$  bits, so this is the required number of bits in the prefix. Notice that this `ipv6 nat prefix` command creates a connected route in R1's routing table.

Example 8-126 displays the output of the `show ipv6 route connected` command, confirming that the NAT-PT 96-bit prefix is there. Notice that this prefix is directly connected to the interface NV10; NV10 is a NAT virtual interface and exists to allow NAT traffic flows. (You may have noticed earlier in Example 8-125 that this interface NV10 came up as soon as NAT-PT was enabled on the Serial 0/0/0 interface.) However, as also shown in the example, R4 does not yet know about this prefix (it is not in R4's IPv6 routing table) and therefore cannot reach it. So, on R1, the `redistribute connected` command (with a seed metric of 3) is entered under the RIPng process. R4 now has a route to the 1144 prefix and can forward traffic to it.

Example 8-126. Verification and Further Configuration of Static NAT-PT

```

Code View: Scroll / Show All
R1(config)#do show ipv6 route connected
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
 M - MIPv6, R - RIP, I1 - ISIS L1, I2 - ISIS L2
 IA - ISIS interarea, IS - ISIS summary, D - EIGRP, EX - EIGRP external

```

```

C 13::64 [0/0]
 via FastEthernet0/0, directly connected
C 14::/64 [0/0]
 via Serial0/0/0, directly connected
C 1144::/96 [0/0]
 via NV10, directly connected
R1(config)#

R4#show ipv6 route | inc 1144
R4#

R1(config)#ipv6 router rip NAT-PT
R1(config-rtr)#redistribute connected metric 3
R1(config-rtr)#

R4#show ipv6 route | inc 1144
R 1144::/96 [120/4]
R4#
R4#show ipv6 route rip
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R 13::/64 [120/2]
 via FE80::1, Serial 1/1.7
R 1144::/96 [120/4]
 via FE80::1, Serial 1/1.7
R4#

```

To test the configuration, a ping is sent from R4 to 1144::1, the IPv6 address representing R2. Example 8-127 shows that the ping is successful. The example also shows the IPv6 NAT table on R1, which includes the two static translation entries (172.16.123.100 to 14::4, and 172.16.123.2 to 1144::1) and the ICMPv6 entry created for the ping.

Note

In the output of the **show ipv6 nat translations** command, each translation is displayed over two lines.

Example 8-127. Further Verification of Static NAT-PT

```

R4#ping 1144::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1144::1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/73 ms
R4#

```

```
R1#show ipv6 nat translations
```

```
Prot IPv4 source IPv6 source
 IPv4 destination IPv6 destination
-- --
 172.16.123.2 1144::1
icmp 172.16.123.100,7364 14::4, 7364
 172.16.123.2, 7364 1144::1, 7364
-- 172.16.123.100 14::4
 -- --
R1#
```

As a final test, the debug ip icmp command is entered on R2, and the ping is reentered on R4. The results are shown in Example 8-128. In the R2 debug output, notice that the ping reply is being sent to the destination address 172.16.123.100, the IPv4 address representing R4 in the IPv4 world.

Example 8-128. Verification of Static NAT-PT on R2

```
R2#debug ip icmp
```

```
ICMP packet debugging is on
```

```
R2#
```

```
R4#ping 1144::1
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 1144::1, timeout is 2 seconds:
```

```
!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/73 ms
```

```
R4#
```

```
R2#
```

```
ICMP: echo reply sent, src 172.16.123.2, dst 172.16.123.100
```

```
ICMP: echo reply sent, src 172.16.123.2, dst 172.16.123.100
```

```
ICMP: echo reply sent, src 172.16.123.2, dst 172.16.123.100
```

```
ICMP: echo reply sent, src 172.16.123.2, dst 172.16.123.100
```

```
ICMP: echo reply sent, src 172.16.123.2, dst 172.16.123.100
```

This example illustrated that static NAT-PT is quite simple to configure, but it requires planning the translated addresses in both domains. Static NAT-PT is not scalable; it would be very cumbersome to create static entries for multiple sources communicating with multiple destinations. The next section discusses a more scalable solution, dynamic NAT-PT.

Dynamic NAT-PT for IPv6

With dynamic NAT-PT, addresses are allocated from an address pool, the same as is done with IPv4 dynamic NAT. And again, the commands have similar syntax to their IPv4 NAT counterparts.

With dynamic NAT-PT, the NAT-PT router receives, for example, a packet with an IPv6 destination address of an arbitrarily assigned 96-bit prefix (the NAT-PT prefix), the same as it did with static NAT-PT. This time though, instead of translating this to an IPv4 address that was statically configured, the NAT-PT router translates it to an IPv4 address from an address pool.

Dynamic NAT-PT Configuration and Verification Commands

The configuration and verification of dynamic NAT-PT in this section use many of the commands explored earlier in this chapter. Other commands used in the upcoming examples are noted in this section.

The `ipv6 nat v4v6 source {list {access-list-number | name} pool name}` global configuration command configures IPv4 to IPv6 dynamic address translation using NAT-PT. The parameters of this command are described in Table 8-16.

Table 8-16. *ipv6 nat v4v6 source* Command Parameters

| Parameter                                    | Description                                                                                                                                                              |
|----------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>list</b> <i>access-list-number   name</i> | Specifies a standard IPv4 access list. Packets with source IPv4 addresses that pass the access list are dynamically translated using IPv6 addresses from the named pool. |
| <b>pool</b> <i>name</i>                      | Specifies the name of the pool from which IPv6 addresses are dynamically allocated.                                                                                      |

The `ipv6 nat v4v6 pool name start-ipv6 end-ipv6 prefix-length prefix-length` defines a pool of IPv6 addresses for NAT-PT. The parameters of this command are described in Table 8-17.

Table 8-17. *ipv6 nat v4v6 pool* Command Parameters

| Parameter                                 | Description                                                                                                                               |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <i>name</i>                               | Specifies the name of the pool.                                                                                                           |
| <i>start-ipv6 end-ipv6</i>                | Specifies the starting and ending IPv6 addresses that define the range of IPv6 addresses in the address pool.                             |
| <b>prefix-length</b> <i>prefix-length</i> | Specifies the number of bits of the address that are network or subnet bits, indicating the subnet to which the pool of addresses belong. |

The `ipv6 nat v6v4 source {list {access-list-name} pool name}` global configuration command configures IPv6-to-IPv4 dynamic address translation using NAT-PT. The parameters of this command are described in Table 8-18.

Table 8-18. *ipv6 nat v6v4 source* Command Parameters

| Parameter                           | Description                                                                                                                                                      |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>list</b> <i>access-list-name</i> | Specifies an IPv6 access list. Packets with source IPv6 addresses that pass the access list are dynamically translated using IPv4 addresses from the named pool. |
| <b>pool</b> <i>name</i>             | Specifies the name of the pool from which IPv4 addresses are dynamically allocated.                                                                              |

The `ipv6 nat v6v4 pool name start-ipv4 end-ipv4 prefix-length prefix-length` defines a pool of IPv4 addresses for NAT-PT. The parameters of this command are described in Table 8-19.

Table 8-19. *ipv6 nat v6v4 pool* Command Parameters

| Parameter                                 | Description                                                                                                                               |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <i>name</i>                               | Specifies the name of the pool.                                                                                                           |
| <i>start-ipv4 end-ipv4</i>                | Specifies the starting and ending IPv4 addresses that define the range of IPv4 addresses in the address pool.                             |
| <b>prefix-length</b> <i>prefix-length</i> | Specifies the number of bits of the address that are network or subnet bits, indicating the subnet to which the pool of addresses belong. |



The **service finger** global configuration command configures a system to accept Finger protocol requests. The Finger service allows remote users to view the output equivalent to the **show users** command. When this command is configured on a router, the router responds to a **telnet addressfinger** command from a remote host by immediately displaying the output of the **show users** command and then closing the connection.

#### Note

The **service finger** command has been replaced by the **ip finger** command. The **service finger** command continues to function in current releases, but may be removed in a future release.

The **show ipv6 nat statistics** EXEC command displays NAT-PT statistics.

The **debug ipv6 nat** EXEC command displays debug messages for NAT-PT translation events.

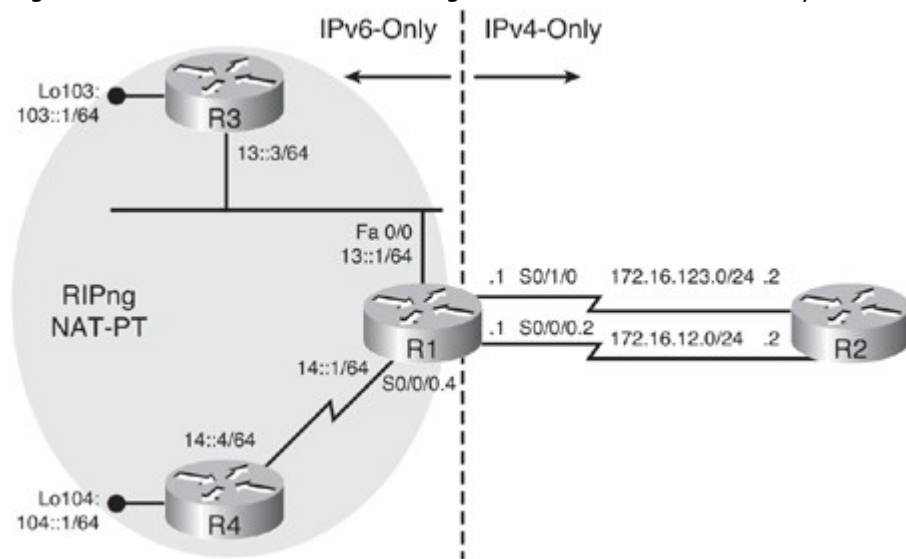
#### Dynamic NAT-PT Configuration and Verification Examples

In this section two examples are presented. The first is for IPv6-to-IPv4 traffic flows, and the second is for IPv4-to-IPv6 traffic flows.

##### IPv6-to-IPv4 Dynamic NAT-PT Configuration and Verification Example

Figure 8-59 illustrates a network used to demonstrate the configuration and verification of IPv6-to-IPv4 traffic flows with dynamic NAT-PT. This network is similar to the one used in the static NAT-PT example, with an additional connection between R1 and R2. R3 and R4 are IPv6-only devices, and R2 is an IPv4-only device. R1 is the NAT-PT router. The loopback interfaces on R3 and R4 are used in this example.

Figure 8-59. Network Used for Configuration and Verification of Dynamic NAT-PT.



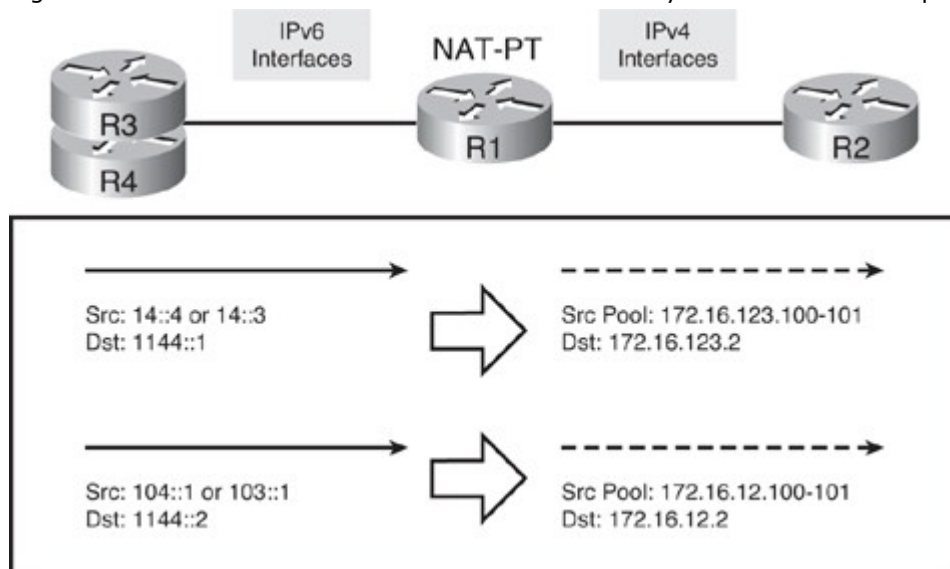
In this example, we want to dynamically translate traffic coming from the R3 and R4 loopback interface addresses to a pool of IPv4 addresses, and to dynamically translate traffic sourced from the R3 and R4 physical interfaces to be dynamically translated to a different pool of IPv4 addresses. R2's two IPv4 interface addresses are destination addresses in this scenario and will be represented by (translated to) two different IPv6 addresses. This scenario simulates multiple IPv6 host devices being translated to different address pools as they communicate with multiple IPv4 hosts.

#### Note

In this example, the translations of the IPv6 destination addresses (R2's addresses) use static translation. In a real-life scenario, DNS ALG functionality would supply the IPv6 destination addresses.

Figure 8-60 displays the translations that will occur in this example. For traffic sourced from the R3 and R4 physical interfaces (13::3 and 14::4), a pool on the 172.16.123.0 subnet is used, specifically the 172.16.123.100 and 172.16.123.101 addresses. For traffic sourced from the R3 and R4 loopback interfaces (103::1 and 104::1), a pool on the 172.16.12.0 subnet is used, specifically the 172.16.12.100 and 172.16.12.101 addresses. The destination address 1144::1 represents R2's 172.16.123.2 interface in the IPv6 world, and the destination address 1144::2 represents R2's 172.16.12.2 interface in the IPv6 world.

Figure 8-60. Address Translations Performed in First Dynamic NAT-PT Example.



Example 8-129 illustrates the NAT-PT configuration on R1. First, NAT-PT is enabled on the four interfaces on which it is required, the interfaces pointing to R2, R3, and R4, with the `ipv6 nat` command. The example also shows the `ipv6 nat v4v6 source` commands, used to configure the static mappings for R2's IPv4 source addresses (172.16.12.2 and 172.16.123.2) and the IPv6 addresses that R2 appears as in the IPv6 world (1144::1 and 1144::2). (These IPv6 address do not exist in the IPv6 world; they are on the NAT-PT prefix, which is configured near the end of Example 8-129.)

Example 8-129. Configuring Dynamic NAT-PT

Code View: Scroll / Show All

```
R1(config)#interface FastEthernet0/0
R1(config-if)#ipv6 nat
R1(config-if)#interface Serial0/0/0.2
R1(config-subif)#ipv6 nat
R1(config-subif)#interface Serial0/0/0.4
R1(config-subif)#ipv6 nat
R1(config-subif)#interface Serial0/1/0
R1(config-if)#ipv6 nat
R1(config-if)#ipv6 nat v4v6 source 172.16.12.2 1144::2
R1(config)#ipv6 nat v4v6 source 172.16.123.2 1144::1
R1(config)#
R1(config)#ipv6 nat v6v4 source list LOOPBACK pool POOL_12
R1(config)#ipv6 nat v6v4 source list PHYSICAL pool POOL_123
R1(config)#
R1(config)#ipv6 nat v6v4 pool POOL_12 172.16.12.100 172.16.12.101 prefix-length 24
R1(config)#ipv6 nat v6v4 pool POOL_123 172.16.123.100 172.16.123.101 prefix-length 24
R1(config)#
R1(config)#ipv6 access-list LOOPBACK
R1(config-ipv6-acl)#permit ipv6 104::/64 any
R1(config-ipv6-acl)#permit ipv6 103::/64 any
R1(config-ipv6-acl)#ipv6 access-list PHYSICAL
R1(config-ipv6-acl)#permit ipv6 13::/64 any
R1(config-ipv6-acl)#permit ipv6 14::/64 any
```

```

R1(config-ipv6-acl)#exit
R1(config)#
R1(config)#ipv6 nat prefix 1144::/96
R1(config)#ipv6 router rip NAT-PT
R1(config-rtr)#redistribute connected metric 1
R1(config-rtr)#exit
R1(config)#

```

Next in Example 8-129, the `ipv6 nat v6v4` source commands are used to configure the dynamic mapping between the IPv6 addresses on R3 and R4 and the IPv4 addresses that R3 and R4 appear as in the IPv4 world. The first command uses the IPv6 access list `LOOPBACK` to identify the R3 and R4 loopback interface IPv6 addresses, and an address pool `POOL_12` to define the IPv4 translated addresses. The second command uses an IPv6 access list called `PHYSICAL` to identify the R3 and R4 physical interface IPv6 addresses, and an address pool `POOL_123` to define the IPv4 translated addresses in a different pool.

Next in Example 8-129 the IPv4 address pools are created. `POOL_12` defines the addresses on the 172.16.12.0 subnet, and `POOL_123` defines the addresses on the 172.16.123.0 subnet. The addresses chosen to be the translated addresses must, of course, be unused addresses in the IPv4 network.

Next in Example 8-129, the IPv6 access lists referenced in the earlier statements are created. The `LOOPBACK` access list permits the R3 and R4 loopback interface addresses, and the `PHYSICAL` access list permits the R3 and R4 physical interface addresses.

Finally in Example 8-129, the IPv6 NAT prefix is defined, and connected routes (including this prefix) are redistributed into RIPng (with a seed metric of 1).

Before conducting the verification tests, the `service finger` command is entered on R2, as shown in Example 8-130. (When this command is configured, and another device uses the `telnet` address `finger` command to connect to R2, R2 displays the output of the `show users` command and then closes the connection. The output displays the source IP address of the connecting device, which confirms that the NAT-PT configuration is working correctly.) On R1, the `debug ipv6 nat` command is entered, and then four Telnet tests are conducted from each of R3 and R4: from each router R3 and R4, a Telnet to each of R2's addresses is performed, from both the router's physical and loopback interfaces.

Example 8-130 includes the first 2 tests, and the resulting debug output on R1. The first Telnet is from R4's physical interface (the default). The output confirms that this Telnet has a source address 172.16.123.101 in the IPv4 world. The second Telnet is from R4's loopback interface. The output confirms that this Telnet has a source address 172.16.12.101 in the IPv4 world. Both are as expected. The first line of the debug output confirms that R4's physical interface (14::4) source address is being translated to 172.16.123.101, and that 1144::1 is being translated to 172.16.123.2, R2's address. The tenth line of the debug output confirms that R4's loopback interface (104::1) source address is being translated to 172.16.12.100, and that 1144::1 is again being translated to 172.16.123.2, R2's address. Notice that both source and destination addresses are being translated, and that both translations are occurring at the same time.

Example 8-130. Verifying Dynamic NAT-PT

Code View: Scroll / Show All

```

R2(config)#service finger
R2(config)#

```

```

R1(config)#do debug ipv6 nat
IPv6 NAT-PT debugging is on
R1(config)#

```

```

R4#telnet 1144::1 finger

```

```

Trying 1144::1, 79 ... Open

```

| Line | User | Host(s) | Idle | Location |
|------|------|---------|------|----------|
|------|------|---------|------|----------|

```

0 con 0 idle 00:01:19
*194 vty 0 idle 00:00:00 172.16.123.101
Interface User Mode Idle Peer Address

```

[Connection to 1144::1 closed by foreign host]

R4#

R4#**telnet 1144::1 finger /source-interface loopback 104**

Trying 1144::1, 79 ... Open

```

Line User Host(s) Idle Location
0 con 0 idle 00:01:48
*194 vty 0 idle 00:00:00 172.16.12.100
Interface User Mode Idle Peer Address

```

[Connection to 1144::1 closed by foreign host]

R4#

R1(config)#

IPv6 NAT: IPv6->IPv4: tcp src (14::4) -> (172.16.123.101), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.123.101) -> (14::4)

IPv6 NAT: IPv6->IPv4: tcp src (14::4) -> (172.16.123.101), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv6->IPv4: tcp src (14::4) -> (172.16.123.101), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.123.101) -> (14::4)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.123.101) -> (14::4)

IPv6 NAT: IPv6->IPv4: tcp src (14::4) -> (172.16.123.101), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv6->IPv4: tcp src (14::4) -> (172.16.123.101), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.123.101) -> (14::4)

IPv6 NAT: IPv6->IPv4: tcp src (104::1) -> (172.16.12.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.12.100) -> (104::1)

IPv6 NAT: IPv6->IPv4: tcp src (104::1) -> (172.16.12.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv6->IPv4: tcp src (104::1) -> (172.16.12.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.12.100) -> (104::1)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.12.100) -> (104::1)

IPv6 NAT: IPv6->IPv4: tcp src (104::1) -> (172.16.12.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv6->IPv4: tcp src (104::1) -> (172.16.12.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.12.100) -> (104::1)

Example 8-131 displays the final two Telnet tests from R4, with similar results. The first one is to R2's second address from R4's physical address, and the second one is to R2's second address from R4's loopback address.

Example 8-131. Continuing Verification of Dynamic NAT-PT from R4

```
Code View: Scroll / Show All
R4#telnet 1144::2 finger
Trying 1144::2, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:06:53
*194 vty 0 idle 00:00:00 172.16.123.101
 Interface User Mode Idle Peer Address
[Connection to 1144::2 closed by foreign host]
R4#
R4#telnet 1144::2 finger /source-interface loopback 104
Trying 1144::2, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:07:11
*194 vty 0 idle 00:00:00 172.16.12.100
 Interface User Mode Idle Peer Address
[Connection to 1144::2 closed by foreign host]
R4#
```

Example 8-132 displays the four Telnet tests from R3, with similar results. The first test is to R2's first address from R3's physical address. The second test is to R2's second address from R3's physical address. The third test is to R2's first address from R3's loopback address. The final test is to R2's second address from R3's loopback address. All are successful and indicate an appropriate source address was selected.

Example 8-132. Continuing Verification of Dynamic NAT-PT from R4

```
Code View: Scroll / Show All
R3#telnet 1144::1 finger
Trying 1144::1, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:07:58
*194 vty 0 idle 00:00:00 172.16.123.100
 Interface User Mode Idle Peer Address
[Connection to 1144::1 closed by foreign host]
R3#telnet 1144::2 finger
Trying 1144::2, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:08:12
*194 vty 0 idle 00:00:00 172.16.123.100
 Interface User Mode Idle Peer Address
[Connection to 1144::2 closed by foreign host]
R3#
```

```

R3#telnet 1144::1 finger /source-interface loopback 103
Trying 1144::2, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:09:52
*194 vty 0 idle 00:00:00 172.16.12.101
 Interface User Mode Idle Peer Address
[Connection to 1144::1 closed by foreign host]
R3#
R3#telnet 1144::2 finger /source-interface loopback 103
Trying 1144::2, 79 ... Open
 Line User Host(s) Idle Location
 0 con 0 idle 00:10:05
*194 vty 0 idle 00:00:00 172.16.12.101
 Interface User Mode Idle Peer Address
[Connection to 1144::2 closed by foreign host]
R3#

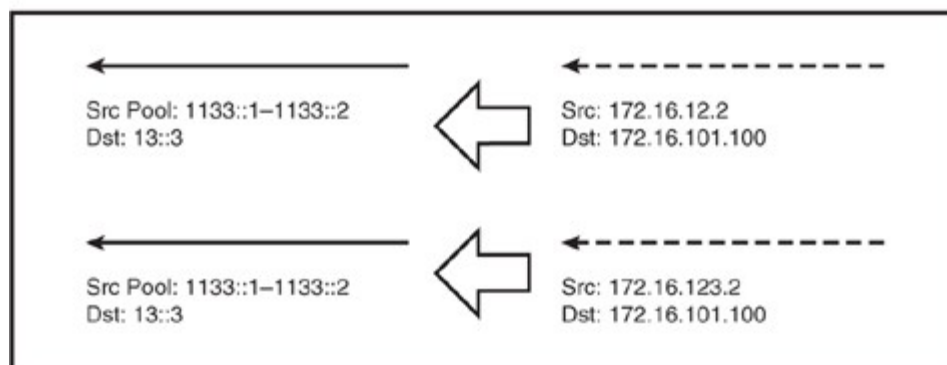
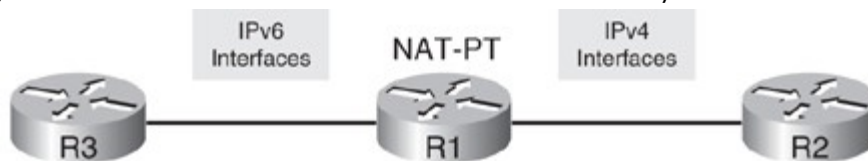
```

#### IPv4-to-IPv6 Dynamic NAT-PT Configuration and Verification Example

This second example uses the same network as in Figure 8-59. In this scenario, however, IPv4 hosts are accessing IPv6 hosts. The traffic flows are sourced from R2's physical interfaces (in the IPv4 world) and destined to R3's physical interface (in the IPv6 world).

Figure 8-61 displays the translations that will occur in this example. For traffic sourced from R2's physical interfaces (172.16.12.2 and 172.16.123.2), a pool of 1133::1 to 1133::2 is used. R3's physical interface address (13::3) is seen in the IPv4-only world as 172.16.101.100.

Figure 8-61. Address Translations Performed in Second Dynamic NAT-PT Example.



IPv6 NAT-PT support is already configured on all four interfaces of router R1. This is confirmed in the show ipv6 nat statistics command output displayed in Example 8-133. Notice that all four of R1's physical interfaces are running NAT-PT, as is the NVI0 interface.

Example 8-133. Confirming NAT-PT is Running on R1's Interfaces

```

R1#show ipv6 nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)

```

NAT-PT interfaces:

FastEthernet0/0, Serial0/0/0.2, Serial0/0/0.4, Serial0/1/0, NVI0

Hits: 0 Misses: 0

Expired translations: 10

R1#

In the IPv6 world, the `ipv6 nat prefix` command defines the prefix that forces traffic to the NAT-PT router. There is no equivalent command in the IPv4 world, so additional configuration is required. R3's physical interface address will be translated to an address on the 172.16.101.0 subnet. R1 therefore needs to advertise this subnet to R2. However, R1 must have an interface with an address in a subnet before it will advertise that subnet. Therefore, as shown in Example 8-134, a loopback interface is created on R1, and it is given an address in this subnet. On R1 and R2, an IPv4 RIP process is created to advertise the 172.16.0.0 network. The example confirms that R2 sees the 172.16.101.0 subnet. In fact, it has two equal-cost routes to it. R2 is therefore able to access the subnet on which R3 will appear to reside in the IPv4 world.

Example 8-134. Configuring and Verifying IPv4 Connectivity

Code View: Scroll / Show All

```
R1(config)#interface loopback 101
```

```
R1(config-if)#
```

```
%LINK-3-UPDOWN: Interface Loopback101, changed state to up
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback101, changed state to up
```

```
R1(config-if)#ip address 172.16.101.1 255.255.255.0
```

```
R1(config-if)#exit
```

```
R1(config)#
```

```
R1(config)#router rip
```

```
R1(config-router)#network 172.16.0.0
```

```
R1(config-router)#
```

```
R2(config)#router rip
```

```
R2(config-router)#network 172.16.0.0
```

```
R2(config-router)#do show ip route
```

Codes: 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

Gateway of last resort is not set

172.16.0.0/24 is subnetted, 3 subnets

```
C 172.16.12.0 is directly connected, Serial0/0/0
```

```
C 172.16.123.0 is directly connected, Serial0/1/0
```

```
R 172.16.101.0 [120/1] via 172.16.123.1, 00:00:07, Serial0/1/0
 [120/1] via 172.16.12.1, 00:00:01, Serial0/0/0
```

```
R2(config-router)#
```



Example 8-135 illustrates the NAT-PT configuration on R1. First, the `ipv6 nat v6v4 source` command configures the static mapping between R3's physical interface (13::3) and the 172.16.101.100 address that R3 appears as in the IPv4 world. Next the `ipv6 nat v4v6 source list` command is used to configure the dynamic mapping for R2's IPv4 source addresses (172.16.12.2 and 172.16.123.2) and the IPv6 addresses that R2 appears as in the IPv6 world (1144::1 and 1144::2). The command uses the access list IPV4 to identify the R2 interface IPv4 addresses, and an address pool POOL\_1144 to define the IPv6 translated addresses. These translated IPv6 address do not exist in the IPv6 world; they are on the NAT-PT prefix, which is configured next. Finally, connected routes (including this prefix) are redistributed into RIPng (with a seed metric of 1).

Example 8-135. Configuring Dynamic NAT-PT

```
R1(config)#ipv6 nat v6v4 source 13::3 172.16.101.100
R1(config)#ipv6 nat v4v6 source list IPV4 pool POOL_1144
R1(config)#ipv6 nat v4v6 pool POOL_1144 1144::1 1144::2 prefix-length 96
R1(config)#ip access-list standard IPV4
R1(config-std-nacl)#permit 172.16.123.0 0.0.0.255
R1(config-std-nacl)#permit 172.16.12.0 0.0.0.255
R1(config-std-nacl)#ipv6 nat prefix 1144::/96
R1(config)#ipv6 router rip NAT-PT
R1(config-rtr)#redistribute connected metric 1
R1(config-rtr)#
```

Example 8-136 shows the output of the `show ipv6 route rip` command on R3, confirming that it has the 1144::/96 prefix in its routing table. Also shown are the IPv6 NAT-PT statistics and translations on R1. Notice that there is one static entry in the table, but no dynamic entries yet because no traffic has been sent.

Example 8-136. Verifying NAT-PT Configuration

```
Code View: Scroll / Show All
R3#show ipv6 route rip
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
 U - Per-user Static route
 I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
 O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
 ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R 14::/64 [120/2]
 via FE80::1, FastEthernet0/0
R 104::/64 [120/3]
 via FE80::1, FastEthernet0/0
R 1144::/96 [120/2]
 via FE80::1, FastEthernet0/0
R3#

R1(config-rtr)#do show ipv6 nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
NAT-PT interfaces:
 FastEthernet0/0, Serial0/0/0.2, Serial0/0/0.4, Serial0/1/0, NVI0
Hits: 0 Misses: 0
Expired translations: 10

R1(config-rtr)#do show ipv6 nat translations
Prot IPv4 source IPv6 source
```

```

 IPv4 destination IPv6 destination
--- 172.16.101.100 13::3
 -- ---
R1(config-rtr)#

```

The service finger command was previously entered on R3. Using a similar set of verification tests to those used in the previous example, debug ipv6 nat is enabled on R1 and some Telnets are performed from R2. The results are shown in Example 8-137. The first Telnet is from R2's Serial 0/1/0 address to R3's IPv4 world address. The second Telnet is from R2's Serial 0/0/0 address to R3's IPv4 world address. The output confirms that R2's addresses are being translated to 1144::1 and 1144::2, as expected. The first line of the debug output confirms that R2's physical interface (172.16.123.2) source address is being translated to 1144::1, and that 172.16.101.100 is being translated to 13::3, R3's address. The tenth line of the debug output confirms that R2's other physical interface (172.16.12.2) source address is being translated to 1144::2, and that 172.16.101.100 is again being translated to 13::3, R3's address. Notice again that both source and destination addresses are being translated, and that both translations are occurring at the same time.

Example 8-137. Verifying Dynamic NAT-PT

Code View: Scroll / Show All

R1#**debug ipv6 nat**

IPv6 NAT-PT debugging is on

R1#

R2#**telnet 172.16.101.100 finger /source-interface s0/1/0**

Trying 172.16.101.100, 79 ... Open

| Line       | User | Host(s) | Idle     | Location     |
|------------|------|---------|----------|--------------|
| 0 con 0    |      | idle    | 00:11:05 |              |
| *194 vty 0 |      | idle    | 00:00:00 | 1144::1      |
| Interface  | User | Mode    | Idle     | Peer Address |

[Connection to 172.16.101.100 closed by foreign host]

R2#

R2#**telnet 172.16.101.100 finger /source-interface s0/0/0**

Trying 172.16.101.100, ... Open

| Line       | User | Host(s) | Idle     | Location     |
|------------|------|---------|----------|--------------|
| 0 con 0    |      | idle    | 00:11:18 |              |
| *194 vty 0 |      | idle    | 00:00:00 | 1144::2      |
| Interface  | User | Mode    | Idle     | Peer Address |

[Connection to 172.16.101.100 closed by foreign host]

R2#

R1#

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.101.100) -> (13::3)

IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.101.100) -> (13::3)

IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.101.100) -> (13::3)

IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::1) -> (172.16.123.2)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.101.100) -> (13::3)

IPv6 NAT: IPv4->IPv6: src (172.16.123.2) -> (1144::1), dst (172.16.101.100) -> (13::3)

```

IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::1) -> (172.16.123.2)
IPv6 NAT: IPv4->IPv6: src (172.16.12.2) -> (1144::2), dst (172.16.101.100) -> (13::3)
IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::2) -> (172.16.12.2)
IPv6 NAT: IPv4->IPv6: src (172.16.12.2) -> (1144::2), dst (172.16.101.100) -> (13::3)
IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::2) -> (172.16.12.2)
IPv6 NAT: IPv4->IPv6: src (172.16.12.2) -> (1144::2), dst (172.16.101.100) -> (13::3)
IPv6 NAT: IPv6->IPv4: tcp src (13::3) -> (172.16.101.100), dst (1144::2) -> (172.16.12.2)
IPv6 NAT: IPv4->IPv6: src (172.16.12.2) -> (1144::2), dst (172.16.101.100) -> (13::3)
IPv6 NAT: IPv4->IPv6: src (172.16.12.2) -> (1144::2), dst (172.16.101.100) -> (13::3)

```

Example 8-138 again examines the NAT-PT translation table on R1. This time the static translation is the last entry. There are two other dynamic entries because of the Telnet testing.

Example 8-138. NAT-PT Translation Table

```

R1#show ipv6 nat translations
Prot IPv4 source IPv6 source
 IPv4 destination IPv6 destination
-- --
 172.16.123.2 1144::1
-- --
 172.16.12.2 1144::2
-- 172.16.101.100 13::3
 -- --
R1#

```

## Summary

In this chapter, you learned about IPv6. The chapter focused on the following topics:

- The issues associated with IPv4, including address depletion, Internet routing table expansion, and the lack of a true end-to-end model.
- The features of IPv6, including larger address space, elimination of NAT and broadcast addresses, simplified header for improved router efficiency, support for mobility and security, and transition richness.
- The features of IPv6 addresses, including stateless autoconfiguration, prefix renumbering, multiple addresses per interface, link-local addresses, and the ability to use provider-dependent or provider-independent addressing.
- The 40-octet IPv6 header, with its 8 fields plus extension headers to handle options.
- The 128-bit IPv6 addresses written in the format is x:x:x:x:x:x:x:x, where x is a 16-bit hexadecimal field; each x is therefore four hexadecimal digits. The two ways to shorten the written form of IPv6 addresses are leading 0s within each set of four hexadecimal digits can be omitted and a pair of colons (::) can be used, once within an address, to represent any number ("a bunch") of successive 0s.
- The IPv6 address interface ID. For Ethernet, it is based on the MAC address of the interface and is in an EUI-64 format. It is formed by inserting the hexadecimal number FFFE between the upper three bytes (the OUI field) and the lower 3 bytes (the vendor code or serial number field) of the MAC address, and setting the seventh bit in the high-order byte to 1 to indicate the uniqueness of the interface ID.
- The IPv6 address types—unicast (including global, link-local, and the deprecated site-local), multicast (for one to many), and anycast (for one to nearest). There are no broadcast addresses.
- The global unicast address that typically consists of a 48-bit global routing prefix, a 16-bit subnet ID, and a 64-bit interface ID (typically in EUI-64 bit format).

- Link-local addresses, that have a scope limited to the local link and are dynamically created on all IPv6 interfaces by using a specific link-local prefix FE80::/10 and a 64-bit interface identifier.
- The deprecated site-local addresses, that were in the FEC0::/10 range.
- IPv6 multicast addresses, defined by the prefix FF00::/8. The second octet of the address contains the prefix and transient (lifetime) flags, and the scope of the multicast address. Reserved multicast addresses include—FF02::1 (“all nodes” on a link, link-local scope); FF02::2 (“all routers” on a link); FF02::9 (“all RIPng routers” on a link). And FF02::1:FFXX:XXXX (solicited-node multicast on a link)
- IPv6 anycast addresses, allocated from the unicast address space and having the same format as unicast addresses. On a Cisco router, an IPv6 address becomes an anycast address when the **anycast** keyword is added to the end of the **ipv6 address** command.
- The ability to summarize IPv6 addresses, similar to IPv4 address summarization.
- IPv6 address configuration and verification commands:
  - **ipv6 unicast-routing** global configuration command to enable the forwarding of IPv6 unicast datagrams.
  - **ipv6 cef** global configuration command to enable CEFv6.
  - **ipv6 address address/prefix-length [eui-64 | link-local]** interface configuration command to configure an IPv6 address for an interface and enable IPv6 processing on the interface. (Using a manually configured interface ID makes it much easier to remember, and therefore use, the link-local address.)
  - **ipv6 address autoconfig [default]** interface configuration command to enable automatic configuration of IPv6 addresses using stateless autoconfiguration on an interface, and enable IPv6 processing on the interface.
  - **ipv6 unnumbered interface-type interface-number** interface configuration command to enable IPv6 processing on an interface without assigning an explicit IPv6 address to the interface.
  - **ipv6 nd reachable-time milliseconds** interface configuration command to specify the number of milliseconds that a remote IPv6 node is considered reachable.
  - **ipv6 neighbor ipv6-address interface-type interface-number hardware-address** global configuration command to statically configure an entry in the IPv6 neighbor discovery cache, mapping the IPv6 address to the hardware address on an interface.
  - **show ipv6 interface [brief] [type number] [prefix]** EXEC command to display the usability and status of interfaces configured for IPv6.
  - **show ipv6 routers [interface-type interface-number] [conflicts]** EXEC command to display IPv6 router advertisement information received from onlink routers (those locally reachable on the link).
  - **show ipv6 neighbors [interface-type interface-number | ipv6-address | ipv6-hostname | statistics]** EXEC command to display IPv6 ND cache information.
  - **debug ipv6 nd** EXEC command to display messages associated with ICMPv6 neighbor discovery.
  - **debug ipv6 packet [access-list access-list-name] [detail]** EXEC command to display information associated with IPv6 packet processing.
- The neighbor discovery or solicitation phase, similar to IPv4’s ARP: It allows an IPv6 router to determine the link-layer address of neighbors on the same link, to find neighbor routers, and to keep track of those neighbors for stateless autoconfiguration.
- Stateless autoconfiguration: When IPv6 is enabled on a router, it sends RAs (to advertise autoconfiguration parameters including the IPv6 prefix on the link), creates link-local addresses, and joins FF02::1 (all hosts) and FF02::2 (all routers) multicast groups on its interfaces.  
A host without an address sends out an RS. A router sends an RA with a prefix, and the host creates its address from the prefix.  
A host statically configured with an address sends an NS for that address.  
A host sends an NS for its own address for DAD. If unique, the host sends an NA for the address.  
Stateless autoconfiguration can also be used for renumbering.
- For a router to perform stateless autoconfiguration for other devices, it must have the **ipv6 unicast-routing** command. Routers configured with this command generate ICMPv6 RA messages. They do not generate RS messages. Routers configured with the **ipv6 address auto-config** command, and not configured with the **ipv6 unicast-routing** command, generate RS messages only. They do not

generate RA messages. Devices keep their autoconfigured address for its lifetime if the interface stays up.

- The processes used to connect IPv6 devices on:
  - **Broadcast multiaccess connections:** Uses the ICMPv6 ND process (NS and NA messages); can create a static mapping instead between an IPv6 unicast address and a MAC address with the **ipv6 neighbor** command.
  - **Point-to-point connections:** Same process as broadcast multiaccess, using MAC address from a LAN interface to create the interface ID for link-local and global unicast addresses (unless manually configured).
  - **Point-to-multipoint connections over Frame Relay:** Requires mapping between IPv6 addresses (link-local and global unicast) and DLCIs. Link-local mapping and broadcast support on the Frame Relay network is required for routing protocol support.
- The routing protocols available for IPv6, including RIPng, OSPFv3, IS-IS for IPv6, EIGRP for IPv6, and MBGP.
- The types of static routes that can be configured (directly attached, recursive, fully specified, and floating).
- Static route configuration and verification commands:
  - **ipv6 route** *ipv6-prefix/prefix-length {ipv6-address | interface-type interface-number [ipv6-address]}* [*administrative-distance*] [*administrative-multicast-distance | unicast | multicast*] [*next-hop-address*] [**tag** *tag*] global configuration command configures a static route
  - **show ipv6 route** [*ipv6-address | ipv6-prefix/prefix-length | protocol | interface-type interface-number*] EXEC command displays the current contents of the IPv6 routing table
- RIPng features: Based on RIPv2, distance vector, maximum of 15 hops, uses link-local addresses as source addresses, uses multicast address FF02::9, has administrative distance 120, sends updates on UDP port 521.
- RIPng configuration and verification commands:
  - **ipv6 rip** *name* **enable** interface configuration command enables an IPv6 RIPng process on an interface.
  - **ipv6 router rip** *name* global configuration command configures the IPv6 routing process and enters router configuration mode.
  - **no split-horizon** router configuration command disables the split-horizon processing of IPv6 RIPng updates.
  - **show ipv6 protocols** [**summary**] EXEC command display the parameters and current state of the active IPv6 routing protocol processes.
  - **debug ipv6 rip** [*interface-type interface-number*] EXEC command displays IPv6 RIPng routing transaction debug messages.
- OSPFv3 features: Uses link-local addresses as source addresses, multiple addresses and OSPFv3 instances per interface are permitted, supports authentication (using IPsec), runs over a link rather than a subnet, uses the same basic packet types and neighbor discovery and adjacency formation as OSPFv2, operates the same as OSPFv2 over NBMA, has the same LSA aging and flooding as OSPFv2, has the same options as OSPFv2, uses FF02::5 (all OSPF routers on the link-local scope) and FF02::6 (all DRs on the link-local scope) multicast addresses, uses 32-bit router ID, and identifies the DR and BDR by router ID.
- OSPFv3 configuration and verification commands:
  - **ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*] interface configuration command enables OSPFv3 on an interface.
  - **ipv6 router ospf** *process-id* global configuration mode enters OSPFv3 router configuration mode.
  - **router-id** *{ip-address}* router configuration command defines a router ID for OSPFv3.
  - **ipv6 ospf priority** *number-value* interface configuration command changes the OSPFv3 priority.
  - **ipv6 ospf cost** *interface-cost* interface configuration command changes the OSPFv3 interface cost.
  - **auto-cost reference-bandwidth** *ref-bw* router configuration command changes the reference bandwidth used in OSPF interface cost calculations.

- **area area-id stub [no-summary]** router configuration command defines an area as a stub area.
- **area area-id range ipv6-prefix /prefix-length [advertise | not-advertise] [cost cost]** router configuration command summarizes routes at an area boundary.
- **clear ipv6 ospf [process-id] {process | force-ospf | redistribution | counters [neighbor [neighbor-interface | neighbor-id]]}** EXEC command triggers SPF recalculation and repopulation of the RIB.
- **show ipv6 ospf [process-id] [area-id] neighbor [interface-type interface-number] [neighbor-id] [detail]** EXEC command displays OSPFv3 neighbor information.
- **show ipv6 ospf [process-id] [area-id] interface [type number] [brief]** EXEC command displays OSPFv3 interface information.
- **show ipv6 ospf [process-id] [area-id]** command displays general information about the IPv6 OSPF processes.
- EIGRP for IPv6 features: Available in Cisco IOS Release 12.4(6)T and later, uses protocol number 88, has all the same features as EIGRP for IPv4, requires a 32-bit router ID, configured on a per-interface basis, has a shutdown feature (the default), and does not automatically summarize
- EIGRP for IPv6 configuration and verification commands:
  - **ipv6 eigrp as-number** interface configuration command enables EIGRP for IPv6 on an interface.
  - **ipv6 router eigrp as-number** global configuration command creates an EIGRP for IPv6 routing process and puts the router in router configuration mode.
  - **eigrp router-id {ip-address}** router configuration command (or **router-id {ip-address}** router configuration command) defines a router ID for EIGRP for IPv6.
  - **no shutdown** router configuration command enables the EIGRP process.
  - **eigrp stub [receive-only | connected | static | summary | redistributed]** router configuration command configures a router as an EIGRP stub.
  - **ipv6 summary-address eigrp as-number ipv6-address [admin-distance]** interface configuration command configures a summary aggregate address for an interface.
- MBGP features: New identifier for the IPv6 address family, scoped addresses, NEXT\_HOP and NLRI attributes are expressed as IPv6 addresses and prefixes, uses the carrier protocol (to establish sessions) and the passenger protocol (for which it advertises routes).
- MBGP configuration and verification commands:
  - **router bgp autonomous-system** global configuration command enters BGP configuration mode and identifies the local autonomous system in which this router belongs.
  - **bgp router-id ip-address** router configuration command configures a fixed router ID for the local BGP routing process.
  - **neighbor {ipv6-address | peer-group-name} remote-as autonomous-system-number** router configuration command activates a BGP session for external and internal neighbors and identifies a peer router with which the local router will establish a session.
  - **address-family ipv6 [unicast | multicast | vpnv6]** router configuration command enters address family configuration mode for configuring BGP routing sessions that use IPv6 address prefixes.
  - **neighbor ipv6-address activate** address family configuration (or router configuration) command enables the exchange of information with a BGP neighbor
  - **network network-number** address family configuration (or router configuration). command specifies the networks to be advertised by the BGP and MBGP routing process.
  - **neighbor ipv6-address route-map name {in | out}** address family configuration (or router configuration) command applies a route map to filter incoming or outgoing MBGP routes.
- Policy routing configuration and verification commands:
  - **route-map map-tag [permit | deny] [sequence-number]** global configuration command creates a route map.
  - **match ipv6 address {prefix-list prefix-list-name | access-list-name}** route-map configuration command specifies a prefix permitted by a prefix list to use in redistribution or an IPv6 access list to use to match packets for PBR for IPv6.



- **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*] route-map configuration command specifies where to output IPv6 packets that pass a match clause of a route map for PBR.
- **ipv6 policy route-map** *route-map-name* interface configuration command configures IPv6 PBR on an interface. (Configure it on the interface that receives the traffic that should be routed differently than the default path.)
- **ipv6 local policy route-map** *route-map-name* global configuration command identifies a route map to use for local policy routing.
- **ipv6 access-list** *access-list-name* global configuration command defines an IPv6 ACL and places the router in IPv6 access list configuration mode. Within access list configuration mode, **permit** and **deny** statements create the access list.
- **permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} IPv6 access list configuration command defines permit conditions for an IPv6 ACL.
- **ping ipv6** *ipv6-address* **source** *interface-name* command pings from the specified address.
- **debug ipv6 policy** [*access-list-name*] displays IPv6 policy routing packet activity.
- **show ipv6 access-list** [*access-list-name*] displays the contents of all or a specified IPv6 ACL.
- **show route-map** [*map-name*] EXEC command displays configured route maps
- Redistribution between routing protocols:
  - Multiple RIPv3 instances can run on the same router and on the same link and by default will send and receive advertisements between each other. Changing the port and or multicast group addresses separates the processes; redistribution must be configured to share information between these separate processes. Seed metrics may need to be configured to control paths.
  - OSPFv3 does not redistribute connected networks by default.
  - The metric and metric type may be reset when redistributing. The solution is to explicitly configure the seed metric and metric type.
  - Suboptimal routing and routing loops may occur, especially if there are multiple paths between routing domains; careful redistribution configuration is required. Route filtering and changing administrative distance may be required.
- Redistribution configuration and verification commands:
  - **port** *port-number* **multicast-group** *multicast-address* router configuration command configures the specified UDP port and multicast group address for RIPv3 process.
  - **redistribute connected** command redistributes connected routes into a RIPv3 routing process.
  - **redistribute connected** [**metric** *metric-value*] [**metric-type** *type-value*] [**tag** *tag-value*] [**route-map** *map-tag*] command redistributes connected routes into an OSPFv3 process.
  - **redistribute ospf** [*process-id*] [**metric** *metric-value*] [**metric-type** *type-value*] [**tag** *tag-value*] [**route-map** *map-tag*] redistributes the specified OSPF process into another routing process.
  - **redistribute rip** *process-id* redistributes the specified RIPv3 process into another routing process.
  - **redistribute bgp** *autonomous-system* address family (or router) configuration command redistributes BGP into another routing process.
  - The **include-connected** keyword, used in the **redistribute** command, causes both the routing process and connected routes to be redistributed, and saves adding the **redistribute connected** command. But the **include-connected** keyword does not actually work when redistributing from BGP into another protocol (even though it is an option on the command).
  - **set metric** *metric-value* route-map configuration command specifies the routing protocol metric.
  - **clear ipv6 route** {*ipv6-address* | *ipv6-prefix/prefix-length* | \*} privileged EXEC command deletes routes from the IPv6 routing table.
  - **ipv6 prefix-list** *list-name* [**seq** *seq-number*] **permit** *ipv6-prefix/prefix-length* global configuration command creates an IPv6 prefix list entry.



- **distance** *distance* router configuration command configures the administrative distance of IPv6 routes.
- **bgp redistribute-internal** address family (or router) configuration command configures IBGP redistribution into an IGP.
- **show ip bgp summary** command displays entries in the BGP table.
- **show bgp ipv6 unicast summary** command displays a summary of unicast entries in the IPv6 BGP routing table. Note: When IPv6 is used as both the carrier and passenger protocol, the **show ip bgp summary** command displays the carrier protocol neighbor relationship information, while the **show bgp ipv6 unicast summary** command displays the passenger protocol neighbor relationship information.
- Transitioning techniques from IPv4 to IPv6: Dual stack (both protocols running), tunneling IPv6 inside IPv4, and translation with stateful NAT-PT.
- When tunneling IPv6 over IPv4:
  - IPv4 is the transport protocol, the protocol over which the tunnel is created.
  - IPv6 is the passenger protocol, the protocol encapsulated in the tunnel and carried through the tunnel.
  - Another protocol, such as GRE, is used to create the tunnel, and is known as the tunneling (or encapsulation or carrier) protocol. It encapsulates the passenger protocol.
- Manual tunnels:
  - Simulate a permanent link between two IPv6 domains over an IPv4 backbone.
  - Tunnel interfaces are created at each end on dual-stack routers, and IPv6 addresses are manually configured on each tunnel interface.
  - Use IPv4 protocol 41 and add a 20-byte IPv4 header.
  - Created between routers, or between routers and hosts.
  - Simple to configure, useful for a small number of sites.
  - Not scalable.
- Manual tunnel configuration and verification commands:
  - **tunnel source** *interface-type interface-number* interface configuration command sets the source address for a tunnel interface as the address of the specified interface.
  - **tunnel destination** *ip-address* interface configuration command specifies the destination address for a tunnel interface.
  - **tunnel mode ipv6ip** interface configuration command sets the encapsulation mode for the tunnel interface to use IPv6 as the passenger protocol, and IPv4 as both the encapsulation and transport protocol.
  - **debug tunnel** EXEC command displays tunnel encapsulation and decapsulation process.
  - **debug ip packet detail** EXEC command displays details about IP packets traversing the router.
- GRE tunnels:
  - Similar to manual tunnels. Simulate a permanent link between two IPv6 domains over an IPv4 backbone.
  - Developed by Cisco and is the default tunneling protocol.
  - More flexible in the protocols that they support; GRE can be deployed on top of multiple transport protocols and carry multiple passenger protocols.
  - GRE tunnel analogy: the road is the transport protocol (can be IPv4 or IPv6), the truck is the carrier protocol (GRE encapsulation), the payload is the passenger protocol (IPv6 or other).
  - Created between routers, or between routers and hosts.
  - Use IPv4 protocol 47.
- GRE tunnel configuration and verification commands:
  - **tunnel mode gre** interface configuration command sets the encapsulation mode for the tunnel interface to use GRE. This is the default mode, so this command is not required unless changing the mode from another mode.
  - **tunnel mode gre ipv6** interface configuration command sets the encapsulation mode for the tunnel interface to use GRE tunneling using IPv6 as the transport protocol.
- 6to4 tunnels:

- Automatic point-to-multipoint tunneling method. 6to4 tunnels are built automatically by the edge routers, based on embedded IPv4 addresses within the IPv6 addresses of the tunnel interfaces on the edge routers.
- 6to4 edge router, which is a dual-stacked device, has an IPv6 address with a /48 prefix, which is the concatenation of 2002::/16 and the hexadecimal representation of the IPv4 address of that edge router. The edge routers automatically build the tunnel using the IPv4 addresses that are embedded in the IPv6 addresses.
- Only static routes or BGP can be used across 6to4 tunnel because the other routing protocols use link-local addresses to form adjacencies and exchange updates, and link-local addresses do not conform to the address requirements for 6to4 tunnels.
- Good approach to use during migration to IPv6, but as a temporary rather than permanent solution. After migration, the IPv6 network will likely be renumbered, to remove the restrictive addressing scheme required by these tunnels.
- 6to4 tunnel configuration and verification commands:
  - **tunnel mode ipv6ip 6to4** interface configuration command specifies IPv6 automatic tunneling mode using a 6to4 address.
  - **debug ipv6 packet detail EXEC** command displays details about IPv6 packets traversing the router.
- IPv4-compatible IPv6 tunnels:
  - Deprecated.
  - Similar to 6to4 tunnels. Automatic point-to-multipoint tunneling method. Tunnels are built automatically by routers, based on embedded IPv4 address within the IPv6 addresses of the tunnel interfaces on the edge routers.
  - The IPv4-compatible IPv6 address is just an IPv6 address with 96 bits filled with zeros followed by the 32-bit IPv4 address in the lower 32 bits. The 32-bit portion is even written in dotted-decimal format.
  - Only static routes or BGP can be used across tunnel because the other routing protocols use link-local addresses to form adjacencies and exchange updates, and link-local addresses do not conform to the address requirements for these tunnels.
- IPv4-compatible IPv6 tunnel configuration and verification command—the **tunnel mode ipv6ip auto-tunnel** interface configuration command specifies IPv6 automatic tunneling mode using an IPv4-compatible IPv6 address.
- ISATAP tunnels:
  - Similar to 6to4 and IPv4-compatible IPv6 tunnels. Automatic; embeds an IPv4 address within the IPv6 address. Tunnel established only when needed.
  - Scalable transition solution.
  - Typically from host to router.
  - Designed to transport IPv6 packets within a site.
  - Hosts can discover the tunnel destination address (the IPv4 address of the router) dynamically through DNS or via a local definition (on the host itself).
  - Format of address: 64-bit prefix is concatenated to a 64-bit interface ID in EUI-64 format. Upper 32 bits of the interface ID are 0000:5EFE, a reserved OUI value indicating an IPv6 ISATAP address. The lower (least significant) 32 bits of the interface ID contain the IPv4 address of the interface (written in hexadecimal).
  - Does not support IPv6 multicast, so IGP's not inherently supported.
  - ISATAP tunnels have predictable link-local addresses that are automatically created and use the tunnel mechanism. ISATAP is designed for traffic within a site, so that for hosts the tunnel appears as a connection to a normal Ethernet interface.
- ISATAP tunnel configuration and verification command: **tunnel mode ipv6ip isatap** interface configuration command specifies IPv6 automatic tunneling mode using an ISATAP address
- Translation using NAT-PT:
  - For direct communication between IPv6-only and IPv4-only networks.
  - All configuration and translation is performed on the NAT-PT router; the other devices in the network are not aware of the existence of the other protocol's network, nor that translations are occurring.

- The NAT-PT router translates source and destination addresses and other packet header fields in both directions: from the IPv4 network to the IPv6 network, and from the IPv6 network to the IPv4 network. Therefore, this router is dual stacked and must have two sets of translation entries for this bidirectional translation.
- The NAT-PT router translates both the source and destination addresses in both directions.
- NAT-PT uses a 96-bit IPv6 network prefix to direct all IPv6 traffic that needs to be translated to the NAT-PT router.
- Static NAT-PT is simple to configure, but it requires planning the translated addresses in both domains. Static NAT-PT is not scalable.
- With dynamic NAT-PT, addresses are allocated from an address pool.
- Static NAT-PT configuration and verification commands:
  - **ipv6 nat** interface configuration command designates that traffic originating from or destined for the interface is subject to NAT-PT.
  - **ipv6 nat prefix** *ipv6-prefix/prefix-length* global or interface configuration command assigns an IPv6 prefix where matching IPv6 packets will be translated using NAT-PT.
  - **ipv6 nat v4v6 source** *ipv4-address ipv6-address* global configuration command configures IPv4 to IPv6 static address translation using NAT-PT.
  - **ipv6 nat v6v4 source** *ipv6-address ipv4-address* global configuration command configures IPv6 to IPv4 static address translation using NAT-PT.
  - **show ipv6 nat translations** EXEC command displays active NAT-PT translations.
- Dynamic NAT-PT configuration and verification commands:
  - **ipv6 nat v4v6 source** {**list** {*access-list-number* | *name*} **pool** *name*} global configuration command configures IPv4 to IPv6 dynamic address translation using NAT-PT. Packets that match the IPv4 ACL are translated to addresses in the IPv6 pool.
  - **ipv6 nat v4v6 pool** *name start-ipv6 end-ipv6 prefix-length prefix-length* defines a pool of IPv6 addresses for NAT-PT.
  - **ipv6 nat v6v4 source** {**list** {*access-list-name*} **pool** *name*} global configuration command configures IPv6 to IPv4 dynamic address translation using NAT-PT. Packets that match the IPv6 ACL are translated to addresses in the IPv4 pool.
  - **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4 prefix-length prefix-length* defines a pool of IPv4 addresses for NAT-PT.
  - **service finger** global configuration command configures a system to accept Finger protocol requests. When this command is configured, the router responds to a **telnet address finger** command from a remote host by immediately displaying the output of the **show users** command and then closing the connection.
  - **show ipv6 nat statistics** EXEC command displays NAT-PT statistics
  - **debug ipv6 nat** EXEC command displays debug messages for NAT-PT translation events.

## References

For additional information, refer to the following resources:

- The Cisco IOS Software Releases 12.4 Mainline Command References, available at [http://www.cisco.com/en/US/products/ps6350/prod\\_command\\_reference\\_list.html](http://www.cisco.com/en/US/products/ps6350/prod_command_reference_list.html)
- The Cisco IOS IPv6 Command Reference, available at [http://www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6\\_book.html](http://www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6_book.html)
- The Cisco IOS IPv6 Configuration Guide, Release 12.4, available at [http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/12\\_4/ipv6\\_12\\_4\\_book.html](http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/12_4/ipv6_12_4_book.html)
- Cisco IOS IPv6 Multicast Introduction, available at [http://www.cisco.com/en/US/tech/tk828/technologies\\_white\\_paper09186a0080203e90.shtml](http://www.cisco.com/en/US/tech/tk828/technologies_white_paper09186a0080203e90.shtml)

## Review Questions

Answer the following questions, and then refer to Appendix A, "Answers to Review Questions," for the answers.

1. What are some of the issues associated with IPv4?
2. What are some of the features of IPv6?

3. In general which node processes IPv6 extension headers?
4. How many bits are in an IPv6 address?
5. In what format are IPv6 addresses written?
6. Which of the following are valid representations of the IPv6 address 2035:0001:2BC5:0000:0000:087C:0000:000A?
  - a. 2035:0001:2BC5::087C::000A
  - b. 2035:1:2BC5::87C:0:A
  - c. 2035:0001:2BC5::087C:0000:000A
  - d. 2035:1:2BC5:0:0:87C::A
  - e. 2035:1:2BC5::087C:A
7. How is an IPv6 interface identifier created for Ethernet interfaces?
8. What is the format of an IPv6 broadcast address?
9. Which of the following are the true statements?
  - a. A packet that is sent to an IPv6 anycast address goes to the closest interface identified by that address.
  - b. A packet that is sent to an IPv6 anycast address goes to all interfaces identified by that address.
  - c. A packet that is sent to an IPv6 multicast address goes to the closest interface identified by that address.
  - d. A packet that is sent to an IPv6 multicast address goes to all interfaces identified by that address.
10. What is the IPv6 unicast address space?
11. What is the IPv6 link-local prefix?
12. What is the IPv6 site-local prefix?
13. What is the IPv6 multicast prefix?
14. What is an IPv6 solicited-node multicast address used for?
15. What is the summary of the following IPv6 routes: 2001: 00C0::/32, 2001: 00D0::/32, 2001: 00E0::/32, and 2001: 00F0::/32?
16. What is the command used to configure an interface with the IPv6 prefix 2001:ABCD::/64, using the interface ID derived from the MAC address?
17. What is the command used to configure a second IPv6 address on an interface?
18. What does the command **ipv6 unnumbered loopback 10** do?
19. Why might you want to statically configure a link-local address?
20. How does an IPv6 device determine if its IPv6 address is unique?
21. How does IPv6's stateless autoconfiguration work?
22. Describe the processes used to connect IPv6 devices on broadcast multiaccess connections, point-to-point connections, and point-to-multipoint connections over Frame Relay?
23. Which of the following routing protocols support IPv6?
  - a. RIPv2
  - b. OSPFv6
  - c. RIPv6
  - d. EIGRP for IPv6
  - e. OSPFv2
  - f. MBGP
24. What does the **ipv6 route 2001:CC1E::/32 2001:12::1** command do?
25. The **ipv6 rip RIPON enable** command is entered on the Fast Ethernet 0/0 interface. Is the **ipv6 router rip RIPON** command also required on the router?
26. What are some of the similarities between OSPFv2 and OSPFv3?
27. Fill in the following table to indicate the OSPFv3 packet types.

| Packet Type | Name |
|-------------|------|
| 1           |      |
| 2           |      |

| Packet Type | Name |
|-------------|------|
| 3           |      |
| 4           |      |
| 5           |      |

28. Which IPv6 multicast addresses does OSPFv3 use?
29. How many bits is an OSPFv3 router ID?
30. Which command is used to summarize IPv6 OSPF routes?
31. How is the EIGRP for IPv6 router ID configured?
32. How does EIGRP for IPv6 auto summarize?
33. MBGP is running in an IPv4 network and is sending IPv6 prefix information. What is the passenger protocol? What is the carrier protocol?
34. Write the MBGP configuration for Router A with one MBGP neighbor, 2001:200:2:4::1. Both routers are in AS 200. Router A's BGP router ID should be set to 2.2.2.2. Router A should redistribute connected routes to its neighbor.
35. Write the configuration for a route map that sets the next hop to 154::5 for packets with a source address in the 888::/64 network coming in on the Fast Ethernet 0/0 interface of a router.
36. How can multiple RIPng instances running on a router be separated so that they do not send and receive information from each other?
37. OSPFv3 process 2 and RIPng process RIPON are running on a router. Write the configuration to redistribute OSPFv3 routes into RIPng with a metric of 1, and to redistribute RIPng routes into OSPFv3.
38. Which command configures IBGP redistribution into an IGP?
39. What are some of the techniques available to transition from IPv4 to IPv6?
40. What protocol numbers do manual and GRE tunnels use?
41. Which command configures a tunnel to use GRE tunneling and IPv6 as the transport protocol?
42. What addresses do the routers involved in 6to4 tunneling use?
43. What is the IPv4-compatible IPv6 address associated with the IPv4 address 172.17.2.1?
44. What is the format of an ISATAP tunnel address?
45. Write the static NAT-PT translation configuration so that a device with address 14::4 can talk to an IPv4 address with device 172.16.14.2. Use 2244::1 to represent the IPv4 device in the IPv6 world. Use 172.16.14.100 to represent the IPv6 device in the IPv4 world. Serial 0/0/0 is the IPv6 interface on the router. Serial 0/1/0 is the IPv4 interface on the router. The RIPng process RIPON is running on the Serial 0/0/0 interface. A metric of 1 should be used for any redistributions.
46. In the **ipv6 nat v4v6 source** {**list** {*access-list-number* | *name*} **pool** *name*} global configuration command, how are the parameters used?
47. In the **ipv6 nat v6v4 source** {**list** {*access-list-name*} **pool** *name*} global configuration command, how are the parameters used?

## Appendix A. Answers to Review Questions

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Chapter 7

Chapter 8

Chapter 1

1. A converged network is one in which data, voice, and video traffic coexists on a single network.
2. Integrated transport, integrated services, and integrated applications
3. b, c, e
4. Campus, data center, branches, teleworkers, and WAN
5. a, c, d
6. The Enterprise Composite Network Model first divides the network into three functional areas, as follows:
  - **Enterprise Campus**— This functional area contains the modules required to build a hierarchical, highly robust campus network.
  - **Enterprise Edge**— This functional area aggregates connectivity from the various elements at the edge of the enterprise network, including to remote locations, the Internet, and remote users.
  - **Service Provider Edge**— This area is not implemented by the organization; instead, it is included to represent connectivity to service providers.
7. The Enterprise Campus functional area comprises the following modules: Building, Building Distribution, Core (also called the backbone), Edge Distribution, Server Farm, and Management.
8. a, b, d
9. Prepare, Plan, Design, Implement, Operate, and Optimize (PPDIOO)
10. In the Design phase
11. b, c, e
12. a, b, d, e
13. c.
14. a, d
15. b
16. a, b
17. c
18. b
19. a
20. b
21. d
22. b
23. When a router that is using a classful routing protocol sends an update about a subnet of a network across an interface belonging to a different network, the router assumes that the remote router will use the default subnet mask for that class of IP address. Therefore, when the router sends the update, it does not include the subnet information; the update packet contains only the major (classful) network information.
24. True
25. b
26. d
27. c
28. b
29. a, d, e

30. Each part of the network might have different routing protocol requirements. For example, BGP might be required in the Corporate Internet module, whereas static routes are often used for remote-access and VPN users. Therefore, enterprises might have to run multiple routing protocols.

## Chapter 2

1. Features of EIGRP include fast convergence, use of partial updates, multiple network layer support, use of multicast and unicast, VLSM support, seamless connectivity across all data link layer protocols and topologies, and sophisticated metrics.
2. EIGRP operational traffic is multicast (and unicast).
3. The four key technologies are neighbor discovery/recovery mechanism, Reliable Transport Protocol (RTP), Diffusing Update Algorithm (DUAL) finite-state machine, and protocol-dependent modules.
4. b
5. EIGRP uses the following five types of packets:
  - **Hello**— Hello packets are used for neighbor discovery. They are sent as multicasts and do not require an acknowledgment. (They carry an acknowledgment number of 0.)
  - **Update**— Update packets contain route change information. An update is sent to communicate the routes a particular router has used to converge. An update is sent only to affected routers. Update packets are sent as multicasts when a new route is discovered and when convergence is completed (in other words, when a route becomes passive). To synchronize topology tables, update packets are sent as unicasts to neighbors during their EIGRP startup sequence. Update packets are sent reliably.
  - **Query**— When a router is performing route computation and does not find an FS, it sends a query packet to its neighbors, asking if they have a successor to the destination. Queries are normally multicast but can be retransmitted as unicast packets in certain cases. They are sent reliably.
  - **Reply**— A reply packet is sent in response to a query packet. Replies are unicast to the originator of the query and are sent reliably. A router must reply to all queries.
  - **Acknowledge (ACK)**— The ACK is used to acknowledge updates, queries, and replies. ACK packets are unicast hello packets and contain a nonzero acknowledgment number. (Note that hello and ACK packets do not require acknowledgment.)
6. EIGRP hello packets are sent every 5 seconds on LAN links.
7. The hello interval determines how often hello packets are sent. It is 5 or 60 seconds by default, depending on the media type.

The hold time is the amount of time a router considers a neighbor up without receiving a hello or some other EIGRP packet from that neighbor. Hello packets include the hold time. The hold time interval is set by default to 3 times the hello interval.
8. a, c, e
9. d
10. EIGRP update packets are generated by the Reliable Transport Protocol (RTP) within EIGRP. Reliable packets have a sequence number assigned to them, and the receiving device must acknowledge them.
11. a, c
12. A feasible successor (FS) is a router providing a backup route. The route through the feasible successor must be loop free. In other words, it must not loop back to the current successor. To qualify as an FS, a next-hop router must have an advertised distance less than the feasible distance of the current successor route for the particular network.

| 13.                | <table><tr><th>Term</th><th>Description Letter</th></tr><tr><td>Successor</td><td>d</td></tr><tr><td>Feasible successor</td><td>e</td></tr><tr><td>Hello</td><td>g</td></tr><tr><td>Topology table</td><td>b</td></tr><tr><td>IP</td><td>a</td></tr><tr><td>Update</td><td>h</td></tr></table> | Term | Description Letter | Successor | d | Feasible successor | e | Hello | g | Topology table | b | IP | a | Update | h |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|--------------------|-----------|---|--------------------|---|-------|---|----------------|---|----|---|--------|---|
| Term               | Description Letter                                                                                                                                                                                                                                                                             |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| Successor          | d                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| Feasible successor | e                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| Hello              | g                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| Topology table     | b                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| IP                 | a                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |
| Update             | h                                                                                                                                                                                                                                                                                              |      |                    |           |   |                    |   |       |   |                |   |    |   |        |   |



| Term          | Description Letter |
|---------------|--------------------|
| IPv6          | a                  |
| Routing table | c                  |
| DUAL          | f                  |
| IPX           | a                  |

14. The first number is the feasible distance for the network through the next-hop router, and the second number is the advertised distance from the next-hop router to the destination network.
15. EIGRP performs autosummarization. True  
EIGRP autosummarization cannot be turned off. False  
EIGRP supports VLSM. True  
EIGRP can maintain independent routing tables. True  
The EIGRP hello interval is an unchangeable fixed value. False
16. IGRP and EIGRP use the same algorithm for metric calculation, but EIGRP's metric value is multiplied by 256 to provide it more granular decision making. EIGRP represents its metrics in 32-bit format rather than the 24-bit representation used by IGRP.
17. The default formula for the EIGRP metric is metric = bandwidth + delay.  
In this formula, the EIGRP delay value is the sum of the delays in the path, in tens of microseconds, multiplied by 256.  
The bandwidth is calculated using the minimum bandwidth link along the path, represented in kbps.  $10^7$  is divided by this value, and then the result is multiplied by 256.
18. The tasks include the following:
  - Enabling the EIGRP routing protocol
  - Configuring the proper network statements
  - Optionally configuring the metric to appropriate interfaces
19. One EIGRP configuration for Router A is as follows:  
RouterA(config)#**router eigrp 100**  
RouterA(config-router)#**network 172.16.2.0 0.0.0.255**  
RouterA(config-router)#**network 172.16.5.0 0.0.0.255**
20. The **passive-interface** command prevents a routing protocol's routing updates from being sent through the specified router interface. When you use the **passive-interface** command with EIGRP, hello messages are not sent out of the specified interface. Neighboring router relationships do not form with other routers that can be reached through that interface (because the hello protocol is used to verify bidirectional communication between routers). Because no neighbors are found on an interface, no other EIGRP traffic is sent.
21. On Router R2 the EIGRP-learned 172.17.0.0 network will be flagged as a candidate default network (indicated by the \* in the routing table). R2 will also set the gateway of last resort as Router R1's address, to reach the default network of 172.17.0.0. Router R1 will have the 172.17.0.0 network flagged as a default network and will have its gateway of last resort set.
22. a, c
23. a, c, d
24. frame-relay map ip 172.16.1.2 100 broadcast
25. By default, split horizon is enabled on most interfaces. Split horizon is disabled by default on Frame Relay physical interfaces, but is enabled by default on Frame Relay multipoint subinterfaces.
26. The **neighbor** command is used to define a neighboring router with which to exchange EIGRP routing information. Instead of using multicast packets, EIGRP exchanges routing information with the specified neighbor using unicast packets. EIGRP does not process any multicast packets coming inbound on that interface and it stops sending multicast packets on that interface.
27. The Layer 2 MPLS VPN provides a Layer 2 service across the backbone, where routers are connected together on the same IP subnet. The Layer 3 MPLS VPN provides a Layer 3 service across the backbone, where routers are connected to ISP edge routers, and on each side, a separate IP subnet is used.

28. A neighbor relationship is established directly between Routers R1 and R2 over the MPLS backbone. From the EIGRP perspective, the MPLS backbone and routers PE1 and PE2 are not visible.
29. a, b, c
30. d
31. c
32. No, the routers must have the same password configured for the link.
33. This command specifies the time period during which the key will be accepted for use on received packets.
34. c
35. The default EIGRP stuck-in-active timer is 3 minutes. If the router does not receive a reply to all the outstanding queries within this time, the route goes to the stuck-in-active state, and the router resets the neighbor relationships for the neighbors that failed to reply.
36. The SIA-Query is sent at the midway point of the active timer (1.5 minutes by default).
37. Summarization limits the query range because a remote router extends the query about a network only if it has an exact match for the network in its routing table.
38. Stub routers are not queried. Instead, hub routers connected to the stub router answer the query on behalf of the stub router.
39. This command makes the router an EIGRP stub. The **receive-only** keyword restricts the router from sharing any of its routes with any other router within the EIGRP autonomous system.
40. EIGRP goodbye messages are sent in hello packets.

### Chapter 3

1. b
2. d
3. a, c
4. b
5. An ABR connects area 0 to a nonbackbone area.
6. b
7. a
8. d
9. d
10. b
11. c. Notice that answer c says the area in which *the router* is to be configured; the necessary information related to area is the area in which *each interface of the router* is to be configured.
12. a
13. b
14. Each router dynamically detect its neighboring router by multicasting its hello packets to all OSPF routers using the address 224.0.0.5.
15. a
16. c
17. Routers A and PEA establish an OSPF adjacency, and Routers B and PEB establish an OSPF adjacency.
18. c
19. a
20. c
21. c
22. If an area becomes too large, some issues that may need to be addressed include:
  - **Frequent SPF algorithm calculations**— In a large network, changes are inevitable, so the routers spend many CPU cycles recalculating the SPF algorithm and updating the routing table.
  - **Large routing table**— OSPF does not perform route summarization by default. If the routes are not summarized, the routing table can become very large, depending on the size of the network.

- **Large LSDB**— Because the LSDB covers the topology of the entire network, each router must maintain an entry for every network in the area, even if not every route is selected for the routing table.

23. 1—C  
2—A  
3—B  
4—D
24. d
25. c
26. b
27. a
28. b
29. b
30. b
31. With OSPF, the specified interface appears as a stub network in the OSPF domain, and OSPF routing information is neither sent nor received through the specified router interface. Configuring an interface as passive disables only the neighbor relationship establishment. The router will still advertise the network to its OSPF neighbors.
32. a, b, d
33. b
34. OSPF does not autosummarize, so it cannot be disabled.
35. The **area 1 range 10.0.0.0 255.255.248.0** command will summarize the specified area 1 routes into area 0.
36. A virtual link is a link that allows discontinuous area 0s to be connected, or a disconnected area to be connected to area 0, via a transit area. The OSPF virtual link feature should be used only in very specific cases, for temporary connections or for backup after a failure. Virtual links should not be used as a primary backbone design feature
37. a
38. c
39. a
40. c
41. b
42. c
43. The **ip ospf authentication-key cisco** interface configuration command configures the password *cisco* for OSPF simple password authentication.
44. a
45. The **ip ospf message-digest-key 1 md5 cisco** interface configuration command configures the password *cisco* as key 1 for OSPF MD5 authentication.

#### Chapter 4

1. Excessive routing updates, incorrectly configured route maps or filters, the number of routing protocols running in the same autonomous system
2. Access lists, route maps, distribute lists, prefix lists
3. You might use multiple routing protocols for the following reasons:
  - When you are migrating from an older IGP to a new IGP, multiple redistribution boundaries might exist until the new protocol has displaced the old protocol completely.
  - You want to use another protocol, but you need to keep the old protocol because of the host system's needs.
  - Different departments might not want to upgrade their routers to support a new routing protocol.
  - If you have a mixed-router vendor environment, you can use a Cisco-specific protocol in the Cisco portion of the network and use a common standards-based routing protocol to communicate with non-Cisco devices.

- Political and geographical border issues, and as your organization merges with or is acquired by other organizations.
- Your organization uses a Layer 3 MPLS VPN service from a service provider that does not use the same IGP as the rest of your network.

4. Cisco routers allow internetworks using different routing protocols (referred to as routing domains or autonomous systems) to exchange routing information through a feature called route redistribution. Redistribution is defined as the capability of boundary routers connecting different routing domains to exchange and advertise routing information between those routing domains (autonomous systems).
5. Redistribution is always performed outbound. The routing table on the router doing the redistribution does not change.
6. The key issues that may arise with redistribution are routing loops, incompatible routing information, and inconsistent convergence times.
7. Depending on how you employ redistribution, routers might send routing information received from one autonomous system back into that same autonomous system. The feedback is similar to the routing loop problem that occurs with distance vector protocols.
8. Each routing protocol is prioritized in order from most to least believable (reliable) using a value called administrative distance. This criterion is the first thing a router uses to determine which routing protocol to believe if more than one protocol provides route information for the same destination. The routing metric is a value representing the path between the local router and the destination network, according to the routing protocol being used. The metric is used to determine the routing protocol's "best" path to the destination.

|    |                                    |                                       |
|----|------------------------------------|---------------------------------------|
| 9. | Routing Protocols                  | Default Administrative Distance Value |
|    | Connected interface                | 0                                     |
|    | Static route out an interface      | 1                                     |
|    | Static route to a next-hop address | 1                                     |
|    | EIGRP summary route                | 5                                     |
|    | External BGP                       | 20                                    |
|    | Internal EIGRP                     | 90                                    |
|    | IGRP                               | 100                                   |
|    | OSPF                               | 110                                   |
|    | IS-IS                              | 115                                   |
|    | RIPv1 and RIPv2                    | 120                                   |
|    | EGP                                | 140                                   |
|    | ODR                                | 160                                   |
|    | External EIGRP                     | 170                                   |
|    | Internal BGP                       | 200                                   |
|    | Unknown                            | 255                                   |

10. When configuring a default metric for redistributed routes, set the metric to a value larger than the largest metric within the receiving autonomous system to help prevent suboptimal routing and routing loops.

|     |                                               |                                                                                                                                                                                      |
|-----|-----------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 11. | Protocol That the Route Is Redistributed Into | Default Seed Metric                                                                                                                                                                  |
|     | RIP                                           | 0, which is interpreted as infinity.                                                                                                                                                 |
|     | IGRP/EIGRP                                    | 0, which is interpreted as infinity.                                                                                                                                                 |
|     | OSPF                                          | 20 for all except BGP routes, which have a default seed metric of 1. (Note that when redistributing OSPF into OSPF, metrics associated with both intra-area and interarea routes are |

| Protocol That the Route Is Redistributed Into | Default Seed Metric                    |
|-----------------------------------------------|----------------------------------------|
|                                               | preserved.)                            |
| IS-IS                                         | 0.                                     |
| BGP                                           | BGP metric is set to IGP metric value. |

12. The safest way to perform redistribution is to redistribute routes in only one direction, on only one boundary router within the network.
13. You can redistribute only protocols that support the same protocol stack. Therefore, redistribution cannot be configured between IPv6 RIPng and IP RIP or between IPv6 EIGRP and IP EIGRP. Redistribution can be configured between IP EIGRP and OSPF.
14. The *metric-value* parameter in the **redistribute** command for RIP is an optional parameter used to specify the RIP seed metric for the redistributed route. The default seed metric for RIP is 0, which is interpreted as infinity. The metric for RIP is hop count.
15. The **subnets** keyword is not configured on this **redistribute** command. As a result, the 10.1.0.0/16 and 10.3.0.0/16 routes are *not* redistributed into the OSPF domain.
16. The components of the EIGRP metric are as follows:
  - **Bandwidth**— The route's minimum bandwidth in kilobits per second.
  - **Delay**—Route delay in tens of microseconds
  - **Reliability**— The likelihood of successful packet transmission expressed as a number from 0 to 255, where 255 means that the route is 100 percent reliable
  - **Loading**— The route's effective loading, expressed as a number from 1 to 255, where 255 means that the route is 100 percent loaded
  - **MTU**— Maximum transmission unit, the maximum packet size along the route in bytes (an integer greater than or equal to 1)
17. If you use the **metric** parameter in a **redistribute** command, you can set a different default metric for each protocol being redistributed. A metric configured in a **redistribute** command overrides the value in the **default-metric** command for that one protocol.
18. The **passive-interface** command prevents routing updates for a routing protocol from being sent through a router interface. The **passive-interface default** command sets all router interfaces to passive.
19. Use static routes, possibly in combination with passive interfaces.
20. The **distance** command is used to change the default administrative distance of routes from specific source addresses that are permitted by an access list. The parameters in this command are as follows.

| Parameter                  | Description                                                                                                                              |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| 150                        | Defines the administrative distance that specified routes are assigned                                                                   |
| 0.0.0.0<br>255.255.255.255 | Defines the source address of the router supplying the routing information-in this case, any router                                      |
| 3                          | Defines the access list to be used to filter incoming routing updates to determine which will have their administrative distance changed |

Thus, routes matching access list 3 from any router are assigned an administrative distance of 150.

21. Some applications of route maps are as follows:
  - Route filtering during redistribution
  - PBR
  - NAT
  - BGP
22. All the **match** statements in the **route-map** statement must be considered true for the **route-map** statement to be considered matched.
23. In a single **match** statement that contains multiple conditions, at least one condition in

- the **match** statement must be true for that **match** statement to be considered a match.
24. The **map-tag** parameter in a **route-map** command is the name of the route map.
  25. Apply the **ip policy route-map map-tag** interface configuration command to use a route map for policy-based routing.
  26. Use the following to configure the use of a route map called TESTING when redistributing OSPF 10 traffic into RIP:  

```
router rip
 redistribute ospf 10 route-map TESTING
```
  27. Routes with a tag of 80 are denied from being redistributed from OSPF into RIP.
  28. Options in the **distribute-list** command allow updates to be filtered based on various factors, including the following:
    - Incoming interface
    - Outgoing interface
    - Redistribution from another routing protocol
  29. The **distribute-list out** command filters updates going *out of* the interface or routing protocol specified in the command, *into* the routing process under which it is configured.  
The **distribute-list in** command filters updates going *into* the interface specified in the command, *into* the routing process under which it is configured.
  30. To assign an access list to filter outgoing routing updates, use the **distribute-list {access-list-number | name} out [interface-name | routing-process [routing-process parameter]]** command. This command is entered at the Router(config-router)# prompt.
  31. In a prefix list, **permit** means to allow the route to be used, and **deny** means that the route is not allowed to be used.
  32. The value range is *length < ge-value < le-value <= 32*.
  33. Prefix list sequence numbers are generated automatically, unless you disable this automatic generation. By default, a prefix list's entries have sequence values of 5, 10, 15, and so on.  
If you leave out the sequence number when configuring *all* statements for the same route map name, the router will assume that you are editing (and possibly adding to) the first statement, which defaults to sequence number 10. Route map sequence numbers do not automatically increment. If no other entry is already defined with the supplied **route-map** tag, an entry is created, with the *sequence-number* set to 10. If only one entry is already defined with the supplied **route-map** tag, that entry is the default entry for the **route-map** command, and the *sequence-number* of the entry is unchanged. (The router assumes you are editing the one entry that is already defined.)
  34. The **traceroute EXEC** command can be used to discover the path that a packet takes through a network.

## Chapter 5

1. Resiliency, availability, adaptability, performance, support for network and application services, predictability, and asymmetric traffic
2. Summarization hides addressing details, isolates routing issues, and defines failure domains.
3. A good addressing design, redistribution and other routing protocol characteristics, passive interfaces, distribute lists, prefix lists, administrative distance, route maps, route tagging, offset lists, Cisco IOS IP service level agreements, and policy-based routing. Advanced path control tools include Cisco IOS Optimized Edge Routing, virtualization, and Cisco Wide Area Application Services.

|    |                                                                          |                  |
|----|--------------------------------------------------------------------------|------------------|
| 4. | Statement                                                                | Routing Protocol |
|    | Metric can be changed only for external routes at redistribution points. | OSPF             |
|    | Next hop can be set for all routes under various conditions.             | EIGRP            |
|    | Can be configured only on ABRs and ASBRs.                                | OSPF             |
|    | Unequal-cost load balancing is available.                                | EIGRP            |
|    | All routes can be tagged.                                                | EIGRP            |

5. b, c, d
6. The *access-list-number* or *access-list-name* parameter defines the access list to be applied. Routes permitted by the access list will have the offset added to their metric.
7. The Cisco IOS IP SLAs use active traffic monitoring, generating traffic in a continuous, reliable, and predictable manner, to measure network performance.
8. There are two types of IP SLAs operations: those in which the target device is running the IP SLAs *responder* component (such as a Cisco router), and those in which the target device is not running the IP SLAs responder component (such as a web server or IP host). An IP SLAs responder is a component embedded in a Cisco IOS device that allows that device to anticipate and respond to IP SLAs request packets. A Cisco IOS device can be configured as an IP SLAs responder and will provide accurate measurements without the need for dedicated probes or any complex or per-operation configuration.
9. At the start of the control phase, the IP SLAs source sends a control message with the configured IP SLAs operation information to IP SLAs control port UDP 1967 on the target router (the responder). The control message includes the protocol and port number that will be used during the operation, and duration of the operation.
10. a, c, d
11. To track the reachability of IOS IP SLAs operation number 100 with object number 2 use the following command: **track 2 ip sla 100 reachability** (or **track 2 rtr 100 reachability**).
12. To start IP SLAs operation number 100 immediately and have it never end use the following command: **ip sla schedule 100 life forever start-time now** (or **ip sla monitor schedule 100 life forever start-time now**).
13. Some benefits of policy-based routing (PBR) are source-based transit provider selection, QoS, cost savings, and load sharing.
14. PBR is applied to incoming packets on an interface.
15. a, c, e
16. These **set** commands are evaluated in the following order.  
 set ip next-hop  
 set interface  
 set ip default next-hop  
 set default interface
17. A packet is routed to the next hop specified by the **set default interface** command only if there is *no* explicit route for the packet's destination address in the routing table. A default route in the routing table is not considered an explicit route for an unknown destination address.
18. To identify a route map to use for policy routing on an interface, use the **ip policy route-map map-tag** interface configuration command. Policy based routing is configured on the interface that *receives* the packets, not on the interface from which the packets are sent.  
 To identify a route map to use for local policy routing, use the **ip local policy route-map map-tag** global configuration command. Packets originating on the router are not normally policy routed; this command applies the route map to these packets.
19. This command configures policy routing to verify the reachability of the next hop of a route map before the router performs policy routing to that next hop.
20. Normal routing, using routing protocols, focuses on detecting a routing path using static routing metrics. OER uses tools such as Cisco IOS IP SLAs to automatically detect network service degradation and to make dynamic routing decisions and adjustments based on criteria such as response time, packet loss, jitter, path availability, traffic load distribution, and so forth.
21. VRF tables are virtual routing tables used to separate the routing function by group, on one physical router.

## Chapter 6

1. An interior gateway protocol (IGP) is a routing protocol used to exchange routing information within an autonomous system. An exterior gateway protocol (EGP) is a routing protocol used to exchange routing information between autonomous systems.



2. BGP is an exterior path vector routing protocol.
3. a, d
4. Dual homing is when a customer has two links toward the same ISP. Dual multihoming is when a customer has two links toward each of two ISPs, to improve resiliency.
5. Three common design options for BGP multihoming are as follows:
  - All ISPs pass only default routes to the autonomous system.
  - All ISPs pass default routes and provider-owned specific routes to the autonomous system.
  - All ISPs pass all routes to the autonomous system.
6. Acquiring a partial BGP table from each ISP is beneficial because path selection for specific routes will be more predictable than when using a default route. For example, the ISP that a specific router within the autonomous system uses to reach the networks that are passed into the autonomous system will be based on the BGP path attributes; it usually is the shortest AS-path. If instead only default routes are passed into the autonomous system, the ISP that a specific router within the autonomous system uses to reach any external address is decided by the IGP metric used to reach the default route within the autonomous system.
7. This configuration requires a lot of resources within the autonomous system because it must process all the external routes.
8. No. BGP specifies that a BGP router can advertise to its peers in neighboring autonomous systems only those routes that it itself uses—in other words, its best path.
9. BGP use in an autonomous system is most appropriate when the effects of BGP are well understood and at least one of the following conditions exists:
  - The autonomous system allows packets to transit through it to reach other autonomous systems. (For example, it is a service provider.)
  - The autonomous system has multiple connections to other autonomous systems.
  - Routing policy and route selection for traffic entering and leaving the autonomous system must be manipulated.
10. It is appropriate to use static routes rather than BGP if at least one of the following conditions exists:
  - A single connection to the Internet or another autonomous system.
  - Lack of memory or processor power on edge routers to handle constant BGP updates.
  - You have limited understanding of route filtering and the BGP path-selection process.
11. BGP uses TCP as its transport protocol. Port 179 has been assigned to BGP.
12. The BGP AS-path is guaranteed to always be loop free because a router running BGP does not accept a routing update that already includes its autonomous system number in the path list. Because the update has already passed through its autonomous system, accepting it again would result in a routing loop.
13. Any two routers that have formed a BGP connection are called BGP peer routers or BGP neighbors.
14. IBGP— When BGP is running between routers within one autonomous system, it is called IBGP.  
EBGP— When BGP is running between routers in different autonomous systems, it is called EBGP.  
Well-known attribute— A well-known attribute is one that all BGP implementations must recognize. Well-known attributes are propagated to BGP neighbors.  
Transitive attribute— An optional attribute; a transitive attribute that is not implemented in a router should be passed to other BGP routers untouched and marked as partial.  
BGP synchronization— The BGP synchronization rule states that a BGP router should not use or advertise to an external neighbor a route learned by IBGP unless that route is local or is learned from an IGP. BGP synchronization is disabled by default in Cisco IOS Software Release 12.2(8)T and later; it was on by default in earlier Cisco IOS Software releases.
15. A router running BGP keeps its own table for storing BGP information received from and sent to other routers. This table is separate from the IP routing table in the router. The router offers the best routes from the BGP table to the IP routing table and can be configured to share information between the two tables (by redistribution). BGP also keeps a neighbor table containing a list of neighbors with which it has a BGP connection.
16. The four BGP message types are open, keepalive, update, and notification.
17. The BGP router ID is an IP address assigned to that router and is determined on startup. The BGP router ID is chosen the same way that the OSPF router ID is chosen—it is the highest active IP address on the router, unless a loopback interface with an IP address exists, in which case it is the highest such

loopback IP address. Alternatively, the router ID can be statically configured, overriding the automatic selection.

18. BGP is a state machine that takes a router through the following states with its neighbors:

- Idle
- Connect
- Active
- Open sent
- Open confirm
- Established

Only when the connection is in the established state are update, keepalive, and notification messages exchanged.

19. The following are well-known mandatory attributes:

- AS-path
- Next-hop
- Origin

The following are well-known discretionary attributes:

- Local preference
- Atomic aggregate

The following are optional transitive attributes:

- Aggregator
- Community

The multi-exit-discriminator (MED) is an optional nontransitive attribute.

20. When IBGP advertises a prefix that was originally learned via EBGP, the value of the next-hop attribute is carried from the EBGP update, by default.

21. When running BGP over a multiaccess network, a BGP router uses the appropriate address as the next-hop address to avoid inserting additional hops into the path. The address used is of the router on the multiaccess network that sent the advertisement. On Ethernet networks, that router is accessible to all other routers on the Ethernet. On an NBMA network, however, all routers on the network might not be accessible to each other, so the next-hop address used might be unreachable. This behavior can be overridden by configuring a router to advertise itself as the next-hop address for routes sent to other routers on the NBMA network.

| Attribute        | Order Preferred In | Highest or Lowest Value Preferred? | Sent to Which Other Routers?                                                                                                                                                                           |
|------------------|--------------------|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Local preference | 2                  | Highest                            | Internal BGP neighbors only.                                                                                                                                                                           |
| MED              | 3                  | Lowest                             | External BGP neighbors. Those routers propagate the MED within their autonomous system, and the routers within the autonomous system use the MED, but do not pass it on to the next autonomous system. |
| Weight           | 1                  | Highest                            | Not sent to any BGP neighbors; local to the router only.                                                                                                                                               |

23. BGP synchronization should only be disabled if all routers in the transit path in the autonomous system (in other words, in the path between the BGP border routers) are running full-mesh IBGP or if the autonomous system is not a transit autonomous system.

24. The **neighbor 10.1.1.1 ebgp-multihop** command sets the Time To Live (TTL) value for the EBGP connection to 10.1.1.1 to 255 (by default). This command is necessary if the EBGP neighbor address 10.1.1.1 is not directly connected to this router. An additional parameter for this command allows you to set the TTL to another value.

25. The BGP configuration for Router A is as follows:

RouterA(config)#**router bgp 65000**

```
RouterA(config-router)#neighbor 10.2.2.2 remote-as 65001
RouterA(config-router)#neighbor 10.2.2.2 update-source loopback 0
RouterA(config-router)#neighbor 10.2.2.2 ebgp-multihop 2
```

The BGP configuration for Router B is as follows:

```
RouterB(config)#router bgp 65001
RouterB(config-router)#neighbor 10.1.1.1 remote-as 65000
RouterB(config-router)#neighbor 10.1.1.1 update-source loopback 0
RouterB(config-router)#neighbor 10.1.1.1 ebgp-multihop 2
```

26. Use the **no synchronization** router configuration command to disable BGP synchronization; in current IOS releases it is disabled by default.
27. The **neighbor 10.1.1.1 remote-as 65000** router configuration command would be used.
28. The **neighbor** command tells BGP *where* to advertise. The **network** command tells BGP *what* to advertise.
29. When you configure **network 172.16.0.0 mask 255.255.0.0**, BGP looks for exactly 172.16.0.0/16 in the routing table. In this case it would find 172.16.0.0/24 but will not find 172.16.0.0/16. Because the routing table does not contain a specific match to the network, BGP does not announce the 172.16.0.0/24 network to any neighbors.
30. The **soft out** option of the **clear ip bgp** command allows BGP to do a soft reset for outbound updates. The router on which this command is issued does not reset the BGP session. Instead, the router creates a new update and sends the whole table to the specified neighbors. This update includes withdrawal commands for networks that the other neighbor will not see anymore based on the new outbound policy. (Note that the **soft** keyword of this command is optional; **clear ip bgp 10.1.1.1 out** does a soft reset for outbound updates.)
31. The **show ip bgp neighbors** command is used to display detailed information about BGP connections to neighbors.
32. The > indicates the best path for a route selected by BGP. This route is offered to the IP routing table.
33. The Metric column displays the MED.
34. If the neighbor state is established, the State/PfxRcd column either is blank or has a number in it. The number represents how many BGP network entries have been received from this neighbor.
35. BGP supports message digest 5 (MD5) neighbor authentication. MD5 sends a "message digest" (also called a "hash"), which is created using the key and a message. The message digest is then sent instead of the key. The key itself is not sent, preventing it from being read while it is being transmitted. This ensures that nobody can eavesdrop on the line and learn keys during transmission.
36. BGP path manipulation can affect which traffic uses which Internet connection. For example, all traffic going to a particular IP address or autonomous system can be forced to go out one connection to the Internet, and all other traffic can be routed out the other connection. Depending on the volume of Internet traffic, the bandwidth of these connections may be affected.
37. The first line of the route map called local\_pref is a **permit** statement with a sequence number of 10; this defines the first **route-map** statement. The **match** condition for this statement checks all networks to see which are permitted by access list 65. The route map sets routes to these networks to a local preference of 300.  
The second statement in the route map called local\_pref is a **permit** statement with a sequence number of 20, but it does not have any **match** or **set** statements. Because there are no match conditions for the remaining updates, they are all permitted with their current settings. In this case, the local preference for routes to the remaining networks stays set at the default of 100. This route map is linked to neighbor 192.168.5.3 as an inbound route map. Therefore, as Router A receives updates from 192.168.5.3, it processes them through the local\_pref route map and sets the local preference accordingly as the routes are placed into Router A's BGP forwarding table.
38. 10 Prefer the path with the lowest neighbor BGP router ID  
6 Prefer the lowest MED  
4 Prefer the shortest AS-path  
9 Prefer the oldest route for EBGp paths  
5 Prefer the lowest origin code  
1 Prefer the highest weight

- 8 Prefer the path through the closest IGP neighbor
  - 2 Prefer the highest local preference
  - 3 Prefer the route originated by the local router
  - 11 Prefer the route with the lowest neighbor IP address
  - 7 Prefer the EBGp path over the IBGP path
39. The **neighbor** {*ip-address* | *peer-group-name*} **weight** *weight* router configuration command is used to assign a weight to updates from a BGP neighbor connection.
  40. The **ip as-path access-list** command defines an access list statement that permits or denies a route based on its AS-path attribute.
  41. AS-path prepending is a feature that extends the AS-path attribute of a route with multiple copies of the autonomous system number of the sender. The receiver of this update is less likely to select the path as the best because its AS-path attribute appears to be longer.
  42. This command would permit the 10.1.0.0/16 route because its mask is between /8 and /20; the 10.2.0.0/24 route would not be permitted because its mask is greater than /20.

## Chapter 7

1. Connectivity technologies, resiliency, routing protocols, service mix, security and compliance, mobility
2. Voice, application firewall, intrusion prevention, virtual private network, WAN optimization, wireless, WAN backup
3. The transform sets define how packets will be encapsulated by identifying an acceptable combination of security protocols, algorithms, and other settings to apply to IPsec-protected traffic.

|    |                                                                                   |                 |
|----|-----------------------------------------------------------------------------------|-----------------|
| 4. | Statement                                                                         | IPsec service   |
|    | Provides assurance that the data is exchanged with the rightful party             | Authentication  |
|    | Provides encryption during the exchange of the data                               | Confidentiality |
|    | Provides a check to confirm that the data was not altered during the transmission | Integrity       |

5. b, d
6. It represents the administrative distance. The AD should be set to a greater value than that of the existing dynamic routing protocol.
7. Asymmetric DSL (ADSL) uses a frequency range from approximately 20 KHz to 1 MHz.
8. The classic firewall is based on ACLs and is the more traditional approach to fire-walling in an IOS router. This classic approach is referred to context-based access control (CBAC).  
The zone-based firewall (ZBF) is based on security zones and access, and it is the more recent approach to implementing the service in routers.
9. The router is encrypting and decrypting GRE traffic. GRE is protocol 47.
10. b, d, e, h
11. ip local pool EZVPN 10.254.254.1 10.254.254.254
12. ip route 10.254.254.0 255.255.255.0 209.165.200.2
13. redistribute static
14. ip nat inside source
15. b, d
16.
  1. Allow IPsec traffic.
  2. Define an address pool for connecting clients.
  3. Provide routing services for VPN subnets.
  4. Tune NAT for VPN traffic flows.
  5. Verify IPsec VPN configuration.
17. The user can pull down the Status menu of the VPN client and select Statistics. Then, when the user clicks the Tunnel Details tab, the VPN Client shows IP security information, listing the IPsec statistics for this VPN tunnel to the private network.
18. The **show crypto isakmp sa** command. The **show crypto session** command could also be used.

19. To define to which IP address the traffic will be translated to.
20. To find out the IP address used and ensure that we do not have overlapping IP addresses between the two LANs.

## Chapter 8

1. The issues associated with IPv4, including address depletion, Internet routing table expansion, and the lack of true end-to-end model.
2. IPv6 features include a larger address space, elimination of NAT and broadcast addresses, simplified header for improved router efficiency, support for mobility and security, and transition richness.
3. Most extension headers are not examined or processed by any node other than the node to which the packet is destined. The destination node examines the first extension header (if there is one). The contents of an extension header determine whether or not the node should examine the next header. Therefore, extension headers must be processed in the order they appear in the packet.
4. There are 128 bits in an IPv6 address.
5. IPv6 addresses are written as hexadecimal numbers with colons between each set of four hexadecimal digits (which is 16 bits). The format is  $x:x:x:x:x:x:x$ , where  $x$  is a 16-bit hexadecimal field. (Note that there are seven colons separating eight sections; each section has four hexadecimal digits.)
6. b, c, d
7. For Ethernet, the interface ID used is based on the Media Access Control (MAC) address of the interface and is in an extended unique identifier 64-bit (EUI-64) format. The EUI-64 format interface ID is derived from the 48-bit link-layer MAC address by inserting the 16-bit hexadecimal number FFFE between the upper three bytes (the organizationally unique identifier [OUI] field) and the lower 3 bytes (the vendor code or serial number field) of the link-layer address. The seventh bit in the high-order byte is set to 1 to indicate the uniqueness of the interface ID.
8. IPv6 does not have broadcast addresses.
9. a, d
10. The IPv6 unicast address space encompasses the entire IPv6 address range, with the exception of the FF00::/8 range (addresses starting with binary 1111 1111), which is used for multicast addresses.
11. The IPv6 link-local prefix is FE80::/10.
12. The IPv6 link-local prefix is FEC0::/10.
13. The IPv6 multicast addresses are defined by the prefix FF00::/8.
14. Solicited-node multicast addresses on a link have addresses FF02::1:FFXX:XXXX, where the XX:XXXX is the far right 24 bits of the corresponding unicast or anycast address of the node. Solicited-node multicast addresses are used in neighbor solicitation (NS) messages, which are sent on a local link when a node wants to determine the link-layer address of another node on the same local link, similar to the Address Resolution Protocol (ARP) in IPv4.
15. 2001:00C0::/26
16. The command used to configure an interface with the IPv6 prefix 2001:ABCD::/64, using the interface ID derived from the MAC address is **ipv6 address 2001:ABCD::/64 eui-64**.
17. The **ipv6 address address/prefix-length [eui-64 | link-local]** interface configuration command configures an IPv6 address on an interface. It can be used multiple times to assign multiple addresses.
18. The **ipv6 unnumbered loopback 10** interface configuration command enables IPv6 processing on an interface without assigning an explicit IPv6 address to the interface; the IPv6 address of interface loopback 10 is used as the source address in packets from this interface.
19. A device's link-local address is assigned dynamically by default, using a prefix FE80::/10 and the EUI-64 format interface ID. Such link-local addresses are very easy to configure because you do not have to do anything, but they are very difficult to work with, because you need to know the data link layer address of the interface to determine where it belongs. Link-local addresses can also be statically assigned, using either the EUI-64 format or a manually configured interface ID. Using a manually configured interface ID makes it much easier to remember, and therefore use, the link-local address.
20. After obtaining an address, the DAD phase occurs, in which the device verifies that its new IPv6 address is unique on that link. The device first sends an NS for that address to query if another node on the link has the same IPv6 address. The NS packet is sent to the solicited-node multicast address of its own IPv6 address. The source address of this packet is the unspecified address ::. If a node responds to the request, it means that the IPv6 address is already in use, and the requesting node should not use that address. DAD is used during the autoconfiguration process to ensure that no other device is using the autoconfiguration address.

21. An IPv6 router on a local link can send (either periodically or upon a host's request) network information, such as the 64-bit prefix of the local link network and the default route, to all the nodes on the local link. Hosts can autoconfigure themselves by appending their IPv6 interface identifier (in EUI-64 format) to the local link 64-bit prefix.
22. On broadcast multiaccess connections IPv6 uses the ICMPv6 ND process (NS and NA messages). You can create a static mapping instead between an IPv6 unicast address and a MAC address with the **ipv6 neighbor** command. On point-to-point connections, the same process as broadcast multiaccess is used, using the MAC address from a LAN interface to create the interface ID for link-local and global unicast addresses (unless manually configured). Point-to-multipoint connections over Frame Relay requires a mapping between IPv6 addresses (link-local and global unicast) and DLCI. Link-local mapping and broadcast support on the Frame Relay network is required for routing protocol support.
23. c, d, f
24. This command creates a recursive static route to the 2001:CC1E::/32 prefix via the next-hop address 2001:12::1.
25. No, the **ipv6 rip RIPON enable** command automatically creates the RIPON process if it does not already exist.
26. The similarities between OSPFv3 and OSPFv2 include the following:
  - OSPFv3 uses the same basic packet types as OSPFv2: hello, DBD (also called DDP), LSR, LSU, and LSAck. Some of the fields within the packets have changed.
  - The mechanisms for neighbor discovery and adjacency formation are identical.
  - OSPFv3 operations over NBMA topologies are the same. The RFC-compliant nonbroadcast and point-to-multipoint modes are supported. OSPFv3 also supports the other modes from Cisco such as point-to-point and broadcast.
  - LSA flooding and aging are the same.

27.

| Packet Type | Name  |
|-------------|-------|
| 1           | Hello |
| 2           | DBD   |
| 3           | LSR   |
| 4           | LSU   |
| 5           | LSAck |

28. The multicast addresses used by OSPFv3 are as follows:  
 FF02::5—This address represents all SPF routers on the link-local scope. It is equivalent to 224.0.0.5 in OSPFv2.  
 FF02::6—This address represents all DRs on the link-local scope. It is equivalent to 224.0.0.6 in OSPFv2.
29. The OSPFv3 router ID remains at 32 bits.
30. To summarize routes at an area boundary use the **area area-id range ipv6-prefix/prefix-length [advertise | not-advertise] [cost cost]** router configuration command.
31. A router ID is required for EIGRP for IPv6 configuration. Similar to OSPFv3, the router ID is a 32-bit IPv4 address. In an IPv6-only environment, there are no IPv4 addresses assigned, so the router ID must be configured manually. The **eigrp router-id {ip-address}** router configuration command (or the **router-id {ip-address}** router configuration command) defines a router ID for EIGRP for IPv6. The *ip-address* is a number in IPv4 address format; in other words it is a 32-bit number in dotted-decimal format. The router ID must be unique on each router.
32. EIGRP for IPv6 does not auto summarize like its IPv4 counterpart. Summarization must be configured.
33. In this case, the carrier protocol is IPv4 (it is used to establish sessions) and the passenger protocol is IPv6 (for which MBGP is advertising the IPv6 prefixes).
34. The configuration is
 

```
router bgp 200
 bgp router-id 2.2.2.2
 neighbor 2001:200:2:4::1 remote-as 200
 !
```



```
address-family ipv6
 neighbor 2001:200:2:4::1 activate
 redistribute connected
```

The **router bgp** configuration defines a 32-bit router ID (which must be configured in an IPv6-only network) and establishes the BGP peering session with the neighbor to send and receive advertisements. The peering session is established using IPv6, so IPv6 is the carrier protocol. The **address-family ipv6** command defines the passenger protocol, the protocol that is to be advertised using MBGP. In this example, the passenger protocol is IPv6. All passenger protocols have an address family section within the **router bgp** configuration. The address family section is where routing policies and specific features for the particular address family are defined. In this example, the neighbor is activated (as it must be for each address family) and redistribution is configured.

35. The configuration is
- ```
route-map PBR permit 10
  match ipv6 address SOURCE888
  set ipv6 next-hop 154::5
ipv6 access-list SOURCE888
  permit ipv6 888::/64 any
interface Fa0/0
  ipv6 policy route-map PBR
```

Recall that PBR is configured on the interface that receives the traffic that should be routed differently than the default path. In this example, the traffic received on the Fast Ethernet 0/0 interface. Packets coming in this interface will be examined and if they are from 888::/64, they will be routed via 154::5; otherwise they will be routed normally. The route map PBR sequence 10 is a **permit** entry. It matches the IPv6 addresses permitted by the access list called SOURCE888. If the traffic matches, it is policy routed and processed according to the **set** command, which specifies that the next-hop address is 154::5. The SOURCE888 ACL permits packets with a source address in the 888::/64 prefix, and with any destination address. The route map is applied to the Fast Ethernet interface. Therefore, a packet received on the Fast Ethernet 0/0 interface, from a source address in the 888 network, will match this route map condition, and its next-hop address will be set to 154::5.

36. Multiple RIPng instances can run on the same router and on the same link and by default will send and receive advertisements between each other. Changing the port and or multicast group addresses separates the processes; redistribution must be configured to share information between these separate processes. Seed metrics may need to be configured to control paths.
37. The configuration is
- ```
ipv6 router rip RIPON
 redistribute ospf 2 metric 1
ipv6 router ospf 2
 redistribute rip RIPON
```

The OSPFv3 routes are redistributed into RIPng with the metric set to 1. This is done under the RIPON process. The RIPng routes are redistributed into OSPFv3. This is done under the OSPF 2 process.

38. The **bgp redistribute-internal** address family (or router) configuration command configures IBGP redistribution into an IGP.
39. The techniques to transition from IPv4 to IPv6 are dual stack, tunneling, and translation.
40. Manual tunnels use IPv4 protocol 41; GRE tunnels use IPv4 protocol 47.
41. The **tunnel mode gre ipv6** interface configuration command sets the encapsulation mode for the tunnel interface to use GRE tunneling using IPv6 as the transport protocol.
42. Each 6to4 edge router, which is a dual-stacked device, has an IPv6 address with a /48 prefix, which is the concatenation of 2002::/16 and the hexadecimal representation of the IPv4 address of that edge router. 2002::/16 is a specially assigned address range for the purpose of 6to4 tunneling. The edge routers automatically build the tunnel using the IPv4 addresses that are embedded in the IPv6 addresses. For example, if the IPv4 address of an edge router is 192.168.99.1, the prefix of its IPv6 address is 2002:c0a8:6301::/48, because c0a86301 is the hexadecimal representation of 192.168.99.1.



43. The IPv4-compatible IPv6 address is simply an IPv6 address with 96 bits filled with 0s followed by the 32-bit IPv4 address in the lower 32 bits. The 32-bit portion is even written in dotted-decimal format. In this example, the IPv4 address is 172.17.2.1, so the IPv4-compatible IPv6 address is ::172.17.2.1.
44. The format of an ISATAP tunnel address is a 64-bit prefix concatenated to a 64-bit interface ID in EUI-64 format. Upper 32 bits of the interface ID are 0000:5EFE, a reserved OUI value indicating an IPv6 ISATAP address. The lower (least significant) 32 bits of the interface ID contain the IPv4 address of the interface (written in hexadecimal).
45. The configuration is
- ```
interface S0/0/0
  ipv6 nat
interface S0/1/0
  ipv6 nat

ipv6 nat v6v4 source 14::4 172.16.14.100
ipv6 nat v4v6 source 172.16.14.2 2244::1
ipv6 nat prefix 2244::/96
ipv6 router rip RIPON
  redistribute connected metric 1
```

The **ipv6 nat** command enables NAT-PT on both interfaces. The **ipv6 nat v6v4 source** command is used to configure the mapping between the IPv6 source address (14::4) and the IPv4 address that the IPv6 device appears as in the IPv4 world (172.16.14.100). The **ipv6 nat v4v6 source** command is used to configure the mapping for return traffic, between the IPv4 device's IPv4 source address (172.16.14.2) and the IPv6 address that it appears as in the IPv6 world (2244::1). This IPv6 address does not exist in the IPv6 world. It is an unused address selected to represent IPv4 devices in the IPv6 world. It is on the NAT-PT prefix, which is configured next with the **ipv6 nat prefix** command. This creates a connected route in the router. The **redistribute connected** command (with a seed metric of 1) is entered under the RIPng process so that route is known in the IPv6 world.

46. Packets that match the IPv4 ACL are translated to addresses in the IPv6 pool.
47. Packets that match the IPv6 ACL are translated to addresses in the IPv4 pool.

Appendix B. IPv4 Supplement

This appendix contains job aids and supplementary information that cover the following topics:

- IPv4 Addresses and Subnetting Job Aid
- Decimal-to-Binary Conversion Chart
- IPv4 Addressing Review
- IPv4 Access Lists
- IPv4 Address Planning
- Hierarchical Addressing Using Variable-Length Subnet Masks
- Route Summarization
- Classless Interdomain Routing

This Internet Protocol Version 4 (IPv4) supplement provides job aids and supplementary information intended for your use when working with IPv4 addresses.

Note

In this appendix, the term *IP* refers to IPv4.

This appendix includes an IP addressing and subnetting job aid and a decimal-to-binary conversion chart. The information in the sections "IPv4 Addressing Review" and "IPv4 Access Lists" should serve as a review of the fundamentals of IP addressing and of the concepts and configuration of access lists, respectively.

The remainder of the sections relate to IP address planning. Scalable, well-behaved networks are not accidental. They are the result of good network design and effective implementation planning. A key element for effective scalable network implementation is a well-conceived and scalable IP addressing plan, as

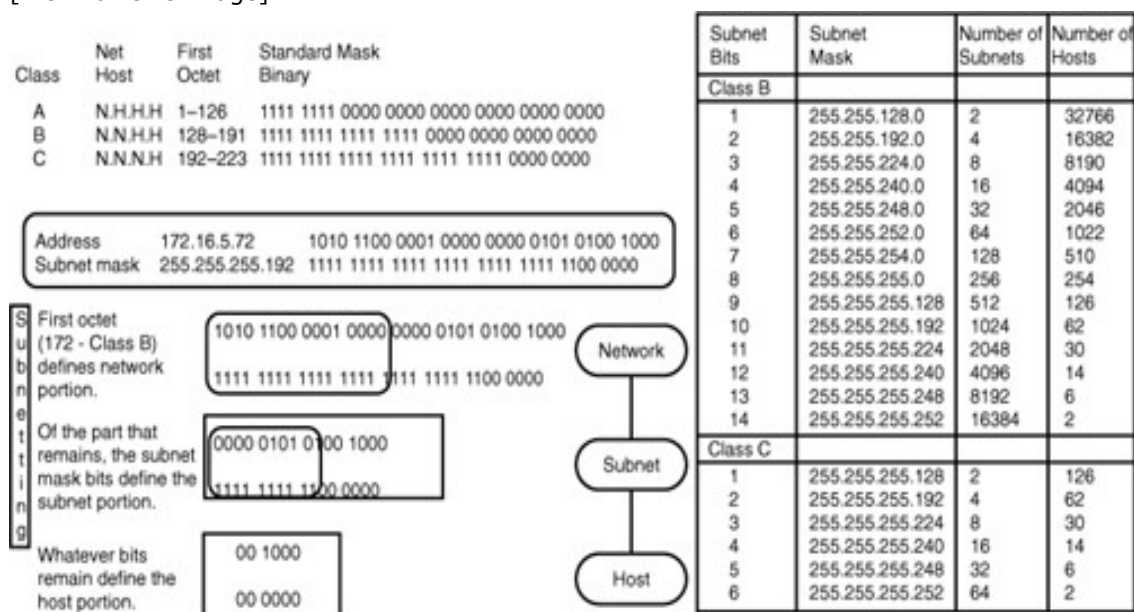
described in the "IPv4 Address Planning" section. Variable-length subnet masking (VLSM), route summarization, and classless interdomain routing (CIDR) are then explored. VLSM allows the network administrator to subnet a previously subnetted address to make the best use of the available address space. Summarization and CIDR are advanced IP addressing techniques that keep the size of the routing tables from increasing as networks grow.

IPv4 Addresses and Subnetting Job Aid

Figure B-1 is a job aid to help you with various aspects of IP addressing, including how to distinguish address classes, the number of subnets and hosts available with various subnet masks, and how to interpret IP addresses.

Figure B-1. IP Addresses and Subnetting Job Aid.

[View full size image]



Decimal-to-Binary Conversion Chart

Table B-1 can be used to convert from decimal to binary and from binary to decimal.

Table B-1. Decimal-to-Binary Conversion Chart

Decimal	Binary	Decimal	Binary	Decimal	Binary
0	00000000	28	00011100	56	00111000
1	00000001	29	00011101	57	00111001
2	00000010	30	00011110	58	00111010
3	00000011	31	00011111	59	00111011
4	00000100	32	00100000	60	00111100
5	00000101	33	00100001	61	00111101
6	00000110	34	00100010	62	00111110
7	00000111	35	00100011	63	00111111
8	00001000	36	00100100	64	01000000
9	00001001	37	00100101	65	01000001
10	00001010	38	00100110	66	01000010
11	00001011	39	00100111	67	01000011
12	00001100	40	00101000	68	01000100

Decimal	Binary	Decimal	Binary	Decimal	Binary
13	00001101	41	00101001	69	01000101
14	00001110	42	00101010	70	01000110
15	00001111	43	00101011	71	01000111
16	00010000	44	00101100	72	01001000
17	00010001	45	00101101	73	01001001
18	00010010	46	00101110	74	01001010
19	00010011	47	00101111	75	01001011
20	00010100	48	00110000	76	01001100
21	00010101	49	00110001	77	01001101
22	00010110	50	00110010	78	01001110
23	00010111	51	00110011	79	01001111
24	00011000	52	00110100	80	01010000
25	00011001	53	00110101	81	01010001
26	00011010	54	00110110	82	01010010
27	00011011	55	00110111	83	01010011
84	01010100	112	01110000	140	10001100
85	01010101	113	01110001	141	10001101
86	01010110	114	01110010	142	10001110
87	01010111	115	01110011	143	10001111
88	01011000	116	01110100	144	10010000
89	01011001	117	01110101	145	10010001
90	01011010	118	01110110	146	10010010
91	01011011	119	01110111	147	10010011
92	01011100	120	01111000	148	10010100
93	01011101	121	01111001	149	10010101
94	01011110	122	01111010	150	10010110
95	01011111	123	01111011	151	10010111
96	01100000	124	01111100	152	10011000
97	01100001	125	01111101	153	10011001
98	01100010	126	01111110	154	10011010
99	01100011	127	01111111	155	10011011
100	01100100	128	10000000	156	10011100
101	01100101	129	10000001	157	10011101
102	01100110	130	10000010	158	10011110
103	01100111	131	10000011	159	10011111
104	01101000	132	10000100	160	10100000
105	01101001	133	10000101	161	10100001
106	01101010	134	10000110	162	10100010

Decimal	Binary	Decimal	Binary	Decimal	Binary
107	01101011	135	10000111	163	10100011
108	01101100	136	10001000	164	10100100
109	01101101	137	10001001	165	10100101
110	01101110	138	10001010	166	10100110
111	01101111	139	10001011	167	10100111
168	10101000	196	11000100	224	11100000
169	10101001	197	11000101	225	11100001
170	10101010	198	11000110	226	11100010
171	10101011	199	11000111	227	11100011
172	10101100	200	11001000	228	11100100
173	10101101	201	11001001	229	11100101
174	10101110	202	11001010	230	11100110
175	10101111	203	11001011	231	11100111
176	10110000	204	11001100	232	11101000
177	10110001	205	11001101	233	11101001
178	10110010	206	11001110	234	11101010
179	10110011	207	11001111	235	11101011
180	10110100	208	11010000	236	11101100
181	10110101	209	11010001	237	11101101
182	10110110	210	11010010	238	11101110
183	10110111	211	11010011	239	11101111
184	10111000	212	11010100	240	11110000
185	10111001	213	11010101	241	11110001
186	10111010	214	11010110	242	11110010
187	10111011	215	11010111	243	11110011
188	10111100	216	11011000	244	11110100
189	10111101	217	11011001	245	11110101
190	10111110	218	11011010	246	11110110
191	10111111	219	11011011	247	11110111
192	11000000	220	11011100	248	11111000
193	11000001	221	11011101	249	11111001
194	11000010	222	11011110	250	11111010
195	11000011	223	11011111	251	11111011
252	11111100	254	11111110		
253	11111101	255	11111111		

IPv4 Addressing Review

This section reviews the basics of IPv4 addresses:

Converting IP addresses between decimal and binary

Determining an IP address class

Private addresses

Extending an IP classful address using subnet masks

Calculating a subnet mask

Calculating the networks for a subnet mask

Using prefixes to represent a subnet mask

Converting IP Addresses Between Decimal and Binary

An *IP address* is a 32-bit, two-level hierarchical number. It is hierarchical because the first portion of the address represents the network, and the second portion of the address represents the node (or host).

The 32 bits are grouped into 4 octets, with 8 bits per octet. The value of each octet ranges from 0 to 255 decimal, or 00000000 to 11111111 binary. IP addresses are usually written in dotted-decimal notation, which means that each octet is written in decimal notation and dots are placed between the octets. Figure B-2 shows how you convert an octet of an IP address in binary to decimal notation.

Figure B-2. Converting an Octet of an IP Address from Binary to Decimal.

Value for Each Bit

2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
128	64	32	16	8	4	2	1

Converting From Binary to Decimal

0	1	0	0	0	0	0	1
128	64	32	16	8	4	2	1
0 + 64 + 0 + 0 + 0 + 0 + 0 + 1 = 65							

It is important that you understand how this conversion is done because it is used when calculating subnet masks, a topic discussed later in this section.

Figure B-3 shows three examples of converting IP addresses between binary and decimal.

Figure B-3. Converting IP Addresses Between Binary and Decimal.

Binary

Address: 00001010.00000001.00010111.00010011

Decimal

Address: 10 . 1 . 23 . 19

Binary

Address: 10101100.00010010.01000001.10101010

Decimal

Address: 172 . 18 . 65 . 170

Binary

Address: 11000000.10101000.00001110.00000110

Decimal

Address: 192 . 168 . 14 . 6

Now that you understand the decimal-to-binary and binary-to-decimal conversion processes, use the following sections to review address classes and the uses of subnet masks.

Determining an IP Address Class

To accommodate large and small networks, the 32-bit IP addresses are segregated into Classes A through E. The first few bits of the first octet determine the class of an address. This then determines how many network bits and host bits are in the address. Figure B-4 illustrates the bits for Class A, B, and C addresses. Each address class allows for a certain number of network addresses and a certain number of host addresses

within a network. Table B-2 shows the address range, the number of networks, and the number of hosts for each of the classes. (Note that Class D and E addresses are used for purposes other than addressing hosts.) Figure B-4. Determining an IP Address Class from the First Few Bits of an Address.

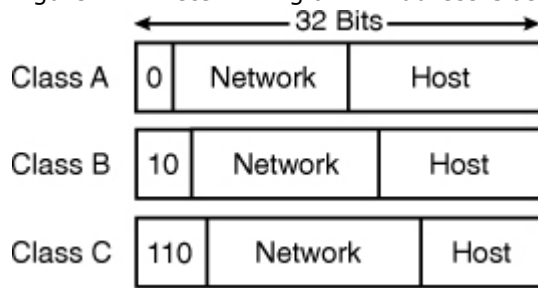


Table B-2. IP Address Classes

Class	Address Range	Number of Networks	Number of Hosts
A[1]	1.0.0.0 to 126.0.0.0	126 ($2^7 - 2$ that are reserved)	16,777,214
B	128.0.0.0 to 191.255.0.0	16,386 (2^{14})	65,532
C	192.0.0.0 to 223.255.255.0	Approximately 2 million (2^{21})	254
D	224.0.0.0 to 239.255.255.255	Reserved for multicast addresses	—
E	240.0.0.0 to 254.255.255.255	Reserved for research	—

[1] The network 127.0.0.0 (any address starting with decimal 127) is reserved for loopback. Network 0.0.0.0 is also reserved and cannot be used to address devices.

Using classes to denote which portion of the address represents the network number and which portion represents the node or host address is called classful addressing. Several issues must be addressed with classful addressing. First, the number of available Class A, B, and C addresses is finite. Another problem is that not all classes are useful for a midsize organization, as illustrated in Table B-2. As can be expected, the Class B range best accommodates a majority of today's organizational network topologies. Subnet masks, as described later in this appendix, in the "Extending an IP Classful Address Using Subnet Masks" section, were introduced to maximize the use of the IP addresses an organization receives, regardless of the class.

Private Addresses

Requests For Comments (RFC) 1918, *Address Allocation for Private Internets*, has set aside the following IPv4 address space for private use:

- Class A network**—10.0.0.0 to 10.255.255.255
- Class B network**—172.16.0.0 to 172.31.255.255
- Class C network**—192.168.0.0 to 192.168.255.255

Note

RFCs are available at <http://www.rfc-editor.org/rfcsearch.html>.

Private addresses are reserved IPv4 addresses to be used only internally within a company's network. These private addresses are not to be used on the Internet, so they must be mapped to a company's external registered address when the company sends anything to a recipient on the Internet.

Note

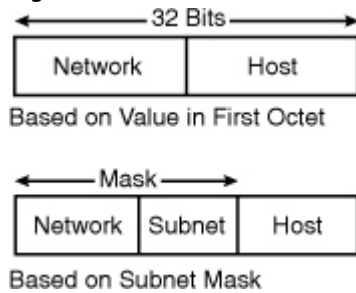
The examples in this book use only private addressing.

Extending an IP Classful Address Using Subnet Masks

RFC 950, *Internet Standard Subnetting Procedure*, was written to address the IP address shortage. It proposed a procedure, called *subnet masking*, for dividing Class A, B, and C addresses into smaller pieces, thereby increasing the number of possible networks.

A subnet mask is a 32-bit value that identifies which address bits represent network bits and which represent host bits. In other words, the router does not determine the network portion of the address by looking at the value of the first octet. Instead, it looks at the subnet mask that is associated with the address. In this way, subnet masks let you extend the usage of an IP address. This is one way of making an IP address a three-level hierarchy, as shown in Figure B-5.

Figure B-5. A Subnet Mask Determines How an IP Address Is Interpreted.



To create a subnet mask for an address, use a binary 1 for each bit that you want to represent the network or subnet portion of the address, and use a binary 0 for each bit that you want to represent the node portion of the address. Note that the 1s in the mask are contiguous. The default subnet masks for Class A, B, and C addresses are as shown Table B-3.

Table B-3. IP Address Default Subnet Masks

Class	Default Mask in Binary	Default Mask in Decimal
A	11111111.00000000.00000000.00000000	255.0.0.0
B	11111111.11111111.00000000.00000000	255.255.0.0
C	11111111.11111111.11111111.00000000	255.255.255.0

Calculating a Subnet Mask

When contiguous 1s are added to the default mask, making the all-1s field in the mask longer, the definition of the network part of an IP address is extended to include subnets. However, adding bits to the network part of an address decreases the number of bits in the host part. Thus, creating additional networks (subnets) is done at the expense of the number of host devices that can occupy each network segment.

The number of subnetworks created is calculated by the formula 2^s , where s is the number of bits by which the default mask was extended.

Note

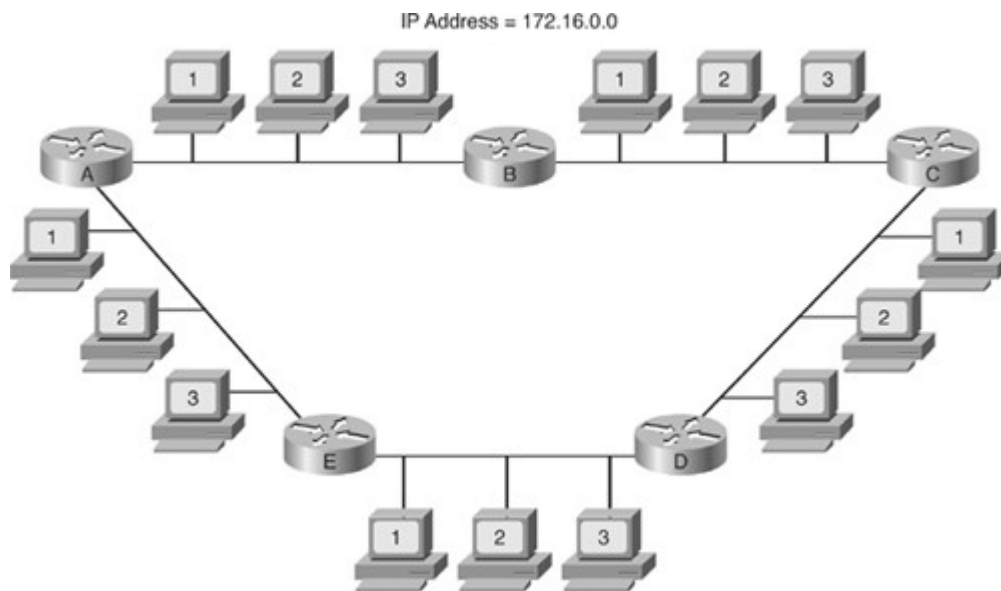
Subnet 0 (where all the subnet bits are 0) must be explicitly allowed using the **ip subnet-zero** global configuration command in Cisco IOS releases earlier than 12.0. In Cisco IOS Release 12.0 and later, this command is enabled by default.

The number of hosts available is calculated by the formula $2^h - 2$, where h is the number of bits in the host portion. The two addresses subtracted in this host formula are for the addresses with all 0s and all 1s in the host field. In the host field, the all-0s bit pattern is reserved as the subnet identifier (sometimes called *the wire*), and the all-1s bit pattern is reserved as a directed broadcast address, to reach all hosts on that subnet.

Because subnet masks extend the number of network addresses you can use by using bits from the host portion, you do not want to randomly decide how many additional bits to use for the network portion. Instead, you want to do some research to determine how many network addresses you need to derive from your given IP address. For example, suppose you have the IP address 172.16.0.0, and you want to configure the network shown in Figure B-6. To establish your subnet mask, do the following:

Figure B-6. Network Used in the Subnet Mask Example.

[View full size image]



- Step 1. Determine the number of networks (subnets) needed. Figure B-6, for example, has five networks.
- Step 2. Determine how many nodes per subnet must be defined. This example has five nodes (two routers and three workstations) on each subnet.
- Step 3. Determine future network and node requirements. For example, assume 100 percent growth.
- Step 4. Given the information gathered in Steps 1 to 3, determine the total number of subnets required. For this example, ten subnets are required. See the earlier section "IPv4 Addresses and Subnetting Job Aid" to select the appropriate subnet mask value that can accommodate 10 networks.

No mask accommodates exactly 10 subnets. Depending on your network growth trends, you might select 4 subnet bits, resulting in a subnet mask of 255.255.240.0. The binary representation of this subnet mask is as follows:

```
11111111.11111111.11110000.00000000
```

The additional 4 subnet bits would result in $2^5 = 2^4 = 16$ subnets.

Calculating the Networks for a Subnet Mask

See Figure B-6. After you identify your subnet mask, you must calculate the ten subnetted network addresses to use with 172.16.0.0 255.255.240.0. One way to do this is as follows:

- Step 1. Write the subnetted address in binary format, as shown at the top of Figure B-7. If necessary, use the decimal-to-binary conversion chart provided in Table B-1.
Figure B-7. Calculating the Subnets Shown in Figure B-6.
[View full size image]

Assigned Address: 172.16.0.0/16

In Binary 10101100.00010000.00000000.00000000

Subnetted Address: 172.16.0.0/20

In Binary 10101100.00010000.00000000.00000000

1 st Subnet:	10101100	.	00010000	.	0000	0000.00000000	=172.16.0.0
2 nd Subnet:	172	.	16	.	0001	0000.00000000	=172.16.16.0
3 rd Subnet:	172	.	16	.	0010	0000.00000000	=172.16.32.0
4 th Subnet:	172	.	16	.	0011	0000.00000000	=172.16.48.0
.							
.							
10 th Subnet:	172	.	16	.	1001	0000.00000000	=172.16.144.0
	Network		Subnet		Host		

Step 2. On the binary address, draw a line between the 16th and 17th bits, as shown in Figure B-7. This is the transition point between the network bits and the subnet bits. Then draw a line between the 20th and 21st bits. This is the transition point between the subnet bits and the host bits, and is the transition point between 1s and 0s in the subnet mask. Now you can focus on the target subnet bits.

Step 3. Historically, it was recommended that you begin choosing subnets from highest (from the far left bit) to lowest, so that you could leave bits available in case you need more host bits later on. However, this strategy does not allow you to adequately summarize subnet addresses, so the present recommendation is to choose subnets from lowest to highest (right to left).

When you calculate the subnet address, all host bits are set to 0. Therefore, for the first subnet, the subnet bits are 0000, and the rest of this third octet (all host bits) is 0000. To convert back to decimal, it is important to note that you must always convert an entire octet, 8 bits.

If necessary, use the decimal-to-binary conversion chart provided in Table B-1, and locate this first number. The third octet of the first subnet number is 00000000, or decimal 0. Do not forget the other 8 host bits in the fourth octet. This fourth octet is also 00000000, or decimal 0.

Step 4. (Optional) List each subnet in binary form to reduce the number of errors. This way, you will not forget where you left off in your subnet address selection.

Step 5. Calculate the second-lowest subnet number. In this case, it is 0001. When combined with the next 4 bits (the host bits) of 0000, this is binary 00010000, or decimal 16. Again, don't forget the other 8 host bits in the fourth octet. This fourth octet is again 00000000, or decimal 0.

Step 6. Continue calculating subnet numbers until you have as many as you need—in this case, 10 subnets, as shown in Figure B-7.

Using Prefixes to Represent a Subnet Mask

As discussed, subnet masks identify the number of bits in an address that represent the network, subnet, and host portions of the address. Another way of indicating this information is to use a *prefix*. A prefix is a slash (/) followed by a numeric value that is the number of bits in the network and subnet portion of the address. In other words, it is the number of contiguous 1s in the subnet mask. For example, assume you are using a subnet mask of 255.255.255.0. The binary representation of this mask is 11111111.11111111.11111111.00000000, which is 24 1s followed by eight 0s. Thus, the prefix is /24, for the 24 bits of network and subnet information, the number of 1s in the mask.

Table B-4 shows some examples of the different ways you can represent a prefix and subnet mask.

Table B-4. Representing Subnet Masks.

IP Address/Prefix	Subnet Mask in Decimal	Subnet Mask in Binary
192.168.112.0/21	255.255.248.0	11111111.11111111.11110000.00000000

IP Address/Prefix	Subnet Mask in Decimal	Subnet Mask in Binary
172.16.0.0/16	255.255.0.0	11111111.11111111.00000000.00000000
10.1.1.0/27	255.255.255.224	11111111.11111111.11111111.11100000

It is important to know how to write subnet masks and prefixes because Cisco routers use both, as shown in Example B-1. You will typically be asked to input a subnet mask when configuring an IP address, but the output generated using show commands typically displays an IP address with a prefix.

Example B-1. Examples of Subnet Mask and Prefix Use on Cisco Routers

```
p1r3#show run
<Output Omitted>
interface Ethernet0
 ip address 10.64.4.1 255.255.255.0
!
interface Serial0
 ip address 10.1.3.2 255.255.255.0
<Output Omitted>

p1r3#show interface ethernet0
Ethernet0 is administratively down, line protocol is down
 Hardware is Lance, address is 00e0.b05a.d504 (bia 00e0.b05a.d504)
 Internet address is 10.64.4.1/24
<Output Omitted>

p1r3#show interface serial0
Serial0 is down, line protocol is down
 Hardware is HD64570
 Internet address is 10.1.3.2/24
<Output Omitted>
```

IPv4 Access Lists

This section reviews IPv4 access lists. It includes the following topics:

- IP access list overview
- IP standard access lists
- IP extended access lists

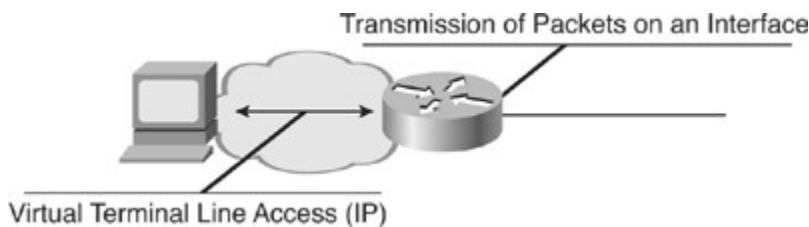
Restricting virtual terminal access

Verifying access list configuration

IP Access List Overview

Packet filtering helps control packet movement through the network, as shown in [Figure B-8](#). Such control can help limit network traffic and restrict network use by certain users or devices. To permit packets to cross or deny packets from crossing specified router interfaces, Cisco provides access lists. An IP access list is a sequential collection of permit and deny conditions that apply to IP addresses or upper-layer IP protocols. IP access lists identify traffic, and can be used for many applications, including filtering packets coming into or going out of an interface, or restricting packets to and from virtual terminal lines.

Figure B-8. Access Lists Control Packet Movement Through a Network.



[Table B-5](#) shows the available types of IP access lists on a Cisco router and their access list numbers. Named access lists are also available for IP.

Table B-5. IP Access List Numbers

Type of Access List	Range of Access List Numbers
IP standard	1 to 99 or from 1300 to 1999
IP extended	100 to 199 or from 2000 to 2699

This section covers IP standard and extended access lists. For information on other types of access lists, see the technical documentation on the Cisco website at <http://www.cisco.com>.

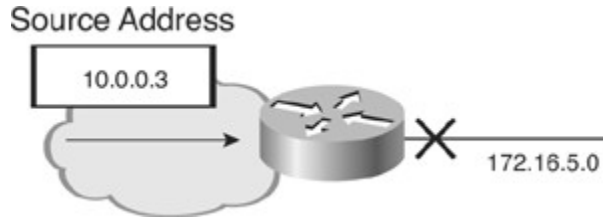
Warning

Cisco IOS Release 10.3 introduced substantial additions to IP access lists. These extensions are backward compatible. Migrating from older releases to the Cisco IOS Release 10.3 or a later image will convert your access lists automatically. However, earlier releases are not upwardly compatible with these changes. Therefore, if you save an access list with the Cisco IOS Release 10.3 or a later image and then use older software, the resulting access list will not be interpreted correctly. This incompatibility can cause security problems. Save your old configuration file before booting Cisco IOS Release 10.3 (or later) images in case you need to revert to an earlier version.

IP Standard Access Lists

Standard access lists permit or deny packets based only on the packet's source IP address, as shown in [Figure B-9](#). The access list number range for standard IP access lists is 1 to 99 or from 1300 to 1999. Standard access lists are easier to configure than their more robust counterparts, extended access lists, but do not provide the granularity available with extended access lists.

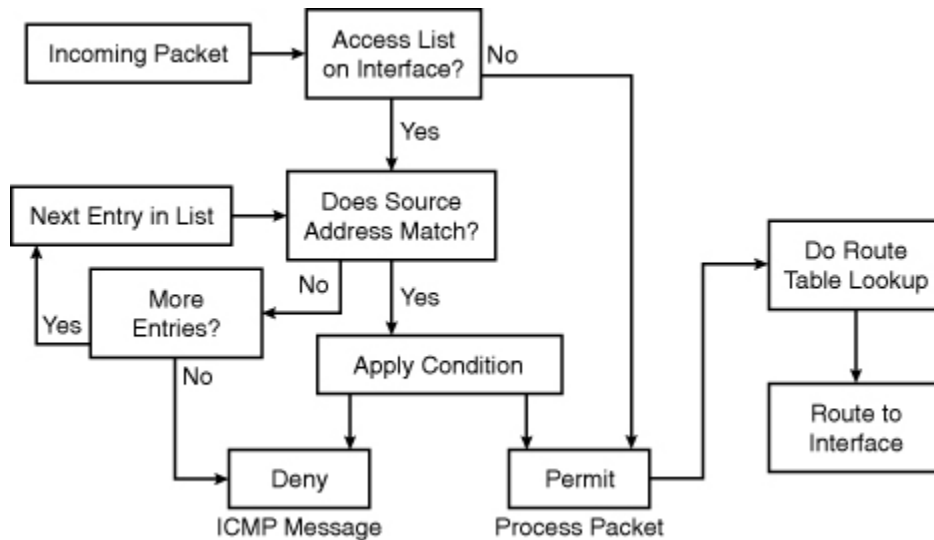
Figure B-9. Standard IP Access Lists Filter Based Only on the Source Address.



A standard access list is a sequential collection of permit and deny conditions that apply to source IP addresses. The router tests addresses against the conditions in an access list one by one. The first match determines whether the router permits or denies the packet. Because the router stops testing conditions after the first match, the order of the conditions is critical. If no conditions match, the router rejects the packet.

Figure B-10 shows the processing of inbound standard access lists. After receiving a packet, the router checks the packet's source address against the access list. If the access list permits the address, the router exits the access list and continues to process the packet. If the access list rejects the address, the router discards the packet and returns an Internet Control Message Protocol (ICMP) administratively prohibited message.

Figure B-10. Inbound Standard IP Access List Processing.

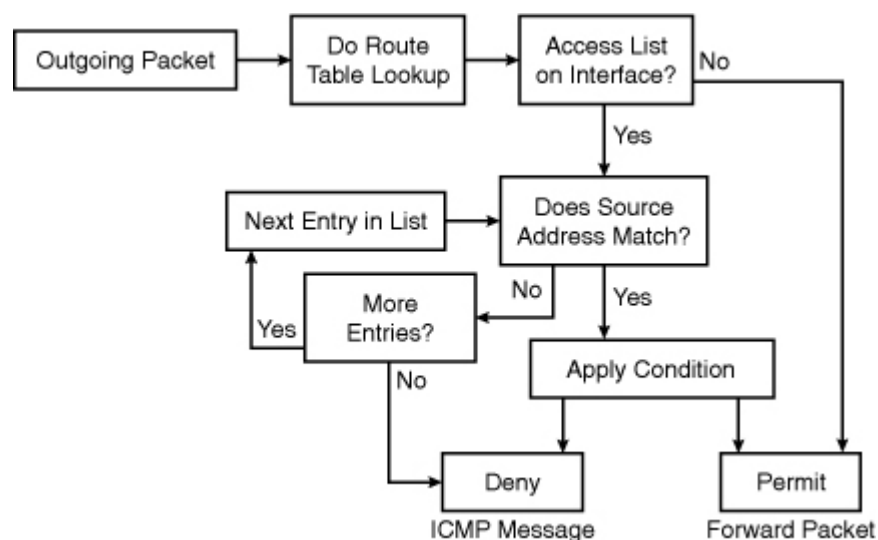


Note that the action taken if no more entries are found in the access list is to deny the packet. This illustrates an important rule to remember when creating access lists: The last entry in an access list is known as an *implicit deny any*; all traffic not explicitly permitted is implicitly denied. For example, consider what will happen if you create a list that just denies traffic that you do not want to let into your network, and you configure this on an interface. If you forget about this rule, *all* of your traffic is denied—the traffic explicitly denied by your list, and the rest of the traffic that is implicitly denied because the access list is applied to the interface.

Another important point to remember when configuring access lists is that order is important. Make sure that you list the entries in order, from specific to general. For example, if you want to deny a specific host address and permit all other addresses, make sure that your entry about the specific host appears first.

Figure B-11 illustrates the processing of outbound standard IP access lists. After receiving and routing a packet to a controlled interface, the router checks the packet's source address against the access list. If the access list permits the address, the router sends the packet. If the access list denies the address, the router discards the packet and returns an ICMP administratively prohibited message.

Figure B-11. Outbound Standard IP Access List Processing.



Wildcard Masks

Both standard and extended IP access lists use a wildcard mask. Like an IP address, a *wildcard mask* is a 32-bit quantity written in dotted-decimal format. The wildcard mask tells the router which bits of the address to use in comparisons:

Address bits corresponding to wildcard mask bits set to 1 are ignored in comparisons.

Address bits corresponding to wildcard mask bits set to 0 are used in comparisons.

An alternative way to think of the wildcard mask is as follows. If a 0 bit appears in the wildcard mask, the corresponding bit location in the access list address and the same bit location in the packet address must match (both must be 0 or both must be 1). If a 1 bit appears in the wildcard mask, the corresponding bit location in the packet matches (whether it is 0 or 1), and that bit location in the access list address is ignored. For this reason, bits set to 1 in the wildcard mask are sometimes called *don't care bits*.

Remember that the order of the access list statements is important because the access list is not processed further after a match is found.

Wildcard Masks

The concept of a wildcard mask is similar to the wildcard character used in DOS-based computers. For example, to delete all files on your computer that begin with the letter *f*, you would enter this:

delete f*.*

The ***** character is the wildcard. Any files that start with *f*, followed by any other characters, and then a dot, and then any other characters, are deleted.

Instead of using wildcard characters, routers use wildcard masks to implement this concept.

Examples of addresses and wildcard masks, and what they match, are shown in [Table B-6](#).

Table B-6. Access List Wildcard Mask Examples

Address	Wildcard Mask	What It Matches
0.0.0.0	255.255.255.255	Any address
172.16.0.0/16	0.0.255.255	Any host on network 172.16.0.0
172.16.7.11/16	0.0.0.0	Host address 172.16.7.11
255.255.255.255	0.0.0.0	Local broadcast address 255.255.255.255
172.16.8.0/21	0.0.7.255	Any host on subnet 172.16.8.0/21

Access List Configuration Tasks

Whether you are creating a standard or extended access list, you need to complete the following two tasks:

- Step 1.** Create an access list in global configuration mode by specifying an access list number and access conditions.

Define a standard IP access list using a source address and wildcard, as shown later in this section.

Define an extended access list using source and destination addresses, and optional protocol-type information for finer granularity of control, as discussed in the “[IP Extended Access Lists](#)” section, later in this appendix.

- Step 2.** Apply the access list in interface configuration mode to interfaces (or in line configuration mode to terminal lines).

After creating an access list, you can apply it to one or more interfaces. Access lists can be applied either outbound or inbound on interfaces.

IP Standard Access List Configuration

Use the **access-list** *access-list-number* {**permit** | **deny**} {*source* [*source-wildcard*] | **any**} [**log**] global configuration command to create an entry in a standard access list, as detailed in [Table B-7](#).

Table B-7. Standard IP *access-list* Command Description

Parameter	Description
<i>access-list-number</i>	Identifies the list to which the entry belongs. A number from 1 to 99 or from 1300 to 1999.
permit deny	Indicates whether this entry allows or blocks traffic from the specified address.
<i>source</i>	Identifies the source IP address.
<i>source-wildcard</i>	(Optional) Identifies which bits in the address field must match. A 1 in any bit position indicates don't care bits, and a 0 in any bit position indicates that the bit must strictly match. If this field is omitted, the wildcard mask 0.0.0.0 is assumed.
any	Use this keyword as an abbreviation for a source and source wildcard of 0.0.0.0 255.255.255.255.
log	(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. Exercise caution when using this keyword, because it consumes CPU

Parameter	Description
	cycles. The message is generated for the first packet that matches, and then at 5-minute intervals, including the number of packets permitted or denied in the prior 5-minute interval.

When a packet does not match any of the configured lines in an access list, the packet is denied by default because there is an invisible line at the end of the access list that is equivalent to **deny any**. (**deny any** is the same as denying an address of 0.0.0.0 with a wildcard mask of 255.255.255.255.)

The keyword **host** can also be used in an access list. It causes the address that immediately follows it to be treated as if it were specified with a mask of 0.0.0.0. For example, configuring **host 10.1.1.1** in an access list is equivalent to configuring **10.1.1.1 0.0.0.0**.

Use the **ip access-group access-list-number {in | out}** interface configuration command to link an existing access list to an interface, as shown in Table B-8. Each interface can have both an inbound and an outbound IP access list.

Table B-8. ip access-group Command Description

Parameter	Description
<i>access-list-number</i>	Indicates the number of the access list to be linked to this interface.
in out	Processes packets arriving on or leaving from this interface. The default is out .

Eliminate the entire list by entering the **no access-list access-list-number** global configuration command. Remove an access list from an interface with the **no ip access-group access-list-number {in | out}** interface configuration command.

Implicit Wildcard Masks

Implicit, or default, wildcard masks reduce typing and simplify configuration, but you must take care when relying on the default mask.

The access list line shown in Example B-2 is an example of a specific host configuration. For standard access lists, if no wildcard mask is specified, the wildcard mask is assumed to be 0.0.0.0. The implicit mask makes it easier to enter a large number of individual addresses.

Example B-2. Standard Access List Using the Default Wildcard Mask

```
access-list 1 permit 172.16.5.17
```

Example B-3 shows common errors found in access list lines.

Example B-3. Common Errors Found in Access Lists

```
access-list 1 permit 0.0.0.0
access-list 2 permit 172.16.0.0
access-list 3 deny any
access-list 3 deny 0.0.0.0 255.255.255.255
```

The first list in Example B-3—**permit 0.0.0.0**—would exactly match the address 0.0.0.0 and then permit it. Because you would never receive a packet from 0.0.0.0, this list would prevent all traffic from getting through (because of the implicit **deny any** at the end of the list).

The second list in [Example B-3](#)—**permit 172.16.0.0**—is probably a configuration error. The intention was probably 172.16.0.0 0.0.255.255. The exact address 172.16.0.0 refers to the network and would never be assigned to a host. As a result, nothing would get through with this list, again because of the implicit **deny any** at the end of the list. To filter networks or subnets, use an explicit wildcard mask.

The next two lines in [Example B-3](#)—**deny any** and **deny 0.0.0.0 255.255.255.255**—are unnecessary to configure because they duplicate the function of the implicit deny that occurs when a packet fails to match all the configured lines in an access list. Although they are not necessary, you might want to add one of these entries for record-keeping purposes.

Configuration Principles

The following general principles help ensure that the access lists you create have the intended results:

Top-down processing

Organize your access list so that more specific references in a network or subnet appear before more general ones.

Place more frequently occurring conditions before less-frequent conditions.

Implicit **deny any**

Unless you end your access list with an explicit **permit any**, it denies all traffic that fails to match any of the access list lines by default.

New lines added to the end by default

Subsequent additions are always added to the end of the access list by default.

Cisco IOS Release 12.2(14)S introduced a feature called IP Access List Entry Sequence Numbering that enables network administrators to apply sequence numbers to **permit** or **deny** statements in a named IP access list and also reorder, add, or remove such statements. Before this feature, network administrators could only add access list entries to the end of an access list (which is the case for numbered access lists), meaning that if statements need to be added anywhere except the end of the access list, the entire access list must be reconfigured. You can selectively add or remove lines when using numbered access lists only by editing the numbered access list as though it is a named access list with the name equal to the number and using the sequence numbers that are automatically assigned to the lines.

An undefined access list equals **permit any**

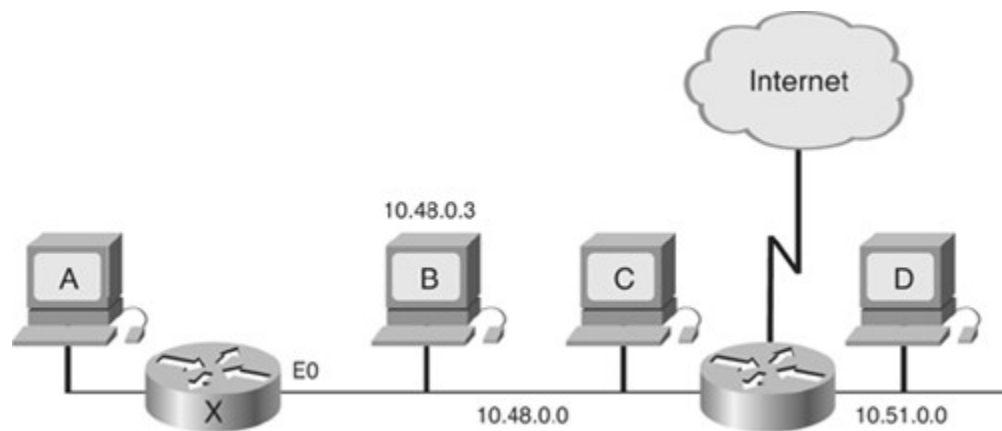
If you apply an access list with the **ip access-group** command to an interface before any access list lines have been created, the result is **permit any**. However, the list is live, so if you enter only one line, it goes from a **permit any** to a **deny most** (because of the implicit **deny any**) as soon as you press **Enter**. For this reason, you should create your access list before applying it to an interface.

Standard Access List Example

[Figure B-12](#) shows a sample network, and [Example B-4](#) shows the configuration on Router X in that figure.

Figure B-12. Network Used for the Standard IP Access List Example.

[\[View full size image\]](#)



Example B-4. Standard Access List Configuration of Router X in Figure B-12

```
Router(config)#access-list 2 permit 10.48.0.3
Router(config)#access-list 2 deny 10.48.0.0 0.0.255.255
Router(config)#access-list 2 permit 10.0.0.0 0.255.255.255
Router(config)#!(Note: all other access implicitly denied)
Router(config)#interface ethernet 0
Router(config-if)#ip access-group 2 in
```

Consider which devices can communicate with Host A in this example:

Host B can communicate with Host A. It is permitted by the first line of the access list, which uses an implicit host mask.

Host C cannot communicate with Host A. Host C is in the subnet that is denied by the second line in the access list.

Host D can communicate with Host A. Host D is on a subnet that is explicitly permitted by the third line of the access list.

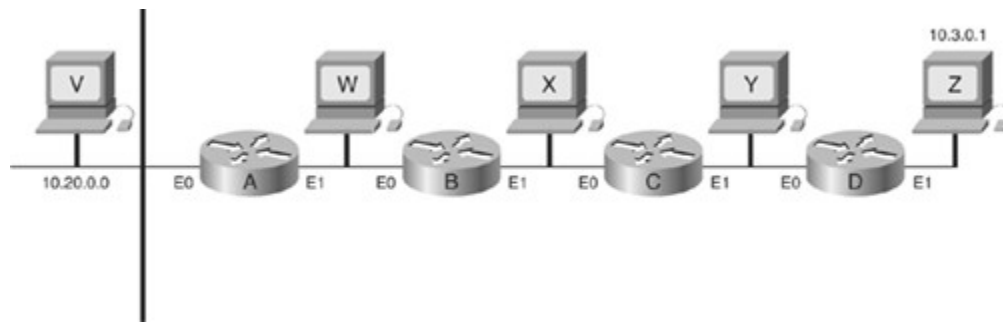
Users on the Internet cannot communicate with Host A. Users outside this network are not explicitly permitted, so they are denied by default with the implicit **deny any** at the end of the access list.

Location of Standard Access Lists

Access list location can be more of an art than a science. Consider the network in Figure B-13 and the access list configuration in Example B-5 to illustrate some general guidelines. If the policy goal is to deny Host Z access to Host V on another network, and not to change any other access policy, determine on which interface of which router this access list should be configured.

Figure B-13. Location of the Standard IP Access List Example.

[\[View full size image\]](#)



Example B-5. Standard Access List to Be Configured on a Router in Figure B-13

```
access-list 3 deny 10.3.0.1
access-list 3 permit any
```

The access list should be placed on Router A because a standard access list can specify only a source address. No hosts beyond the point in the path where the traffic is denied can connect.

The access list could be configured as an outbound list on E0 of Router A. However, it would most likely be configured as an inbound list on E1 so that packets to be denied would not have to be routed through Router A first.

Consider the effect of placing the access list on other routers:

Router B— Host Z could not connect with Host W (and Host V).

Router C— Host Z could not connect with Hosts W and X (and Host V).

Router D— Host Z could not connect with Hosts W, X, and Y (and Host V).

Therefore, for standard access lists, the rule is to place them as close to the *destination* as possible to exercise the most control. Note, however, that this means that traffic is routed through the network, only to be denied close to its destination.

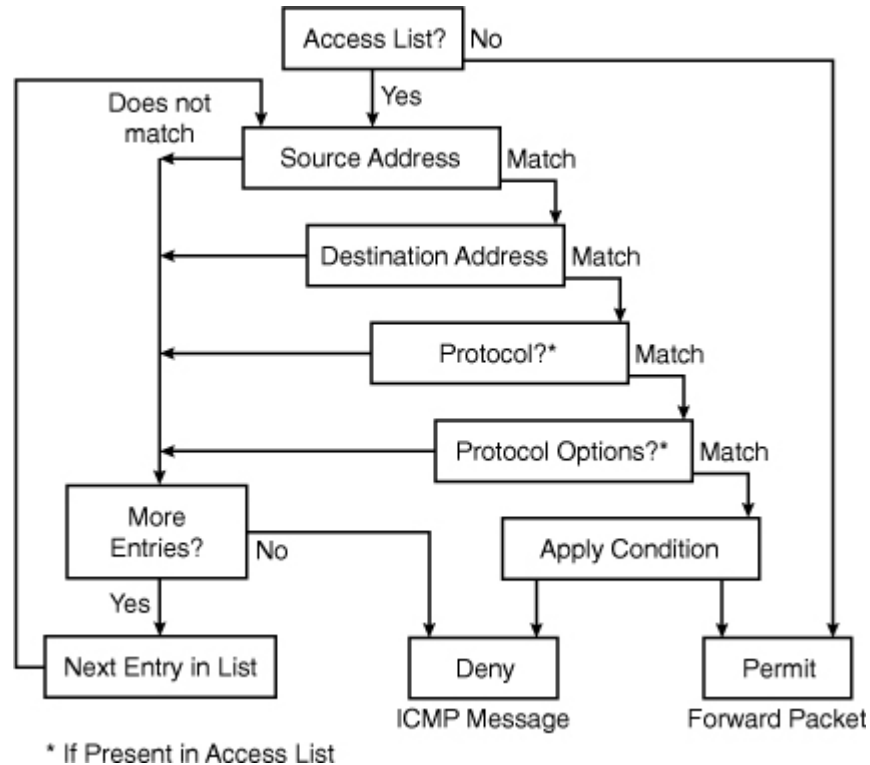
IP Extended Access Lists

Standard access lists offer quick configuration and low overhead in limiting traffic based on source addresses in a network. *Extended access lists* provide a higher degree of control by enabling filtering based on the source and destination addresses, transport layer protocol, and application port number. These features make it possible to limit traffic based on the uses of the network.

Extended Access List Processing

As shown in Figure B-14, every condition tested in a line of an extended access list must match for the line of the access list to match and for the permit or deny condition to be applied. As soon as one parameter or condition fails, the next line in the access list is compared.

Figure B-14. Extended IP Access List Processing Flow.



The extended access list checks source address, destination address, and protocol. Depending on the configured protocol, more protocol-dependent options might be tested. For example, a TCP port might be checked, which allows routers to filter at the application layer.

Extended IP Access List Configuration

Use the **access-list** *access-list-number* {**permit** | **deny**} *protocol* {*source source-wildcard* | **any**} {*destination destination-wildcard* | **any**} [*protocol-specific-options*] [**log**] global configuration command to create an entry in an extended-traffic filter list. Table B-9 describes this command.

Table B-9. ip access-group Command Description

Parameter	Description
<i>access-list-number</i>	Identifies the list to which the entry belongs (a number from 100 to 199 or from 2000 to 2699).
permit deny	Indicates whether this entry allows or blocks traffic.
<i>protocol</i>	ip , tcp , udp , icmp , igmp , gre , eigrp , ospf , nos , ipinip , pim , or a number from 0 to 255. To match any Internet protocol, use the keyword ip . As shown later in this section, some protocols allow more options that are supported by an alternative syntax for this command.
<i>source</i> and <i>destination</i>	Identifies the source and destination IP addresses.
<i>source-wildcard</i> and <i>destination-wildcard</i>	Identifies which bits in the address field must match. A 1 in any bit position indicates don't care bits, and a 0 in any bit position indicates that the bit must strictly match.
any	Use this keyword as an abbreviation for a source and source wildcard or destination and destination wildcard of 0.0.0.0

Parameter	Description
	255.255.255.255.
log	(Optional) Causes informational logging messages about a packet that matches the entry to be sent to the console. Exercise caution when using this keyword, because it consumes CPU cycles. The message is generated for the first packet that matches, and then at 5-minute intervals, including the number of packets permitted or denied in the prior 5-minute interval.

The wildcard masks in an extended access list operate the same way as they do in standard access lists, but note that they are not optional in extended access lists.. The keyword **any** in either the source or the destination position matches any address and is equivalent to configuring an address of 0.0.0.0 with a wildcard mask of 255.255.255.255. [Example B-6](#) shows an example of an extended access list.

Example B-6. Use of the Keyword *any*

Code View: Scroll / [Show All](#)

```
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
! (alternative configuration)
access-list 101 permit ip any any
```

The keyword **host** can be used in either the source or the destination position. It causes the address that immediately follows it to be treated as if it were specified with a mask of 0.0.0.0. [Example B-7](#) shows an example.

Example B-7. Use of the Keyword *host*

```
access-list 101 permit ip 0.0.0.0 255.255.255.255 172.16.5.17 0.0.0.0
! (alternative configuration)
access-list 101 permit ip any host 172.16.5.17
```

Use the **access-list** *access-list-number* {**permit** | **deny**} **icmp** {*source source-wildcard* | **any**} {*destination destination-wildcard* | **any**} [*icmp-type* [*icmp-code*] | *icmp-message*] global configuration command to filter ICMP traffic. The protocol keyword **icmp** indicates that an alternative syntax is being used for this command and that protocol-specific options are available, as described in [Table B-10](#).

Table B-10. Extended IP *access-list icmp* Command Description

Parameter	Description
<i>access-list-number</i>	Identifies the list to which the entry belongs (a number from 100 to 199 or from 2000 to 2699).
permit deny	Indicates whether this entry allows or blocks traffic.
<i>source</i> and <i>destination</i>	Identifies the source and destination IP addresses.
<i>source-wildcard</i> and <i>destination-wildcard</i>	Identifies which bits in the address field must match. A 1 in any bit position indicates don't care bits, and a 0 in any bit position indicates that the bit must strictly match.

Parameter	Description
any	Use this keyword as an abbreviation for a source and source wildcard or destination and destination wildcard of 0.0.0.0 255.255.255.255.
<i>icmp-type</i>	(Optional) Packets can be filtered by ICMP message type. The type is a number from 0 to 255.
<i>icmp-code</i>	(Optional) Packets that have been filtered by ICMP message type can also be filtered by ICMP message code. The code is a number from 0 to 255.
<i>icmp-message</i>	(Optional) Packets can be filtered by a symbolic name representing an ICMP message type or a combination of ICMP message type and ICMP message code. These names are listed in Table B-11 .

Cisco IOS Release 10.3 and later versions provide symbolic names that make configuring and reading complex access lists easier. With symbolic names, it is no longer critical to understand the meaning of the ICMP message type and code (for example, message 8 and message 0 can be used to filter the **ping** command). Instead, the configuration can use symbolic names, as shown in [Table B-11](#). For example, the **echo** and **echo-reply** symbolic names can be used to filter the **ping** command. (You can use the Cisco IOS context-sensitive help feature by entering **?** when entering the **access-list** command to verify the available names and proper command syntax.)

Table B-11. ICMP Message and Type Names

Administratively-prohibited	Dod-host-prohibited
Alternate-address	Dod-net-prohibited
Conversion-error	Echo
Host-precedence-unreachable	Net-tos-redirect
Host-redirect	Net-tos-unreachable
Host-tos-redirect	Net-unreachable
Host-tos-unreachable	Network-unknown
Host-unknown	No-room-for-option
Host-unreachable	Option-missing
Information-reply	Packet-too-big
Information-request	Parameter-problem
Mask-reply	Port-unreachable
Mask-request	Precedence-unreachable
Mobile-redirect	Protocol-unreachable
Net-redirect	Reassembly-timeout

Use the **access-list** *access-list-number* **{permit | deny}** **tcp** *{source source-wildcard | any}* [*operator source-port | source-port*]{*destination destination-wildcard | any*}[*operator destination-port | destination-port*][**established**] global configuration command to filter TCP traffic. The protocol keyword **tcp** indicates that an alternative syntax is being used for this command and that protocol-specific options are available, as described in [Table B-12](#).

Table B-12. Extended IP *access-list tcp* Command Description

Parameter	Description
<i>access-list-number</i>	Identifies the list to which the entry belongs (a number from 100 to 199 or from 2000 to 2699).
permit deny	Indicates whether this entry allows or blocks traffic.
<i>source</i> and <i>destination</i>	Identifies the source and destination IP addresses.
<i>source-wildcard</i> and <i>destination-wildcard</i>	Identifies which bits in the address field must match. A 1 in any bit position indicates don't care bits, and a 0 in any bit position indicates that the bit must strictly match.
any	Use this keyword as an abbreviation for a source and source wildcard or destination and destination wildcard of 0.0.0.0 255.255.255.255.
<i>operator</i>	(Optional) A qualifying condition. Can be lt , gt , eq , or neq .
<i>source-port</i> and <i>destination-port</i>	(Optional) A decimal number from 0 to 65535 or a name that represents a TCP port number.
established	(Optional) A match occurs if the TCP segment has the ACK or RST bits set. Use this if you want a Telnet or other activity to be established in one direction only.

established Keyword in Extended Access Lists

When a TCP session is started between two devices, the first segment that is sent has the synchronize (SYN) code bit set but does not have the acknowledge (ACK) code bit set in the segment header, because it is not acknowledging any other segments. All subsequent segments sent do have the ACK code bit set, because they are acknowledging previous segments sent by the other device. This is how a router can distinguish between a segment from a device that is attempting to *start* a TCP session and a segment of an ongoing *established* session. The RST code bit is set when an established session is being terminated.

When you configure the **established** keyword in a TCP extended access list, it indicates that that access list statement should match only TCP segments in which the ACK or RST code bit is set. In other words, only segments that are part of an established session are matched. Segments that are attempting to start a session do not match the access list statement.

Table B-13 lists TCP port names that can be used instead of port numbers. You can find the port numbers corresponding to these protocols by entering a ? in place of a port number or by looking at the port numbers on <http://www.iana.org/assignments/port-numbers>.

Table B-13. TCP Port Names

Bgp	Hostname
Chargen	Irc
Daytime	Klogin
Discard	Kshell
Domain	Lpd
Echo	Nntp
Finger	Pop2
Ftp control	Pop3

Bgp	Hostname
Ftp-data	Smtip
Gopher	Sunrpc

Other port numbers can be found at <http://www.iana.org/assignments/port-numbers>. A partial list of the assigned TCP port numbers is shown in Table B-14.

Table B-14. Some Reserved TCP Port Numbers

Port Number (Decimal)	Keyword	Description
7	ECHO	Echo
9	DISCARD	Discard
13	DAYTIME	Daytime
19	CHARGEN	Character generator
20	FTP-DATA	File Transfer Protocol (data)
21	FTP-CONTROL	File Transfer Protocol
23	TELNET	Terminal connection
25	SMTP	Simple Mail Transfer Protocol
37	TIME	Time of day
43	WHOIS	Who is
53	DOMAIN	Domain name server
79	FINGER	Finger
80	WWW	World Wide Web HTTP
101	HOSTNAME	NIC hostname server

Use the **access-list** *access-list-number* {**permit** | **deny**} **udp** {*source source-wildcard* | **any**} [*operator source-port* | *source-port*] {*destination destination-wildcard* | **any**} [*operator destination-port* | *destination-port*] global configuration command to filter User Datagram Protocol (UDP) traffic. The protocol keyword **udp** indicates that an alternative syntax is being used for this command and that protocol-specific options are available, as described in Table B-15.

Table B-15. Extended IP *access-list udp* Command Description

Parameter	Description
<i>access-list-number</i>	Identifies the list to which the entry belongs (a number from 100 to 199 or from 2000 to 2699).
permit deny	Indicates whether this entry allows or blocks traffic.
<i>source</i> and <i>destination</i>	Identifies the source and destination IP addresses.
<i>source-wildcard</i> and <i>destination-wildcard</i>	Identifies which bits in the address field must match. A 1 in any bit position indicates don't care bits, and a 0 in any bit position indicates that the bit must strictly match.
any	Use this keyword as an abbreviation for a source and source wildcard or destination and destination wildcard of 0.0.0.0 255.255.255.255.

Parameter	Description
<i>operator</i>	(Optional) A qualifying condition. Can be lt, gt, eq, or neq.
<i>source-port and destination-port</i>	(Optional) A decimal number from 0 to 65535 or a name that represents a UDP port number.

Table B-16 lists UDP port names that can be used instead of port numbers. You can find port numbers corresponding to these protocols by entering a ? in place of a port number or by looking at <http://www.iana.org/assignments/port-numbers>.

Table B-16. UDP Port Names

Biff	Nameserver	Syslog
Bootpc	NetBios-dgm	Tacacs-ds
Bootps	NetBios-ns	Talk
Discard	Ntp	Tftp
Dns	Rip	Time
Dnsix	Snmp	Whois
Echo	Snmptrap	Xdmcp
Mobile-ip	Sunrpc	

Other port numbers can be found at <http://www.iana.org/assignments/port-numbers>. A partial list of the assigned UDP port numbers is shown in Table B-17.

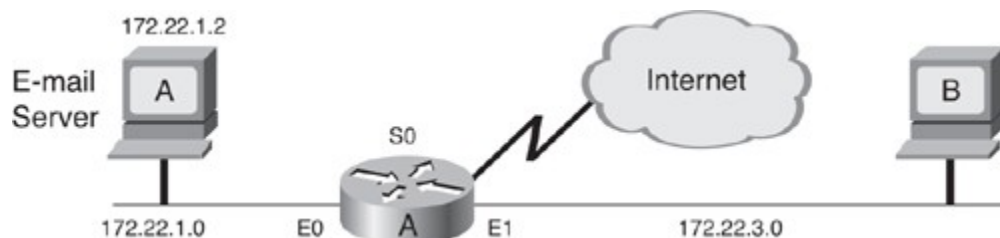
Table B-17. Some Reserved UDP Port Numbers

Port Number (Decimal)	Keyword	Description
7	ECHO	Echo
9	DISCARD	Discard
37	TIME	Time of day
42	NAMESERVER	Host name server
43	WHOIS	Who is
53	DNS	Domain name server
67	BOOTPS	Bootstrap protocol server
68	BOOTPC	Bootstrap protocol client
69	TFTP	Trivial File Transfer Protocol
123	NTP	Network Time Protocol
137	NetBios-ns	NetBIOS name service
138	NetBios-dgm	NetBIOS datagram service
161	SNMP	SNMP
162	SNMPTrap	SNMP traps
520	RIP	RIP

Extended Access List Examples

In Figure B-15, Router A's interface Ethernet 1 is part of a Class B subnet with the address 172.22.3.0, Router A's interface Serial 0 is connected to the Internet, and the e-mail server's address is 172.22.1.2. The access list configuration applied to Router A is shown in Example B-8.

Figure B-15. Network Used for the Extended IP Access List Example.



Example B-8. Configuration on Router A in Figure B-15

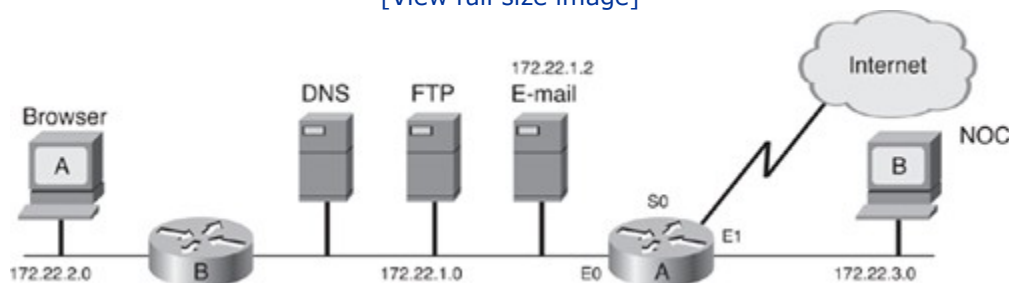
```
access-list 104 permit tcp any 172.22.0.0 0.0.255.255
established
access-list 104 permit tcp any host 172.22.1.2 eq smtp
access-list 104 permit udp any any eq dns
access-list 104 permit icmp any any echo
access-list 104 permit icmp any any echo-reply
!
interface serial 0
ip access-group 104 in
```

In Example B-8, access list 104 is applied inbound on Router A's Serial 0 interface. The keyword **established** is used only for the TCP protocol to indicate an established connection. A match occurs if the TCP segment has the ACK or RST bits set, which indicate that the packet belongs to an existing connection. If the session is not already established (the ACK bit is not set and the SYN bit is set), this means that someone on the Internet is attempting to initialize a session, in which case the packet is denied. This configuration also permits Simple Mail Transfer Protocol (SMTP) traffic from any address to the e-mail server. UDP domain name server packets and ICMP echo and echo-reply packets are also permitted from any address to any other address.

Another example is shown in Figure B-16. Example B-9 shows the access list configuration applied to Router A.

Figure B-16. Extended IP Access List Example with Many Servers.

[View full size image]



Example B-9. Configuration on Router A in Figure B-16

Code View: Scroll / [Show All](#)

```
access-list 118 permit tcp any 172.22.0.0 0.0.255.255 eq www established
access-list 118 permit tcp any host 172.22.1.2 eq smtp
access-list 118 permit udp any any eq dns
access-list 118 permit udp 172.22.3.0 0.0.0.255 172.22.1.0 0.0.0.255 eq
snmp

access-list 118 deny icmp any 172.22.0.0 0.0.255.255 echo
access-list 118 permit icmp any any echo-reply
!
interface ethernet 0
ip access-group 118 out
```

In [Example B-9](#), access list 118 is applied outbound on Router A's Ethernet 0 interface. With the configuration shown in [Example B-9](#), *replies* to queries from the Client A browser (or any other host on the corporate network) to the Internet are allowed back into the corporate network (because they are established sessions). Browser queries *from* external sources are not explicitly allowed and are discarded by the implicit **deny any** at the end of the access list.

The access list in [Example B-9](#) also allows e-mail (SMTP) to be delivered exclusively to the mail server. The name server is permitted to resolve DNS requests. The 172.22.1.0 subnet is controlled by the network management group located at the NOC server (Client B), so network-management queries (Simple Network Management Protocol [SNMP]) will be allowed to reach these devices in the server farm. Attempts to ping the corporate network from the outside or from subnet 172.22.3.0 will fail because the access list blocks the echo requests. However, replies to echo requests generated from within the corporate network are allowed to reenter the network.

Location of Extended Access Lists

Because extended access lists can filter on more than a source address, location is no longer the constraint it was when considering the location of a standard access list. Policy decisions and goals are frequently the driving forces behind extended access list placement.

If your goal is to minimize traffic congestion and maximize performance, you might want to push the access lists close to the source to minimize cross-network traffic and administratively prohibited ICMP messages. If your goal is to maintain tight control over access lists as part of your network security strategy, you might want them to be more centrally located. Notice how changing network goals affects access list configuration.

Here are some things to consider when placing extended access lists:

- Minimize distance traveled by traffic that will be denied (and ICMP unreachable messages).
- Keep denied traffic off the backbone.
- Select the router that will have the CPU overhead from processing the access lists.
- Consider the number of interfaces affected.
- Consider access list management and security.
- Consider network growth impacts on access list maintenance.

Restricting Virtual Terminal Access

This section discusses how you can use standard access lists to limit virtual terminal access. Standard and extended access lists block packets from going *through* the router. They are not designed to block packets

that originate within the router. For example, an outbound Telnet extended access list does not prevent router-initiated Telnet sessions by default.

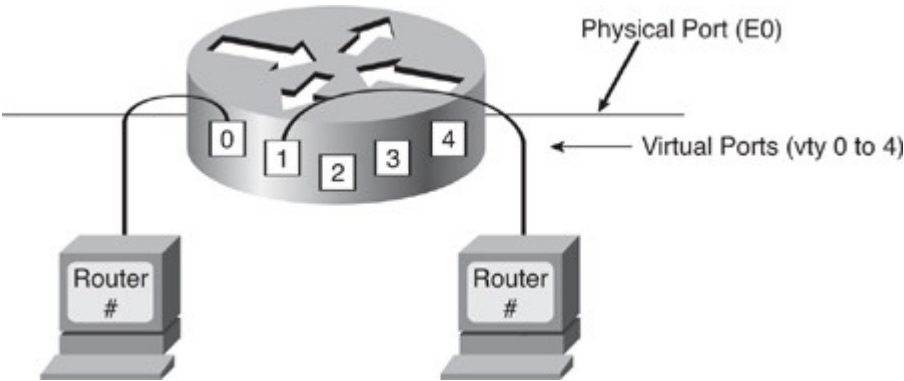
For security purposes, users can be denied virtual terminal (vty) access to the router, or they can be permitted vty access to the router but denied access to destinations from that router. Restricting vty access is less of a traffic-control mechanism than one technique for increasing network security.

Vty access is accomplished using the Telnet or Secure Shell (SSH) protocol. There is only one type of vty access list.

How to Control vty Access

Just as a router has physical ports or interfaces such as Ethernet 0 and Ethernet 1, it also has virtual ports. These virtual ports are called virtual terminal lines. By default, there are five such virtual terminal lines on a router, numbered vty 0 to 4, as shown in Figure B-17.

Figure B-17. A Router Has Five Virtual Terminal Lines (Virtual Ports) by Default.



You should set identical restrictions on all virtual terminal lines, because you cannot control on which virtual terminal line a user will connect.

Virtual Terminal Line Access Configuration

Use the **line vty** {*vty-number* | *vty-range*} global configuration command to place the router in line configuration mode, as described in Table B-18.

Table B-18. line vty Command Description

Parameter	Description
<i>vty-number</i>	Indicates the number of the vty line to be configured
<i>vty-range</i>	Indicates the range of vty lines to which the configuration applies

Use the **access-class** *access-list-number* {**in** | **out**} line configuration command to link an existing access list to a terminal line or range of lines, as described in Table B-19.

Table B-19. access-class Command Description

Parameter	Description
<i>access-list-number</i>	Indicates the number of the standard access list to be linked to a terminal line. This is a decimal number from 1 to 99 or from 1300 to 1999.

Parameter	Description
in	Prevents the router from receiving incoming connections from the addresses defined in the access list.
out	Prevents someone from initiating a Telnet to the addresses defined in the access list.

Note

When you use the **out** keyword in the **access-class** command, the addresses in the specified standard access list are actually treated as *destination* addresses, rather than as source addresses.

In [Example B-10](#), any device on network 192.168.55.0 is permitted to establish a virtual terminal session (for example, a Telnet session) with the router. Of course, the user must know the appropriate passwords for entering user mode and privileged mode.

Example B-10. Configuration to Restrict Telnet Access to a Router

```
access-list 12 permit 192.168.55.0
0.0.0.255
!
line vty 0 4
access-class 12 in
```

Notice that in this example, identical restrictions have been set on all virtual terminal lines (0 to 4), because you cannot control on which virtual terminal line a user will connect. Note that the implicit **deny any** still applies to this alternative application of access lists.

Verifying Access List Configuration

Use the **show access-lists** [*access-list-number* | *name*] privileged EXEC command to display access lists from all protocols, as described in [Table B-20](#). If no parameters are specified, all access lists are displayed.

Table B-20. *show access-lists* Command Description

Parameter	Description
<i>access-list-number</i>	(Optional) Number of the access list to display
<i>name</i>	(Optional) Name of the access list to display

The system counts how many packets match each line of an access list. The counters are displayed by the **show access-lists** command.

[Example B-11](#) illustrates sample output from the **show access-lists** command. In this example, the first line of the access list has been matched three times, and the last line has been matched 629 times. The second line has not been matched.

Example B-11. Output of the *show access-lists* Command

```
p1r1#show access-lists
Extended IP access list 100
  deny tcp host 10.1.1.2 host 10.1.1.1 eq telnet (3 matches)
  deny tcp host 10.1.2.2 host 10.1.2.1 eq telnet
```



```
permit ip any any (629 matches)
```

Use the **show ip access-list** [*access-list-number* | *name*] EXEC command to display IP access lists, as described in [Table B-21](#). If no parameters are specified, all IP access lists are displayed.

Table B-21. *show ip access-list* Command Description

Parameter	Description
<i>access-list-number</i>	(Optional) Number of the IP access list to display
<i>name</i>	(Optional) Name of the IP access list to display

Use the **clear access-list counters** [*access-list-number* | *name*] EXEC command to clear the counters for the number of matches in an extended access list, as described in [Table B-22](#). If no parameters are specified, the counters are cleared for all access lists.

Table B-22. *clear access-list counters* Command Description

Parameter	Description
<i>access-list-number</i>	(Optional) Number of the access list for which to clear the counters
<i>name</i>	(Optional) Name of the access list for which to clear the counters

Use the **show line** [*line-number*] EXEC command to display information about terminal lines. The *line-number* is optional and indicates the absolute line number of the line for which you want to list parameters. If a line number is not specified, all lines are displayed.

IPv4 Address Planning

A well-designed large-scale internetwork with an effective IP addressing plan has many benefits, as described in this section.

Benefits of an Optimized IP Addressing Plan

An optimized IP addressing plan uses hierarchical addressing.

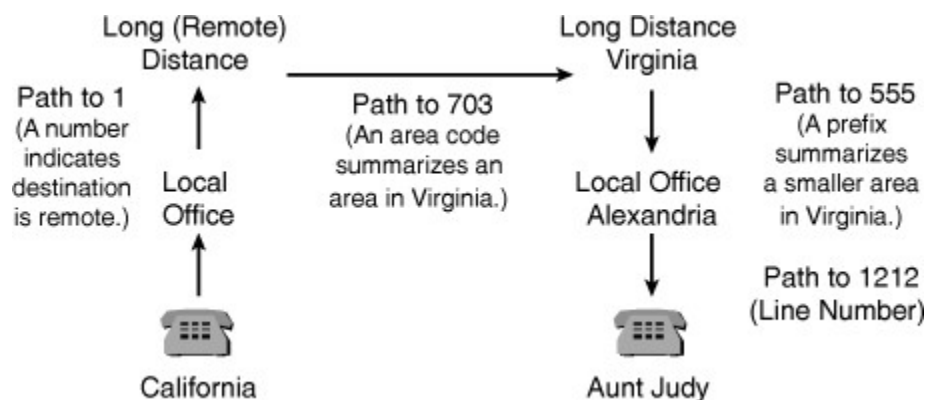
Perhaps the best-known addressing hierarchy is the telephone network. The telephone network uses a hierarchical numbering scheme that includes country codes, area codes, and local exchange numbers. For example, if you are in San Jose, California, and you call someone else in San Jose, you dial the San Jose prefix, 528, and the person's four-digit line number. Upon seeing the number 528, the central office recognizes that the destination telephone is within its area, so it looks up the four-digit number and transfers the call.

Note

In many places in North America now, the area code must also be dialed for local calls. This is because of changes in the use of specific digits for area codes and local exchange numbers. The telephone network is suffering from *address exhaustion*, just like the IP network. Changes in how telephone numbers are used is one solution being implemented to solve this problem.

In another example (see [Figure B-18](#)), to call Aunt Judy in Alexandria, Virginia, from San Jose, you dial 1, and then the area code 703, and then the Alexandria prefix 555, and then Aunt Judy's local line number, 1212. The central office first sees the number 1, indicating a remote call, and then looks up the number 703. The central office immediately routes the call to a central office in Alexandria. The San Jose central office does not know exactly where 555-1212 is in Alexandria, nor does it have to. It needs to know only the area codes, which summarize the local telephone numbers within an area.

Figure B-18. The Telephone Network Uses an Addressing Hierarchy.



Note

As you might have noticed, the telephone number used in this example is the number for international directory assistance. It is used for illustration purposes to ensure that Aunt Judy's personal number is not published.

If there were no hierarchical structure, every central office would need to have every telephone number worldwide in its locator table. Instead, the central offices have summary numbers, such as area codes and country codes. A summary number (address) represents a group of numbers. For example, an area code such as 408 is a summary number for the San Jose area. In other words, if you dial 1-408 from anywhere in the United States or Canada, followed by a seven-digit telephone number, the central office routes the call to

a San Jose central office. Similarly, a routed network can employ a hierarchical addressing scheme to take advantage of those same benefits.

One of the benefits of hierarchical addressing is a reduced number of routing table entries. Whether it is with your Internet routers or your internal routers, you should try to keep your routing tables as small as possible by using route summarization.

Summarization (also called *aggregation*, *supernetting*, or *information hiding*) is not a new concept. When a router announces a route to a given network, the route is a summarization of all the host and device individual addresses that reside on that network. Route summarization is a way of having a single IP address represent a collection of IP addresses. This is most easily accomplished when you employ a hierarchical addressing plan. By summarizing routes, you can keep your routing table entries (on the routers that receive the summarized routes) manageable, which offers the following benefits:

- More efficient routing.
- A reduced number of CPU cycles when recalculating a routing table or sorting through the routing table entries to find a match.
- Reduced router memory requirements.
- Reduced bandwidth required to send the fewer, smaller routing updates.
- Faster convergence after a change in the network.
- Easier troubleshooting.
- Increased network stability. Because summarization limits the propagation of detailed routes, it also reduces the impact to the network when these detailed routes fail.

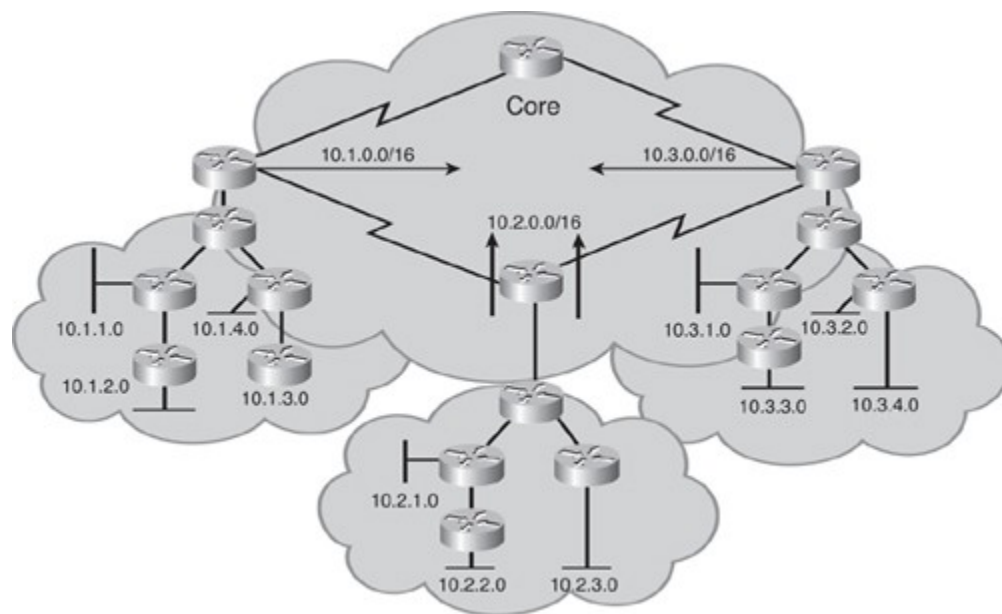
Another benefit of hierarchical addressing is the efficient allocation of addresses. Hierarchical addressing lets you take advantage of all possible addresses because you group them contiguously. With random address assignment, you might end up wasting groups of addresses because of addressing conflicts. For example, classful routing protocols (discussed in the later section "[Implementing VLSM in a Scalable Network](#)") automatically create summary routes at a network boundary. Therefore, these protocols do not support discontinuous addressing (as you can see in [Chapter 1](#), "Routing Services"), so some addresses would be unusable if not assigned contiguously.

Scalable Network Addressing Example

The network illustrated in [Figure B-19](#) shows an example of scalable addressing. In this example, a U.S. national drugstore chain plans to have a retail outlet in every city in the country with a population greater than 10,000. Each of the 50 states has up to 100 stores, with 2 Ethernet LANs in each store, as follows:

Figure B-19. Scalable Addressing Allows Summarization.

[\[View full size image\]](#)



- One LAN is used to track customer prescriptions, pharmacy inventory, and reorder stock.
- The second LAN is used to stock the rest of the store and connect the cash registers to a corporatewide, instantaneous point-of-sale evaluation tool.

The total number of Ethernet LAN networks is 50 states * 100 stores per state * 2 LANs per store = 10,000. (An equal number of serial links interconnects these stores.)

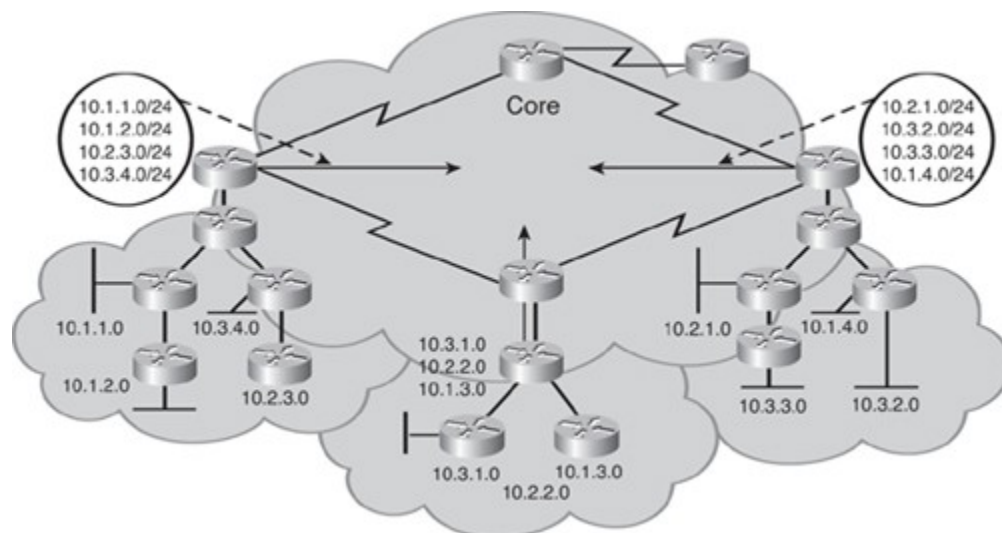
Using a scalable design and creating 51 divisions (one for each state and one for the backbone interconnecting the divisions), the corporation can assign each division a block of IP addresses 10.x.0.0 /16. Each LAN is assigned a /24 subnet of network 10.0.0.0, and each division has 200 such subnets (two for each of the 100 stores). The network will have 10,000 subnets; without summarization, each of the 5000 routers will have all these networks in their routing tables. If each division router summarizes its block of networks 10.x.0.0 /16 at the entry point to the core network, any router in a division has only the 200 /24 subnets within that division, plus the 49 10.x.0.0 /16 summarizations that represent the other divisions, in its routing table. This results in a total of 249 networks in each IP routing table.

Nonscalable Network Addressing

In contrast to the previous example, if a hierarchical addressing plan is not used, summarization is not possible, as is the case in [Figure B-20](#). Problems can occur in this network related to the frequency and size of routing table updates and how topology changes are processed in summarized and unsummarized networks. These problems are described next.

Figure B-20. Nonscalable Addressing Results in Large Routing Tables.

[\[View full size image\]](#)



Update Size

Routing protocols such as the Routing Information Protocol (RIP), which sends a periodic update every 30 seconds, use valuable bandwidth to maintain a table without summarization. A single RIP update packet is limited to carrying 25 routes. Therefore, 10,000 routes means that RIP on every router must create and send 400 packets every 30 seconds. With summarized routes, the 249 routes means that only 10 packets need to be sent every 30 seconds.

Unsummarized Internetwork Topology Changes

A routing table with 10,000 entries constantly changes. To illustrate this constant change, consider the sample network with a router at each of 5000 different sites. A power outage occurs at site A, a backhoe digs a trench at site B, a newly hired system administrator begins work at site C, a Cisco IOS software upgrade is in progress at site D, and a newly added router is being installed at site E.

Every time a route changes, all the routing tables must be updated. For example, when using a routing protocol such as Open Shortest Path First (OSPF), an upgrade or topology change on the internetwork causes a shortest path first (SPF) calculation. The SPF calculations are large because each router needs to calculate all known pathways to each of the 10,000 networks. Each change a router receives requires time and CPU resources to process.

Summarized Network Topology Changes

In contrast to an unsummarized network, a summarized network responds efficiently to network changes. For example, in the sample drugstore network with 200 routes for each division, the routers within the division see all the subnets for that division. When a change occurs on one of the 200 routes in the division, all other routers in the division recalculate to reflect the topology change of those affected networks. However, the core router of that division passes a summarized /16 route and suppresses the /24 networks from advertisement to the core routers of other divisions. The summarized route is announced as long as any portion of the summarized block can be reached from that core router. The more specific routes are suppressed so that changes from this division are not propagated to other divisions.

In this scenario, each router has only 200 /24 networks, compared to the 10,000 /24 networks in an unsummarized environment. Obviously, the amount of CPU resources, memory, and bandwidth required for the 200 networks is less than the 10,000 networks. With summarization, each division hides more-specific information from the other divisions and passes only the summarized route that represents that overall division.

Hierarchical Addressing Using Variable-Length Subnet Masks

VLSM is a crucial component of an effective IP addressing plan for a scalable network. This section introduces VLSM, provides examples, and discusses methods of determining the best subnet mask for a given address requirement.

Network Mask

This section discusses the purpose of the network mask and its use within a network.

Use of the Network Mask

If a PC has an IP address of 192.168.1.67 with a mask of 255.255.255.240 (or a prefix length of /28), it uses this mask to determine the valid host addresses for devices on its local connection. These devices have the first 28 bits in their IP address in common (the range of these local devices is 192.168.1.65 through 192.168.1.78). If communication with any of these devices is necessary, the PC uses Address Resolution Protocol (ARP) to find the device's corresponding Media Access Control (MAC) address (assuming that it does not already have a destination MAC address for the IP address in its ARP table). If a PC needs to send information to an IP device that is not in the local range, the PC instead forwards the information to its default gateway. (The PC also uses ARP to discover the MAC address of the default gateway.)

A router behaves in a similar manner when it makes a routing decision. A packet arrives on the router and is passed to the routing table. The router compares the packet's destination IP address to the entries in the routing table. These entries have a prefix length associated with them.

The router uses the prefix length as the minimum number of destination address bits that must match to use the corresponding outbound interface that is associated with a network entry in the routing table.

Network Mask Example

Consider a scenario in which an IP packet with a destination address of 192.168.1.67 is sent to a router. [Example B-12](#) shows the router's IP routing table.

Example B-12. IP Routing Table for Network Mask Example

```
192.168.1.0 is subnetted, 4 subnets
O 192.168.1.16/28 [110/1800] via 172.16.1.1, 00:05:17, Serial 0
C 192.168.1.32/28 is directly connected, Ethernet 0
O 192.168.1.64/28 [110/10] via 192.168.1.33, 00:05:17, Ethernet 0
O 192.168.1.80/28 [110/1800] via 172.16.2.1, 00:05:17, Serial 1
```

In this scenario, the router determines where to send a packet that is destined for 192.168.1.67 by looking at the routing table. The routing table has four entries for network 192.168.1.0. The router compares the destination address to each of the four entries for this network.

The destination address of 192.168.1.67 has the first three octets in common with all four entries in the routing table, but it is not clear by looking at the decimal representation which of those entries is the best match to route this packet. A router handles all packets in binary, not dotted-decimal, notation.

Following is the binary representation of the last octet for destination address 192.168.1.67 and the binary representation of the last octet for the four entries in the IP routing table. Because the prefix length is 28 and all four entries match at least the first 24 bits of 192.168.1, the router must find the routing table entry that matches the first 4 bits (bits 25 to 28) of the number 67. It is not important if the last 4 bits match (because they are host bits), so the target is 0100xxxx. The routing entry 64, which has a value of 0100 in the first 4 bits, is the only one that matches the requirement:

- 67: 01000011
- 16: 00010000
- 32: 00100000

- 64: 01000000
- 80: 01010000

The router therefore uses the 192.168.1.64 entry in the routing table and forwards this packet out of its Ethernet 0 interface to the next router (192.168.1.33).

Implementing VLSM in a Scalable Network

A major network (also known as a classful network) is a Class A, B, or C network.

With classful routing, routing updates do not carry the subnet mask. Therefore, only one subnet mask can be used within a major network. This is known as fixed-length subnet masking (FLSM). An example of a classful routing protocol is RIP Version 1 (RIPv1).

With classless routing, routing updates do carry the subnet mask. Therefore, different masks may be used for different subnets within a major network. This is known as VLSM. Examples of classless routing protocols are RIP Version 2 (RIPv2), OSPF, Intermediate System-to-Intermediate System (IS-IS), and Enhanced Interior Gateway Routing Protocol (EIGRP).

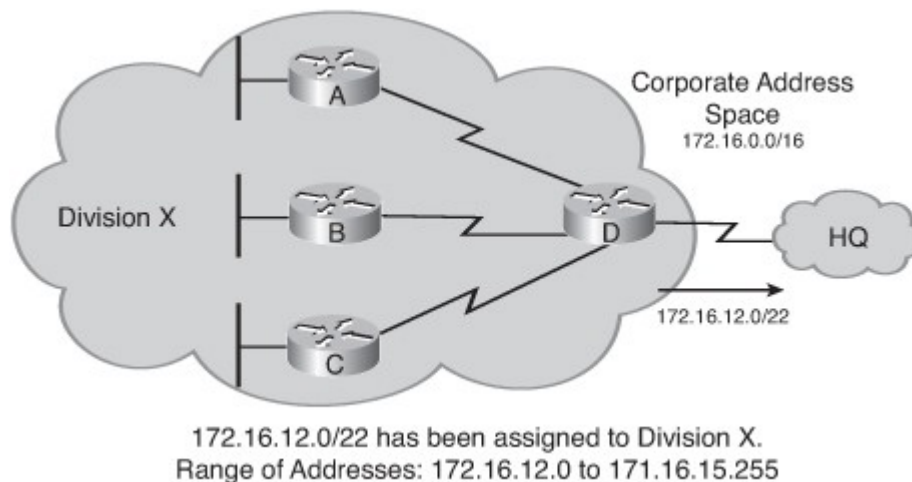
Note

Classful and classless routing protocols are discussed further in [Chapter 1](#).

VLSM allows more than one subnet mask within a major network and enables the subnetting of a previously subnetted network address.

The network shown in [Figure B-21](#) is used to illustrate how VLSM works.

Figure B-21. Network for the VLSM Example.



The following are some characteristics that permit VLSMs to conserve IP addresses:

- **Efficient use of IP addresses**—Without the use of VLSMs, companies are locked into implementing a single subnet mask within an entire Class A, B, or C network number.
 - For example, suppose a network architect decides to use the 172.16.0.0/16 address space to design a corporate network. The architect determines that 64 blocks of addresses with up to 1022 hosts in each are required. Therefore, 10 host bits ($2^{10} - 2 = 1022$) and 6 subnet bits ($2^6 = 64$) are required for each block. The mask is therefore 255.255.252.0. The prefix is / 22.

- The network architect assigns address block 172.16.12.0/22 to Division X, as shown in [Figure B-21](#). The prefix mask of /22 indicates that all addresses within that range have the first 22 bits in common (when reading from left to right). The prefix mask provides Division X with a range of addresses from 172.16.12.0 through 172.16.15.255. The details of the range of addresses available to Division X are shown in the center block of [Figure B-22](#). Within Division X, the networks are assigned addresses in this range, with varying subnet masks. Details of these address assignments are provided in the next section.

Figure B-22. Center Block Is Range of Addresses for VLSM for Division X in Figure B-21.

Dotted Decimal Notation	Binary Notation
172.16.11.0	10101100.00010000.00001011.00000000
(Text Omitted for Continuation of Bit/Number Pattern)	
172.16.12.0	10101100.00010000.00001100.00000000
172.16.12.1	10101100.00010000.00001100.00000001
172.16.12.255	10101100.00010000.00001100.11111111
172.16.13.0	10101100.00010000.00001101.00000000
172.16.13.1	10101100.00010000.00001101.00000001
172.16.13.255	10101100.00010000.00001101.11111111
172.16.14.0	10101100.00010000.00001110.00000000
172.16.14.1	10101100.00010000.00001110.00000001
172.16.14.255	10101100.00010000.00001110.11111111
172.16.15.0	10101100.00010000.00001111.00000000
172.16.15.1	10101100.00010000.00001111.00000001
172.16.15.255	10101100.00010000.00001111.11111111
(Text Omitted for Continuation of Bit/Number Pattern)	
172.16.16.0	10101100.00010000.00010000.00000000

- **Greater capability to use route summarization**—VLSMs allow for more hierarchical levels within an addressing plan and thus allow better route summarization within routing tables. For example, in [Figure B-21](#), address 172.16.12.0/22 summarizes all the subnets that are further subnets of 172.16.12.0/22.
- **Reduced number of routing table entries**—In a hierarchical addressing plan, route summarization allows a single IP address to represent a collection of IP addresses. When VLSM is used in a hierarchical network, it allows summarized routes, which keeps routing table entries (on the routers that receive the summarized routes) manageable and provides the benefits described earlier in the “[IPv4 Address Planning](#)” section.
- Because of the reduced router requirements, it also might be possible to use some less-powerful (and therefore less-expensive) routers in the network.

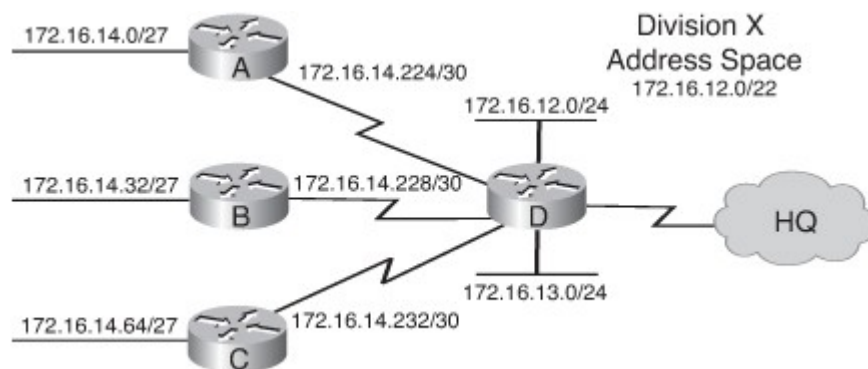
The address 172.16.12.0/22 represents all the addresses that have the same first 22 bits as 172.16.12.0. [Figure B-22](#) displays the binary representation of networks 172.16.11.0 through 172.16.16.0. Notice that 172.16.12.0 through 172.16.15.255 all have the first 22 bits in common, whereas 172.16.11.0 and 172.16.16.0 do not have the same first 22 bits. Therefore, the address 172.16.12.0/22 represents the range of addresses 172.16.12.0 through 172.16.15.255.

VLSM Calculation Example

You can best understand the design and implementation of a scalable IP address plan if you study a detailed example of how a VLSM network is laid out.

[Figure B-23](#) shows a detailed view of the same Division X shown in [Figure B-21](#).

Figure B-23. Detailed IP Addressing of Division X in Figure B-21.



In Division X, the following exist:

- One virtual LAN (VLAN) on each of the 2 Ethernet ports of Router D, each with 200 users.
- Three remote sites, at Routers A, B, and C, each with a 24-port Cisco switch. The number of users at each remote site does not exceed 20.
- Three serial links to the remote sites. The serial links are point-to-point Frame Relay and require an address on each side.

VLSM allows you to further subnet the 172.16.12.0/22 address space, using variable masks, to accommodate the network requirements. For example, because point-to-point serial lines require only two host addresses, you can use a subnetted address that has only two host addresses and therefore does not waste scarce subnet numbers.

To start the VLSM process, determine the number of subnets necessary for the networks to which you need to assign IP addresses, and determine the number of hosts necessary per subnetwork. You can determine the number of hosts by checking corporate policy to see whether a limit is set per segment or VLAN, checking the physical number of ports on a switch, and checking the current size of the network or networks at other sites that fulfill the same role.

Note

The decimal-to-binary conversion chart earlier in this appendix might be helpful when you are calculating VLSMs.

LAN Addresses

Because IP addresses are binary, they are used in blocks of powers of 2. A block of addresses contains 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, and so on addresses. Two addresses are lost each time you create a subnet: one for the network (wire) address and the other for the directed broadcast address.

The lowest address of the range, where the host bits are all 0s, is known as the network number or the wire address. The top of the address range, where the host bits are all 1s, is the directed broadcast address. The number of addresses in a block that can be assigned to devices is $2^h - 2$, where h is the number of host bits. For example, with 3 host bits, $2^3 - 2 = 8 - 2 = 6$ addresses can be assigned.

To determine the size of the block of addresses needed for a subnet, follow these steps:

- Step 1.** Calculate the maximum number of hosts on that subnet.
- Step 2.** Add 2 to that number for the broadcast and subnet numbers.

Step 3. Round up to the next higher power of 2.

In this example, the VLANs each have 200 users. Therefore, the number of addresses required is $200 + 2 = 202$. Rounding up to the next power of 2 gives you 256. Thus, 8 ($2^8 = 256$) host bits are required for the VLANs. Therefore, the prefix is /24 (32 bits – 8 bits for the host = 24 bits). The network administrator subnets the 172.16.12.0/22 into four /24 subnets on Router D.

172.16.12.0/24 is assigned to VLAN 1, and 172.16.13.0/24 is assigned to VLAN 2. This leaves two /24 subnets, 172.16.14.0/24 and 172.16.15.0/24, to use for the switches at the three remote sites and the three serial point-to-point links.

The number of addresses required for the LANs at each remote site is $20 + 2 = 22$. Rounding this up to the next power of 2 gives you 32. Thus, 5 host bits ($2^5 = 32$) are required to address the remote users at each site. Therefore, the prefix to use is /27 (32 bits – 5 bits for the host = 27).

You cannot use the 172.16.12.0/24 or 172.16.13.0/24 networks, because they are assigned to VLANs 1 and 2 on Router D. The process to further subnet 172.16.14.0/24 into /27 subnets is shown in Figure B-24. The first three subnets calculated in Figure B-24 are used on the LANs in Figure B-23.

Figure B-24. Calculating Subnet Addresses for the LANs in Figure B-23.

Subnetted Address: 172.16.14.0/24				
In Binary 10101100.00010000.00001110.00000000				
VLSM Address: 172.16.14.0/27				
In Binary 10101100.00010000.00001110.00000000				
1st Subnet:	10101100	. 00010000	.00001110.	000 00000=172.16.14.0/27
2nd Subnet:	172	. 16	.00001110.	001 00000=172.16.14.32/27
3rd Subnet:	172	. 16	.00001110.	010 00000=172.16.14.64/27
4th Subnet:	172	. 16	.00001110.	011 00000=172.16.14.96/27
5th Subnet:	172	. 16	.00001110.	100 00000=172.16.14.128/27
6th Subnet:	172	. 16	.00001110.	101 00000=172.16.14.160/27
7th Subnet:	172	. 16	.00001110.	110 00000=172.16.14.192/27
8th Subnet:	172	. 16	.00001110.	111 00000=172.16.14.224/27
	Network		Subnet VLSM Subnet	Host

Serial Line Addresses

After you establish the addresses for the LANs at the remote sites, you must address the serial links between the remote sites and Router D. Because the serial links require two addresses, the number of addresses required is $2 + 2 = 4$ (the 2 additional addresses are for the network number and the directed broadcast address).

Note

Because only two devices exist on point-to-point links, a specification has been developed (as documented in RFC 3021, *Using 31-Bit Prefixes on IPv4 Point-to-Point Links*) to allow the use of only 1 host bit on such links, resulting in a /31 mask. The two addresses created—with the host bit equal to 0 and with the host bit equal to 1—are interpreted as the addresses of the interfaces on either end of the link rather than as the subnet address and the directed broadcast address. Support for /31 masks is provided on some Cisco devices running IOS Release 12.2 and later. You can find details about the support for this (and other features) on specific platforms and IOS releases at the Cisco feature navigator site (<http://www.cisco.com/go/fn>). In the example in this section, we do not assume the use of this feature.

In this case, there is no need to round up, because 4 is a power of 2. Therefore, 2 host bits will allow for two hosts per subnet. A network mask of /30 (32 bits – 2 host bits = 30 bits) is used. This prefix allows for only two hosts—just enough hosts for a point-to-point connection between a pair of routers.

To calculate the subnet addresses for the WAN links, further subnet one of the unused /27 subnets. In this example, 172.16.14.224/27 is further subnetted with a prefix of /30. The three additional subnet bits result in $2^3 = 8$ subnets for the WAN links.

It is important to remember that only *unused* subnets should be further subnetted. In other words, if you use any addresses from a subnet, that subnet should not be further subnetted. In [Figure B-23](#), three subnet numbers are used on the LANs. Another, as-yet-unused subnet, 172.16.14.224/27, is further subnetted for use on the WANs.

The WAN addresses derived from 172.16.14.224/27 are as follows. The shaded bits are the 3 additional subnet bits:

- 172.16.14.11100000 = 172.16.14.224/30
- 172.16.14.11100100 = 172.16.14.228/30
- 172.16.14.11101000 = 172.16.14.232/30
- 172.16.14.11101100 = 172.16.14.236/30
- 172.16.14.11110000 = 172.16.14.240/30
- 172.16.14.11110100 = 172.16.14.244/30
- 172.16.14.11111000 = 172.16.14.248/30
- 172.16.14.11111100 = 172.16.14.252/30

The first three of these subnets are used on the WANs shown in [Figure B-23](#). The address information for the Router A to Router D link is as follows:

- **Network number**—172.16.14.224
- **Router A serial interface**—172.16.14.225
- **Router D serial interface**—172.16.14.226
- **Broadcast address**—172.16.14.227

The address information for the Router B to Router D link is as follows:

- **Network number**—172.16.14.228
- **Router B serial interface**—172.16.14.229
- **Router D serial interface**—172.16.14.230
- **Broadcast address**—172.16.14.231

The address information for the Router C to Router D link is as follows:

- **Network number**—172.16.14.232
- **Router C serial interface**—172.16.14.233
- **Router D serial interface**—172.16.14.234
- **Broadcast address**—172.16.14.235

Note that to provide the most flexibility for future growth, the 172.16.14.224/27 subnet was selected for the WANs instead of using the next available subnet, 172.16.14.96/27. For example, if the company purchases

more switches, the next IP segment could be assigned the 172.16.14.96/27 subnet, and the new remote site would be connected to Router D with the 172.16.14.236/30 serial subnet.

The 172.16.15.0/24 block could have been used for these /30 subnets, but only three subnets are currently needed, so a lot of the address space would be unused. The 172.16.15.0/24 block is now available to use on another LAN in the future.

Summary of Addresses Used in the VLSM Example

Figure B-25 summarizes the addresses, in binary, used in this example.

Figure B-25. Binary Representation of the Addresses Used in Figure B-23.

VLSM Addresses for /24 for 172.16.12.0–172.16.15.255:					
172.16.12.0	10101100.00010000.000011	00	00000000		VLAN 1
172.16.13.0	10101100.00010000.000011	01	00000000		VLAN 2
172.16.14.0	10101100.00010000.000011	10	00000000		Nodes
172.16.15.0	10101100.00010000.000011	11	00000000		Not Used
VLSM Addresses for /27 for 172.16.14.0–172.16.14.255:					
172.16.14.0	10101100.00010000.000011	10	000	00000	Nodes Site A
172.16.14.32	10101100.00010000.000011	10	001	00000	Nodes Site B
172.16.14.64	10101100.00010000.000011	10	010	00000	Nodes Site C
VLSM Addresses for /30 for 172.16.14.224–172.16.14.255:					
172.16.14.224	10101100.00010000.000011	10	111	000 00	A-D Serial
172.16.14.228	10101100.00010000.000011	10	111	001 00	B-D Serial
172.16.14.232	10101100.00010000.000011	10	111	010 00	C-D Serial
172.16.14.236	10101100.00010000.000011	10	111	011 00	Not Used
172.16.14.240	10101100.00010000.000011	10	111	100 00	Not Used
172.16.14.244	10101100.00010000.000011	10	111	101 00	Not Used
172.16.14.248	10101100.00010000.000011	10	111	110 00	Not Used
172.16.14.252	10101100.00010000.000011	10	111	111 00	Not Used

Original Prefix
 Mask (VLAN) Mask 2 (Nodes) Mask 3 (Serial Links)

Another VLSM Example

This section illustrates another example of calculating VLSM addresses. In this example, you have a subnet address 172.16.32.0/20, and you need to assign addresses to a network that has 50 hosts. With this subnet address, however, you have $2^{12} - 2 = 4094$ host addresses, so you would be wasting more than 4000 IP addresses. With VLSM, you can further subnet the address 172.16.32.0/20 to give you more subnetwork addresses and fewer hosts per network, which would work better in this network topology. For example, if you subnet 172.16.32.0/20 to 172.16.32.0/26, you gain 64 (2^6) subnets, each of which can support 62 ($2^6 - 2$) hosts.

To further subnet 172.16.32.0/20 to 172.16.32.0/26, do the following, as illustrated in Figure B-26:

Figure B-26. Further Subnetting a Subnetted Address.

Subnetted Address: 172.16.32.0/20				
In Binary 10101100. 00010000.00100000.00000000				
VLSM Address: 172.16.32.0/26				
In Binary 10101100. 00010000.00100000.00000000				
1st Subnet:	10101100	.	00010000	.0010 0000.00 000000=172.16.32.0/26
2nd Subnet:	172	.	16	.0010 0000.01 000000=172.16.32.64/26
3rd Subnet:	172	.	16	.0010 0000.10 000000=172.16.32.128/26
4th Subnet:	172	.	16	.0010 0000.11 000000=172.16.32.192/26
5th Subnet:	172	.	16	.0010 0001.00 000000=172.16.33.0/26
	Network		Subnet	VLSM Subnet Host

- Step 1.** Write 172.16.32.0 in binary.
- Step 2.** Draw a vertical line between the 20th and 21st bits, as shown in [Figure B-26](#). This is the transition point between the original subnet bits and the VLSM subnet bits.
- Step 3.** Draw a vertical line between the 26th and 27th bits, as shown in [Figure B-26](#). This is the transition point between the VLSM subnet bits and the host bits.
- Step 4.** Calculate the 64 subnet addresses using the bits between the two vertical lines, from lowest to highest. [Figure B-26](#) shows the first five subnets available.

Route Summarization

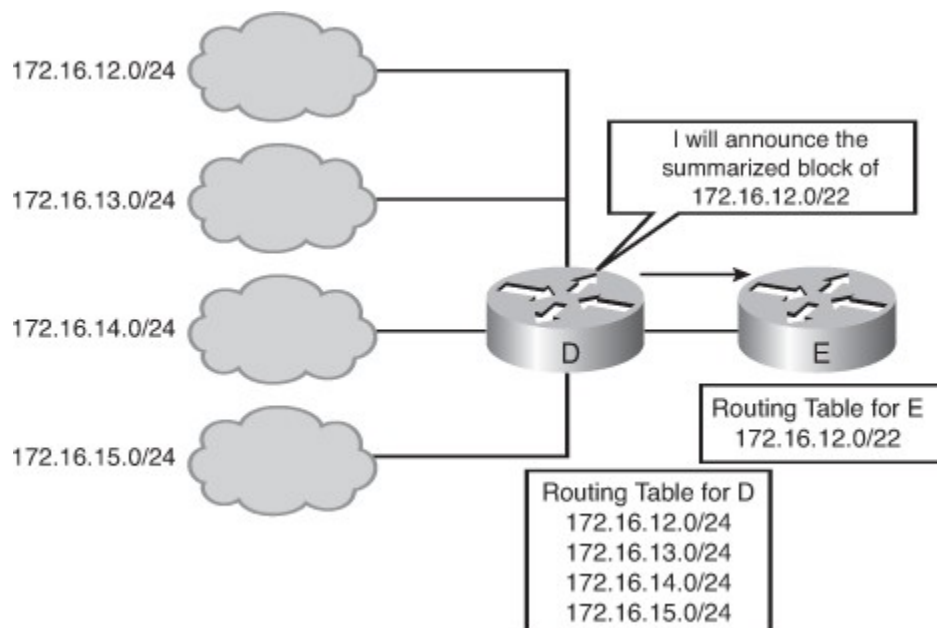
As the result of corporate expansion and mergers, the number of subnets and network addresses in routing tables is increasing rapidly. This growth taxes CPU resources, memory, and bandwidth used to maintain the routing table. Route summarization and CIDR techniques can manage this corporate growth much like Internet growth has been managed. With a thorough understanding of route summarization and CIDR, you can implement a scalable network. This section describes summarization (CIDR is covered in the later section "[Classless Interdomain Routing](#)"). The relationship between summarization and VLSM is also examined. With VLSM, you break a block of addresses into smaller subnets. In route summarization, a group of subnets is rolled up into a summarized routing table entry.

Route Summarization Overview

In large internetworks, hundreds, or even thousands, of network addresses can exist. It is often problematic for routers to maintain this volume of routes in their routing tables. As mentioned in the "[IPv4 Address Planning](#)" section earlier, route summarization can reduce the number of routes that a router must maintain, because it is a method of representing a series of network numbers in a single summary address.

For example, in [Figure B-27](#), Router D can either send four routing update entries or summarize the four addresses into a single network number. If Router D summarizes the information into a single network number entry, the following things happen:

Figure B-27. Routers Can Summarize to Reduce the Number of Routes.



- Bandwidth is saved on the link between Routers D and E.
- Router E needs to maintain only one route and therefore saves memory.
- Router E also saves CPU resources because it evaluates packets against fewer entries in its routing table.

A summary route is announced by the summarizing router as long as at least one specific route in its routing table matches the summary route.

Another advantage of using route summarization in a large, complex network is that it can isolate topology changes from other routers. For example, in [Figure B-27](#), if a specific subnet (such as 172.16.13.0/24) is *flapping* (going up and down rapidly), the summary route (172.16.12.0/22) does not change. Therefore, Router E does not need to continually modify its routing table as a result of this flapping activity.

Note

Flapping is a common term used to describe intermittent interface or link failures.

Route summarization is possible only when a proper addressing plan is in place. Route summarization is most effective within a subnetted environment when the network addresses are in contiguous blocks in powers of 2. For example, 4, 16, or 512 addresses can be represented by a single routing entry because summary masks are binary masks—just like subnet masks—so summarization must take place on binary boundaries (powers of 2). If the number of network addresses is not contiguous or not a power of 2, you can divide the addresses into groups and try to summarize the groups separately.

Routing protocols summarize or aggregate routes based on shared network numbers within the network. Classless routing protocols (such as RIPv2, OSPF, IS-IS, and EIGRP) support route summarization based on subnet addresses, including VLSM addressing. Classful routing protocols (such as RIPv1) automatically summarize routes on the classful network boundary and do not support summarization on any other bit boundaries. Classless routing protocols support summarization on any bit boundary.

Note

Summarization is described in RFC 1518, *An Architecture for IP Address Allocation with CIDR*.

Route Summarization Calculation Example

Router D in Figure B-27 has the following networks in its routing table:

- 172.16.12.0/24
- 172.16.13.0/24
- 172.16.14.0/24
- 172.16.15.0/24

To determine the summary route on Router D, determine the number of highest-order (far left) bits that match in all the addresses. To calculate the summary route, follow these steps:

Step 1. Convert the addresses to binary format and align them in a list.

Step 2. Locate the bit where the common pattern of digits ends. (It might be helpful to draw a vertical line marking the last matching bit in the common pattern.)

Step 3. Count the number of common bits. The summary route number is represented by the first IP address in the block, followed by a slash, followed by the number of common bits. As Figure B-28 illustrates, the first 22 bits of the IP addresses from 172.16.12.0 through 172.16.15.255 are the same. Therefore, the best summary route is 172.16.12.0/22.

Figure B-28. Summarizing Within an Octet, for Router D in Figure B-27.

172.16.11.0/24 =	10101100	.	00010000	.	00001011	.	00000000
172.16.12.0/24 =	172	.	16	.	000011	00	.00000000
172.16.13.0/24 =	172	.	16	.	000011	01	.00000000
172.16.14.0/24 =	172	.	16	.	000011	10	.00000000
172.16.15.0/24 =	172	.	16	.	000011	11	.00000000
172.16.15.255/24 =	172	.	16	.	000011	11	.11111111
172.16.16.0/24 =	172	.	16	.	000100	00	.00000000
				Number of Common Bits = 22 Summary: 172.16.12.0/22			
				Number of Noncommon Bits = 10			

Note

In this network, the four subnets are contiguous, and the summary route covers all the addresses in the four subnets and only those addresses. Consider, for example, what would happen if 172.16.13.0/24 were not behind Router D, but instead were used elsewhere in the network, and only the other three subnets were behind Router D. The summary route 172.16.12.0/22 should no longer be used on Router D, because it includes 172.16.13.0/24 and might result in confusing routing tables. (However, this depends on how other routers in the network summarize. If the 172.16.13.0/24 route is propagated to all routers, they choose the route with the most bits that match the destination address and should route properly. This is further described in the section "Route Summarization Operation in Cisco Routers.")

Note

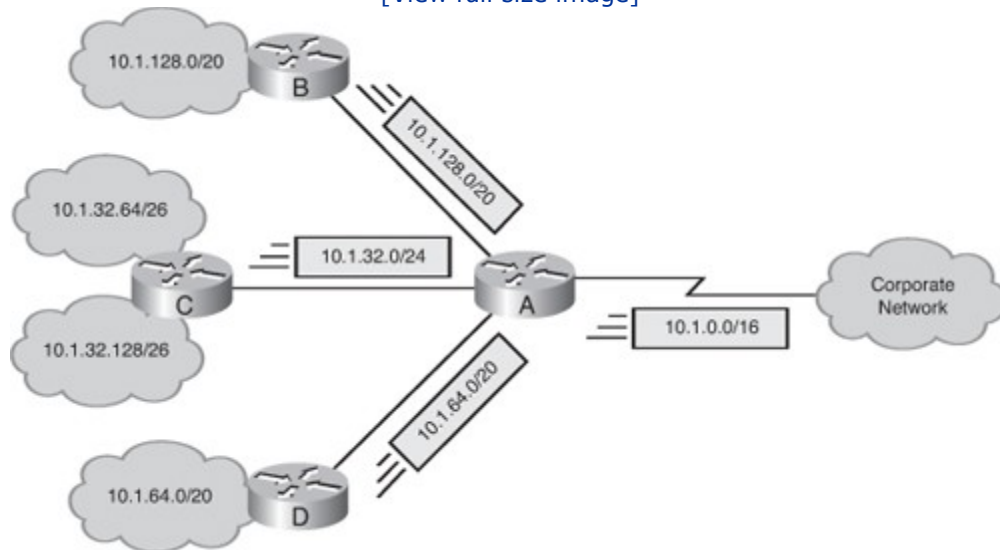
In Figure B-28, the subnets before and after the subnets to be summarized are also shown. Observe that they do not have the same first 22 bits in common and therefore are not covered by the 172.16.12.0/22 summary route.

Summarizing Addresses in a VLSM-Designed Network

A VLSM design allows for maximum use of IP addresses and more-efficient routing update communication when using hierarchical IP addressing. In Figure B-29, route summarization occurs at the following two levels:

Figure B-29. VLSM Addresses Can Be Summarized.

[\[View full size image\]](#)



- Router C summarizes two routing updates from networks 10.1.32.64/26 and 10.1.32.128/26 into a single update: 10.1.32.0/24.
- Router A receives three different routing updates. However, Router A summarizes them into a single routing update, 10.1.0.0/16, before propagating it to the corporate network.

Route Summarization Implementation

Route summarization reduces memory use on routers and routing protocol network traffic, because it results in fewer entries in the routing table (on the routers that receive the summarized routes). For summarization to work correctly, the following requirements must be met:

- Multiple IP addresses must share the same highest-order bits.
- Routing protocols must base their routing decisions on a 32-bit IP address and a prefix length that can be up to 32 bits.
- Routing updates must carry the prefix length (the subnet mask) along with the 32-bit IP address.

Route Summarization Operation in Cisco Routers

This section discusses generalities of how Cisco routers handle route summarization. Details about how route summarization operates with a specific protocol are discussed in the corresponding protocol chapter of this book.

Cisco routers manage route summarization in two ways:

- **Sending route summaries**— Routing information advertised out an interface is automatically summarized at major (classful) network address boundaries by RIP and EIGRP. Specifically, this automatic summarization occurs for routes whose classful network address differs from the major network address of the interface to which the advertisement is being sent. For OSPF and IS-IS, you must configure summarization.

Route summarization is not always a solution. You would not want to use route summarization if you needed to advertise all networks across a boundary, such as when you have discontinuous networks. When using EIGRP and RIPv2, you can disable this automatic summarization.

- **Selecting routes from route summaries**— If more than one entry in the routing table matches a particular destination, the longest prefix match in the routing table is used. Several routes might match one destination, but the longest matching prefix is used. For example, if a routing table has the paths shown in [Figure B-30](#), packets addressed to destination 172.16.5.99 are routed through the 172.16.5.0/24 path, because that address has the longest match with the destination address.

Figure B-30. Routers Use the Longest Match When Selecting a Route.

172.16.5.33	/32	host
172.16.5.32	/27	subnet
172.16.5.0	/24	network
172.16.0.0	/16	block of networks
0.0.0.0	/0	default

Note

When running classful protocols (for example, RIPv1), you must enable **ip classless** if you want the router to select a default route when it must route to an unknown subnet of a network for which it knows some subnets. See the “[The ip classless Command](#)” section in [Chapter 1](#) for more details.

Note that by default (and for historical reasons) the routing table on Cisco routers acts in a classful manner, as described in the sidebar “[The Routing Table Acts Classfully](#)” in [Chapter 1](#).

Route Summarization in IP Routing Protocols

[Table B-23](#) summarizes the route summarization support available in the various IP routing protocols.

Table B-23. Routing Protocol Route Summarization Support

Protocol	Automatic Summarization at Classful Network Boundary?	Capability to Turn Off Automatic Summarization?	Capability to Summarize at Other Than a Classful Network Boundary?
RIPv1	Yes	No	No
RIPv2	Yes	Yes	Yes
IGRP ^[1]	Yes	No	No
EIGRP	Yes	Yes	Yes
OSPF	No	—	Yes
IS-IS	No	—	Yes

^[1] Interior Gateway Routing Protocol (IGRP) is no longer supported, as of Cisco IOS Release 12.3.

Classless Interdomain Routing

CIDR is a mechanism developed to help alleviate the problem of exhaustion of IP addresses and growth of routing tables. The idea behind CIDR is that blocks of multiple addresses (for example, blocks of Class C address) can be combined, or aggregated, to create a larger classless set of IP addresses, with more hosts allowed. Blocks of Class C network numbers are allocated to each network service provider. Organizations using the network service provider for Internet connectivity are allocated subsets of the service provider's address space as required. These multiple Class C addresses can then be summarized in routing tables, resulting in fewer route advertisements. (Note that the CIDR mechanism can be applied to blocks of Class A, B, and C addresses. It is not restricted to Class C.)

Note

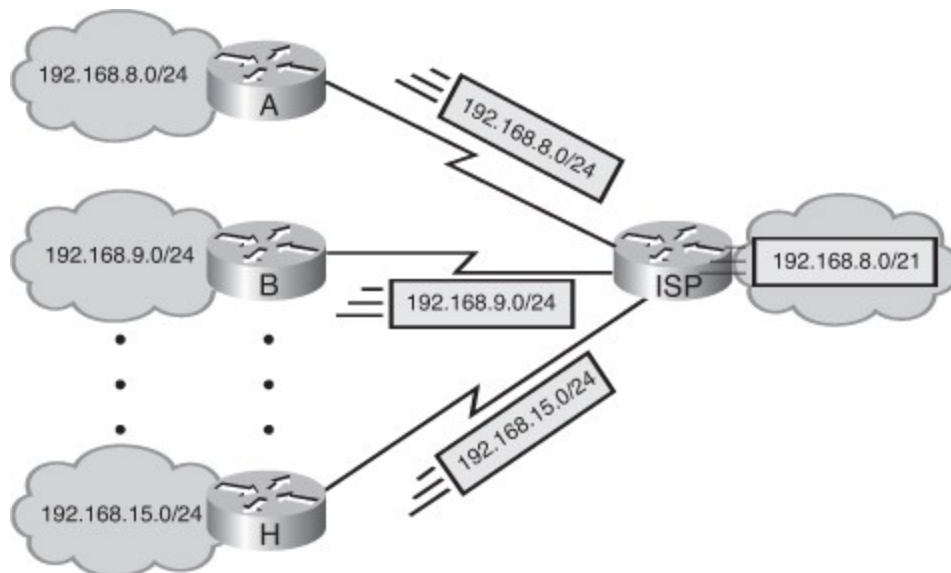
CIDR is described further in RFC 1518, *An Architecture for IP Address Allocation with CIDR*, and RFC 4632, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*. RFC 2050, *Internet Registry IP Allocation Guidelines*, specifies guidelines for the allocation of IP addresses.

Note that the difference between CIDR and route summarization is that route summarization is generally done within, or up to, a classful boundary, whereas CIDR combines several classful networks.

CIDR Example

Figure B-31 shows an example of CIDR and route summarization. The Class C network addresses 192.168.8.0/24 through 192.168.15.0/24 are being used and are being advertised to the Internet service provider (ISP) router. When the ISP router advertises the available networks, it can summarize these into one route instead of separately advertising the eight Class C networks. By advertising 192.168.8.0/21, the ISP router indicates that it can get to all destination addresses whose first 21 bits are the same as the first 21 bits of the address 192.168.8.0.

Figure B-31. CIDR Allows a Router to Summarize Multiple Class C Addresses.



The mechanism used to calculate the summary route to advertise is the same as shown in the earlier "Route Summarization" section. The Class C network addresses 192.168.8.0/24 through 192.168.15.0/24 are being used and are being advertised to the ISP router. To summarize these addresses, find the common bits, as shown here (in bold):

192.168.8.0 192.168.**00001**000.00000000

192.168.9.0	192.168. 00001 001.00000000
192.168.10.0	192.168. 00001 010.00000000
...	
192.168.14.0	192.168. 00001 110.00000000
192.168.15.0	192.168. 00001 111.00000000

The route 192.168.00001xxx.xxxxxxxx or 192.168.8.0/21 (also written as 192.168.8.0 255.255.248.0) summarizes these eight routes.

In this example, the first octet is 192, which identifies the networks as Class C networks. Combining these Class C networks into a block of addresses with a mask of less than /24 (the default Class C network mask) indicates that CIDR, not route summarization, is being performed.

In this example, the eight separate 192.168.x.0 Class C networks that have the prefix /24 are combined into a single summarized block of 192.168.8.0/21. (At some other point in the network, this summarized block may be further combined into 192.168.0.0/16, and so on.)

Consider another example. A company that uses four Class B networks has the IP addresses 172.16.0.0/16 for Division A, 172.17.0.0/16 for Division B, 172.18.0.0/16 for Division C, and 172.19.0.0/16 for Division D. They can all be summarized as a single block: 172.16.0.0/14. This one entry represents the whole block of four Class B networks. This process is CIDR. The summarization goes beyond the Class B boundaries.

Appendix C. BGP Supplement

This appendix contains supplementary Border Gateway Protocol (BGP) information and covers the following topics:

- [BGP Route Summarization](#)
- [Redistribution with IGP](#)s
- [Communities](#)
- [Route Reflectors](#)

This appendix provides you with some additional information about the Border Gateway Protocol (BGP).

BGP Route Summarization

This section reviews classless interdomain routing (CIDR) and describes how BGP supports CIDR and summarization of addresses. Both the **network** and **aggregate-address** commands are described.

CIDR and Aggregate Addresses

As discussed in [Appendix B](#), "IPv4 Supplement," CIDR is a mechanism developed to help alleviate the problem of exhaustion of IP addresses and the growth of routing tables. The idea behind CIDR is that blocks of multiple addresses (for example, blocks of Class C address) can be combined, or aggregated, to create a larger classless set of IP addresses. These multiple addresses can then be summarized in routing tables, resulting in fewer route advertisements.

Earlier versions of BGP did not support CIDR. BGP Version 4 (BGP-4) does. BGP-4 support includes the following:

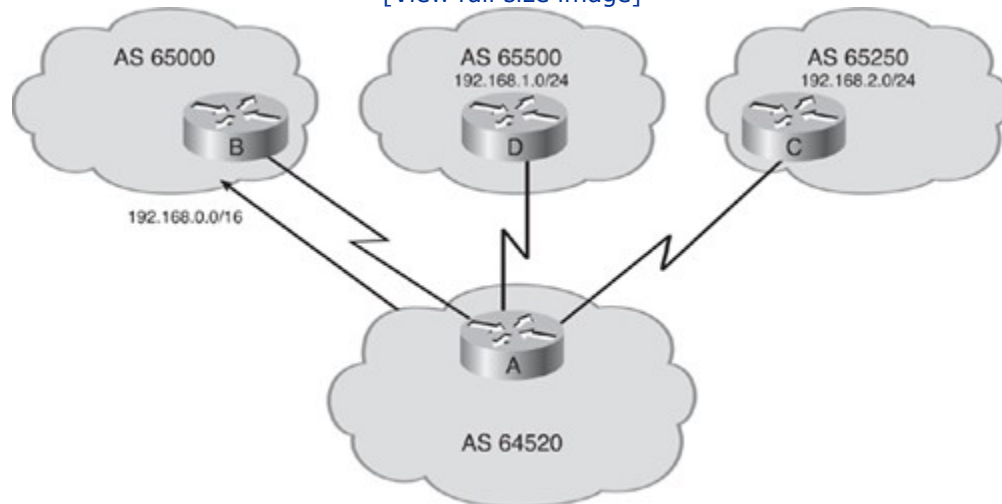
- The BGP update message includes both the prefix and the prefix length. Previous versions included only the prefix. The length was assumed from the address class.

- Addresses can be aggregated when advertised by a BGP router.
- The autonomous system (AS)-path attribute can include a combined unordered list of all autonomous systems that all the aggregated routes have passed through. This combined list should be considered to ensure that the route is loop-free.

For example, in [Figure C-1](#), Router C is advertising network 192.168.2.0/24, and Router D is advertising network 192.168.1.0/24. Router A could pass those advertisements to Router B. However, Router A could reduce the size of the routing tables by aggregating the two routes into one (for example, 192.168.0.0/16).

Figure C-1. Using CIDR with BGP.

[\[View full size image\]](#)



Note

In [Figure C-1](#), the aggregate route that Router A is sending covers more than the two routes from Routers C and D. The example assumes that Router A also has jurisdiction over all the other routes covered by this aggregate route.

Two BGP attributes are related to aggregate addressing:

- **Atomic aggregate**— A well-known discretionary attribute that informs the neighbor autonomous system that the originating router has aggregated the routes
- **Aggregator**— An optional transitive attribute that specifies the BGP router ID and autonomous system number of the router that performed the route aggregation

By default, the aggregate route is advertised as coming from the autonomous system that did the aggregation and has the atomic aggregate attribute set to show that information might be missing. The autonomous system numbers from the nonaggregated routes are not listed.

You can configure the router to include the unordered list of all autonomous systems contained in all paths that are being summarized.

Note

Indications are that aggregate addresses are not used in the Internet as much as they could be because autonomous systems that are multihomed (connected to more than one Internet service provider [ISP]) want to make sure that their routes are advertised without being aggregated into a summarized route.

In [Figure C-1](#), by default the aggregated route 192.168.0.0/16 has an autonomous system path attribute of (64520). If Router A were configured to include the combined unordered list, it would include the set {65250 65500} and (64520) in the AS-path attribute. The AS-path would be the unordered set {64520 65250 65500}.

Network Boundary Summarization

BGP was originally not intended to be used to advertise subnets. Its intended purpose was to advertise classful, or better, networks. *Better* in this case means that BGP can summarize blocks of individual classful networks into a few large blocks that represent the same address space as the individual network blocks—in other words, CIDR blocks. For example, 32 contiguous Class C networks can be advertised individually as 32 separate entries, with each having a network mask of /24. Or it might be possible to announce these same networks as a single entry with a /19 mask.

Consider how other protocols handle summarization. The Routing Information Protocol Version 1 (RIPv1), Routing Information Protocol Version 2 (RIPv2), and Enhanced Interior Gateway Routing Protocol (EIGRP) protocols all summarize routes on the classful network boundary by default. In contrast, Open Shortest Path First (OSPF) and Intermediate System-to-Intermediate System (IS-IS) do not summarize by default, but you can configure summarization manually.

You can turn off autosummarization for RIPv2 and EIGRP. For example, if you are assigned a portion of a Class A, B, or C address, summarization needs to be turned off. Otherwise, you risk claiming ownership of the whole Class A, B, or C address.

Note

The Internet Assigned Numbers Authority (IANA) is reclaiming Class A addresses from organizations that no longer need them. IANA breaks these Class A addresses into blocks of /19 address space, which are assigned to various ISPs to be given out in place of Class C addresses. This process has helped make the Internet a classless environment.

BGP works differently than the other protocols. As discussed in [Chapter 6](#), “Implementing a Border Gateway Protocol Solution for ISP Connectivity,” the **network network-number [mask network-mask] [route-map map-tag]** router configuration command for BGP permits BGP to advertise a network if it is present in the IP routing table. This command allows classless prefixes. The router can advertise individual subnets, networks, or supernets. The default mask is the classful mask and results in only the classful network number being announced. Note that at least one subnet of the specified major network must be present in the IP routing table for BGP to start announcing the classful network. However, if you specify the *network-mask*, an exact match to the network (both address and mask) must exist in the routing table for the network to be advertised.

The BGP **auto-summary** command determines how BGP handles redistributed routes. The **no auto-summary** router configuration command turns off BGP autosummarization. When summarization is enabled (with **auto-summary**), all redistributed subnets are summarized to their classful boundaries in the BGP table. When summarization is disabled (with **no auto-summary**), all redistributed subnets are present in their original form in the BGP table. For example, if an ISP assigns a network of 10.100.50.0/24 to an autonomous system, and that autonomous system then uses the **redistribute connected** command to introduce this network into BGP, BGP announces that the autonomous system owns 10.0.0.0/8 if the **auto-summary** command is on. To the Internet, it appears as if this autonomous system owns all the Class A network 10.0.0.0/8, which is not true. Other organizations that own a portion of the 10.0.0.0/8 address space might have connectivity problems because of this autonomous system claiming ownership for the whole block of addresses. This outcome is undesirable if the autonomous system does not own the entire address space. Using the **network 10.100.50.0 mask 255.255.255.0** command rather than the **redistribute connected** command ensures that this assigned network is announced correctly.

Caution

Recall that in Cisco IOS Release 12.2(8)T, the default behavior of the **auto-summary** command was changed to disabled. In other words

- Before 12.2(8)T, the default is **auto-summary**.

- Starting in 12.2(8)T, the default is **no auto-summary**.

BGP Route Summarization Using the network Command

To advertise a simple classful network number, use the **network network-number** router configuration command without the **mask** option. To advertise an aggregate of prefixes that originate in this autonomous system, use the **network network-number [mask network-mask]** router configuration command with the **mask** option (but remember that the prefix must exactly match [both address and mask] an entry in the IP routing table for the network to be advertised).

When BGP has a **network** command for a classful address and it has at least one subnet of that classful address space in its routing table, it announces the classful network and not the subnet. For example, if a BGP router has network 172.16.22.0/24 in the routing table as a directly connected network, and a BGP **network 172.16.0.0** command, BGP announces the 172.16.0.0/16 network to all neighbors. If 172.16.22.0 is the only subnet for this network in the routing table and it becomes unavailable, BGP will withdraw 172.16.0.0/16 from all neighbors. If instead the command **network 172.16.22.0 mask 255.255.255.0** is used, BGP will announce 172.16.22.0/24 and not 172.16.0.0/16.

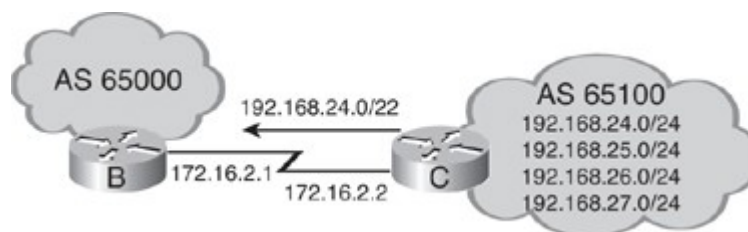
For BGP, the **network** command requires that there be an exact match in the routing table for the prefix or mask that is specified. This exact match can be accomplished by using a static route with a null 0 interface, or it might already exist in the routing table, such as because of the Interior Gateway Protocol (IGP) performing the summarization.

Cautions When Using the network Command for Summarization

The **network** command tells BGP what to advertise but not how to advertise it. When using the BGP **network** command, the network number specified must also be in the IP routing table before BGP can announce it.

For example, consider Router C in Figure C-2. It has the group of addresses 192.168.24.0/24, 192.168.25.0/24, 192.168.26.0/24, and 192.168.27.0/24 already in its routing table. The configuration in Example C-1 is put on Router C.

Figure C-2. BGP Network for Summarization Examples.



Example C-1. Sample BGP Configuration for Router C in Figure C-2

```
router bgp 65100
 network 192.168.24.0
 network 192.168.25.0
 network 192.168.26.0
 network 192.168.27.0
 network 192.168.24.0 mask 255.255.252.0
 neighbor 172.16.2.1 remote-as 65000
```

Each of the four Class C networks is announced because each already exists in the routing table. These networks are summarized with the **network 192.168.24.0 mask 255.255.252.0** command on Router C.

However, with this command the 192.168.24.0/22 route is *not* announced by default because that route is not in the routing table. If the IGP supports manual summarization (as EIGRP or OSPF do), and the same summarization is performed by the IGP command, BGP announces that summarized route. If route summarization is not performed with the IGP, and BGP is required to announce this route, a static route should be created that allows this network to be installed in the routing table.

The static route should point to the null 0 interface (using the command **ip route 192.168.24.0 255.255.252.0 null0**). Remember that 192.168.24.0/24, 192.168.25.0/24, 192.168.26.0/24, and 192.168.27.0/24 addresses are already in the routing table. This command creates an additional entry of 192.168.24.0/22 as a static route to null 0. If a network, such as 192.168.25.0/24, becomes unreachable, and packets arrive for 192.168.25.1, the destination address is compared to the current entries in the routing table using the longest-match criteria. Because 192.168.25.0/24 no longer exists in the routing table, the best match is 192.168.24.0/22, which points to the null 0 interface. The packet is sent to the null 0 interface, and an Internet Control Message Protocol (ICMP) unreachable message is generated and sent to the packet's originator. Dropping these packets prevents traffic from using up bandwidth following a default route that is either deeper into your autonomous system or (in a worst-case scenario) back out to the ISP (when the ISP would route it back to the autonomous system because of the summarized route advertised to the ISP, causing a routing loop).

In this example, five networks are announced using **network** commands: the four Class C networks plus the summary route. The purpose of summarization is to reduce the advertisement's size, and the size of the Internet routing table. Announcing these more specific networks along with the summarized route actually increases the table's size.

Example C-2 shows a more efficient configuration. A single entry represents all four networks, and a static route to null 0 installs the summarized route in the IP routing table so that BGP can find a match. By using this **network** command, the autonomous system 65100 router advertises a summarized route for the four Class C addresses (192.168.24.0/24, 192.168.25.0/24, 192.168.26.0/24, and 192.168.27.0/24) assigned to the autonomous system. For this new **network** command (192.168.24.0/22) to be advertised, it must first appear in the local routing table. Because only the more specific networks exist in the IP routing table, a static route pointing to null 0 has been created to allow BGP to announce this network (192.168.24.0/22) to autonomous system 65000.

Example C-2. More-Efficient BGP Configuration for Router C in Figure C-2

```
router bgp 65100
 network 192.168.24.0 mask 255.255.252.0
 neighbor 172.16.2.1 remote-as 65000
 ip route 192.168.24.0 255.255.252.0 null 0
```

Although this configuration works, the **network** command itself was not designed to perform summarization. The **aggregate-address** command, described in the next section, was designed to perform summarization.

Creating a Summary Address in the BGP Table Using the aggregate-address Command

The **aggregate-address ip-address mask [summary-only] [as-set]** router configuration command is used to create an aggregate, or summary, entry in the BGP table. The parameters of this command are described in Table C-1.

Table C-1. aggregate-address Command Description

Parameter	Description
<i>ip-address</i>	Identifies the aggregate address to be created.
<i>mask</i>	Identifies the mask of the aggregate address to be created.
summary-only	(Optional) Causes the router to advertise only the aggregated route. The default is to advertise both the aggregate and the more specific routes.

Parameter	Description
as-set	(Optional) Generates autonomous system path information with the aggregate route to include all the autonomous system numbers listed in all the paths of the more specific routes. The default for the aggregate route is to list only the autonomous system number of the router that generated the aggregate route.

Notice the difference between the **aggregate-address** and the **network** command:

- The **aggregate-address** command aggregates only networks that are already in the *BGP table*.
- With the BGP **network** command, the network must exist in the *IP routing table* for the summary network to be advertised.

When you use the **aggregate-address** command without the **as-set** keyword, the aggregate route is advertised as coming from your autonomous system, and the atomic aggregate attribute is set to show that information might be missing. The atomic aggregate attribute is set unless you specify the **as-set** keyword.

Without the **summary-only** keyword, the router still advertises the individual networks. This can be useful for redundant ISP links. For example, if one ISP is advertising only summaries, and the other is advertising a summary plus the more specific routes, the more specific routes are followed. However, if the ISP advertising the more specific routes becomes inaccessible, the other ISP advertising only the summary is followed.

When the **aggregate-address** command is used, a BGP route to null 0 is automatically installed in the IP routing table for the summarized route

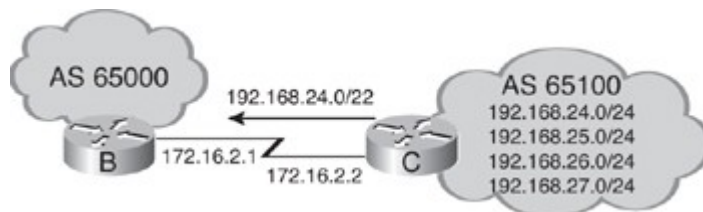
If any route already in the BGP table is within the range indicated by the **aggregate-address**, the summary route is inserted into the BGP table and is advertised to other routers. This process creates more information in the BGP table. To get any benefits from the aggregation, the more-specific routes covered by the route summary should be suppressed using the **summary-only** option. When the more specific routes are suppressed, they are still present in the BGP table of the router doing the aggregation. However, because the routes are marked as suppressed, they are never advertised to any other router.

For BGP to announce a summary route using the **aggregate-address** command, at least one of the more specific routes must be in the BGP table. This is usually a result of having **network** commands for those routes.

If you use only the **summary-only** keyword on the **aggregate-address** command, the summary route is advertised, and the path indicates only the autonomous system that did the summarization (all other path information is missing). If you use only the **as-set** keyword on the **aggregate-address** command, the set of autonomous system numbers is included in the path information (and the command with the **summary-only** keyword is deleted if it existed). However, you may use *both* keywords on one command. This causes only the summary address to be sent and all the autonomous systems to be listed in the path information.

Figure C-3 illustrates a sample network (it is the same network as in Figure C-2, repeated here for your convenience). Example C-3 shows the configuration of Router C using the **aggregate-address**.

Figure C-3. BGP Network for Summarization Examples.



Example C-3. Configuration for Router C in Figure C-3 Using the *aggregate-address* Command

```
router bgp 65100
 network 192.168.24.0
 network 192.168.25.0
 network 192.168.26.0
 network 192.168.27.0
 neighbor 172.16.2.1 remote-as 65000
 aggregate-address 192.168.24.0 255.255.252.0 summary-only
```

This configuration on Router C shows the following:

- **router bgp 65100**— Configures a BGP process for autonomous system 65100. This part of the configuration describes *what* to advertise
- **network commands**— Configure BGP to advertise the four Class C networks in autonomous system 65100.
- **neighbor 172.16.2.1 remote-as 65000**— Specifies the router at this address (Router B) as a neighbor in autonomous system 65000. This part of the configuration describes *where* to send the advertisements.
- **aggregate-address 192.168.24.0 255.255.252.0 summary-only**— Specifies the aggregate route to be created but suppresses advertisements of more specific routes to all neighbors. This part of the configuration describes *how* to advertise. Without the **summary-only** option, the new summarized route would be advertised along with the more specific routes. In this example, however, Router B receives only one route (192.168.24.0/22) from Router C. The **aggregate-address** command tells the BGP process to perform route summarization and automatically installs the null route representing the new summarized route.

The following summarizes the differences between the main BGP commands:

- The **network** command tells BGP *what* to advertise.
- The **neighbor** command tells BGP *where* to advertise.
- The **aggregate-address** command tells BGP *how* to advertise the networks.

The **aggregate-address** command does not replace the **network** command. At least one of the more specific routes to be summarized must be in the BGP table. In some situations, the more-specific routes are injected into the BGP table by other routers, and the aggregation is done in another router or even in another autonomous system. This approach is called *proxy aggregation*. In this case, the aggregation router needs only the proper **aggregate-address** command, not the **network** commands, to advertise the more specific routes.

The **show ip bgp** command provides information about route summarization and displays the local router ID, the networks recognized by the BGP process, the accessibility to remote networks, and autonomous system path information. In [Example C-4](#), notice the *s* in the first column for the lower four networks. These networks are being suppressed. They were learned from a **network** command on this router. The next-hop address is 0.0.0.0, which indicates that this router created these entries in BGP. Notice that this router also created the summarized route 192.168.24.0/22 in BGP (this route also has a next hop of 0.0.0.0, indicating that this router created it). The more-specific routes are suppressed, and only the summarized route is announced.

Example C-4. *show ip bgp* Command Output with Routes Suppressed

Code View: Scroll / [Show All](#)

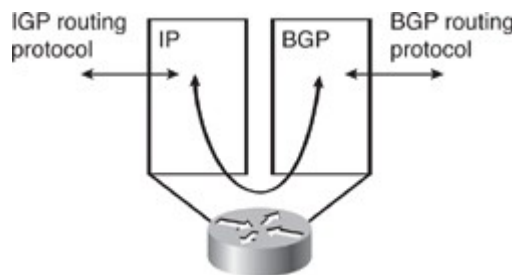
```
RouterC#show ip bgp
BGP table version is 28, local router ID is 172.16.2.1
```

Status codes: s = suppressed, * = valid, > = best, and i = internal					
Origin codes : i = IGP, e = EGP, and ? = incomplete					
Network	Next Hop	Metric	LocPrf	Weight	Path
*>192.168.24.0/22	0.0.0.0	0		32768	i
s>192.168.24.0	0.0.0.0	0		32768	i
s>192.168.25.0	0.0.0.0	0		32768	i
s>192.168.26.0	0.0.0.0	0		32768	i
s>192.168.27.0	0.0.0.0	0		32768	i

Redistribution with IGP

Chapter 4, “Manipulating Routing Updates,” discusses route redistribution and how it is configured. This section examines the specifics of when redistribution between BGP and IGPs is appropriate. As noted in Chapter 6, and as shown in Figure C-4, a router running BGP keeps a table of BGP information, separate from the IP routing table. The router offers the best routes from the BGP table to the IP routing table and can be configured to share information between the two tables (by redistribution).

Figure C-4. Router Running BGP Keeps Its Own Table, Separate from the IP Routing Table.



Advertising Networks into BGP

Route information is sent from an autonomous system into BGP in one of the following ways:

- **Using the network command**— As discussed, the **network** command allows BGP to advertise a network that is already in the IP table. The list of **network** commands must include all the networks in the autonomous system you want to advertise.
- **By redistributing static routes to interface null 0 into BGP**— Redistribution occurs when a router running different protocols advertises routing information received by one protocol to the other protocol. Static routes in this case are considered a protocol, and static information is advertised to BGP. (The use of the null 0 interface is discussed in the earlier section “[Cautions When Using the network Command for Summarization](#).”)
- **By redistributing dynamic IGP routes into BGP**— This solution is not recommended because it might cause instability.

Redistributing from an IGP into BGP is not recommended because any change in the IGP routes—for example, if a link goes down—might cause a BGP update. This method could result in unstable BGP tables.

If redistribution is used, care must be taken that only local routes are redistributed. For example, routes learned from other autonomous systems (that were learned by redistributing BGP into the IGP) must not be sent out again from the IGP. Otherwise, routing loops could result. Configuring this filtering can be complex.

Using a **redistribute** command into BGP results in an incomplete origin attribute for the route, as indicated by the **?** in the **show ip bgp** command output.

Advertising from BGP into an IGP

Route information may be sent from BGP into an autonomous system by redistributing the BGP routes into the IGP.

Because BGP is an external routing protocol, care must be taken when exchanging information with internal protocols because of the amount of information in BGP tables.

For ISP autonomous systems, redistributing from BGP normally is not required. Other autonomous systems may use redistribution, but the number of routes means that filtering normally is required. Each of these situations is examined in the following sections.

ISP: No Redistribution from BGP into IGP Is Required

An ISP typically has all routers in the autonomous system (or at least all routers in the transit path within the autonomous system) running BGP. Of course, this would be a full-mesh Internal BGP (IBGP) environment, and IBGP would be used to carry the External BGP (EBGP) routes across the autonomous system. All the BGP routers in the autonomous system would be configured with the **no synchronization** command (which is on by default in Cisco IOS Software Release 12.2(8)T and later), because synchronization between IGP and BGP is not required. The BGP information then would not need to be redistributed into the IGP. The IGP would need to route only information local to the autonomous system and routes to the next-hop addresses of the BGP routes.

One advantage of this approach is that the IGP protocol does not have to be concerned with all the BGP routes. BGP takes care of them. BGP also converges faster in this environment because it does not have to wait for the IGP to advertise the routes.

Non-ISP: Redistribution from BGP into IGP Might Be Required

A non-ISP autonomous system typically does not have all routers in the autonomous system running BGP, and it might not have a full-mesh IBGP environment. If this is the case, and if knowledge of external routes is required inside the autonomous system, redistributing BGP into the IGP is necessary. However, because of the number of routes that would be in the BGP tables, filtering normally is required.

As discussed in the “[BGP Multihoming Options](#)” section in [Chapter 6](#), an alternative to receiving full routes from BGP is that the ISP could send only default routes, or default routes and some external routes, to the autonomous system.

Note

An example of when redistributing into an IGP might be necessary is in an autonomous system that is running BGP only on its border routers and that has other routers in the autonomous system that do not run BGP but that require knowledge of external routes.

Communities

As discussed in [Chapter 6](#), BGP communities are another way to filter incoming or outgoing BGP routes. Distribute lists and prefix lists are cumbersome to configure for a large network with a complex routing

policy. For example, individual **neighbor** statements and access lists or prefix lists have to be configured for each neighbor on each router involved in the policy.

The BGP communities function allows routers to tag routes with an indicator (the *community*) and allows other routers to make decisions (filter) based on that tag. BGP communities are used for destinations (routes) that share some common properties and that, therefore, share common policies; routers, therefore, act on the community, rather than on individual routes. Communities are not restricted to one network or autonomous system, and they have no physical boundaries.

Community Attribute

The community attribute is an optional transitive attribute. If a router does not understand the concept of communities, it passes it on to the next router. However, if the router does understand the concept, it must be configured to propagate the community. Otherwise, communities are dropped by default.

Each network can be a member of more than one community.

The community attribute is a 32-bit number. It can have a value in the range 0 to 4,294,967,200. The upper 16 bits indicate the autonomous system number of the autonomous system that defined the community. The lower 16 bits are the community number and have local significance. The community value can be entered as one decimal number or in the format *AS:nn*, where *AS* is the autonomous system number, and *nn* is the lower 16-bit local number. The community value is displayed as one decimal number by default.

Setting and Sending the Communities Configuration

Route maps can be used to set the community attributes.

The **set community** {[*community-number*] [*well-known-community*] [**additive**]} | **none** route map configuration command is used within a route map to set the BGP community attribute. The parameters of this command are described in [Table C-2](#).

Table C-2. set community Command Description

Parameter	Description
<i>community-number</i>	The community number. Values are 1 to 4,294,967,200.
<i>well-known-community</i>	The following are predefined, well-known community numbers: internet— Advertises this route to the Internet community and any router that belongs to it no-export— Does not advertise to EBGp peers no-advertise— Does not advertise this route to any peer local-AS— Does not send outside the local autonomous system
additive	(Optional) Specifies that the community is to be added to the existing communities.
none	Removes the community attribute from the prefixes that pass the

Parameter	Description
	route map.

The **set community** command is used along with the **neighbor route-map** command to apply the route map to updates.

The **neighbor** {*ip-address* | *peer-group-name*} **send-community** router configuration command is used to specify that the BGP communities attribute should be sent to a BGP neighbor. Table C-3 explains the parameters of this command.

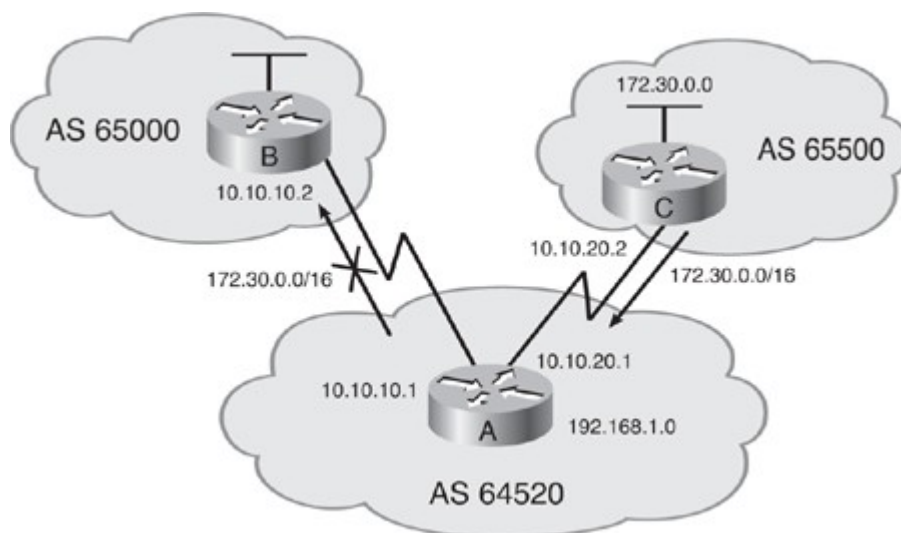
Table C-3. neighbor send-community Command Description

Parameter	Description
<i>ip-address</i>	The IP address of the BGP neighbor to which the communities attribute is sent.
<i>peer-group-name</i>	The name of a BGP peer group.

By default, the communities attribute is not sent to any neighbor. (Communities are stripped in outgoing BGP updates.)

In Figure C-5, Router C is sending BGP updates to Router A, but it does not want Router A to propagate these routes to Router B.

Figure C-5. Network for BGP Communities Example.



Example C-5 shows the configuration for Router C in this example. Router C sets the community attribute in the BGP routes that it is advertising to Router A. The **no-export** community attribute is used to indicate that Router A should not send the routes to its external BGP peers.

Example C-5. Configuration of Router C in Figure C-5

```
router bgp 65500
 network 172.30.0.0
 neighbor 10.10.20.1 remote-as 64520
```

```

neighbor 10.10.20.1 send-community
neighbor 10.10.20.1 route-map SETCOMM out
!
route-map SETCOMM permit 10
match ip address 1
set community no-export
!
access-list 1 permit 0.0.0.0 255.255.255.255

```

In this example, Router C has one neighbor, 10.10.20.1 (Router A). When communicating with Router A, the community attribute is sent, as specified by the **neighbor send-community** command. The route map SETCOMM is used when sending routes to Router A to set the community attribute. Any route that matches **access-list 1** has the community attribute set to **no-export**. Access list 1 permits any routes. Therefore, all routes have the community attribute set to **no-export**.

In this example, Router A receives all of Router C's routes but does not pass them to Router B.

Using the Communities Configuration

The **ip community-list** *community-list-number* {**permit** | **deny**} *community-number* global configuration command is used to create a community list for BGP and to control access to it. The parameters of this command are described in [Table C-4](#).

Table C-4. *ip community-list* Command Description

Parameter	Description
<i>community-list-number</i>	The community list number, in the range of 1 to 99
permit deny	Permits or denies access for a matching condition
<i>community-number</i>	The community number or well-known-community configured by a set community command

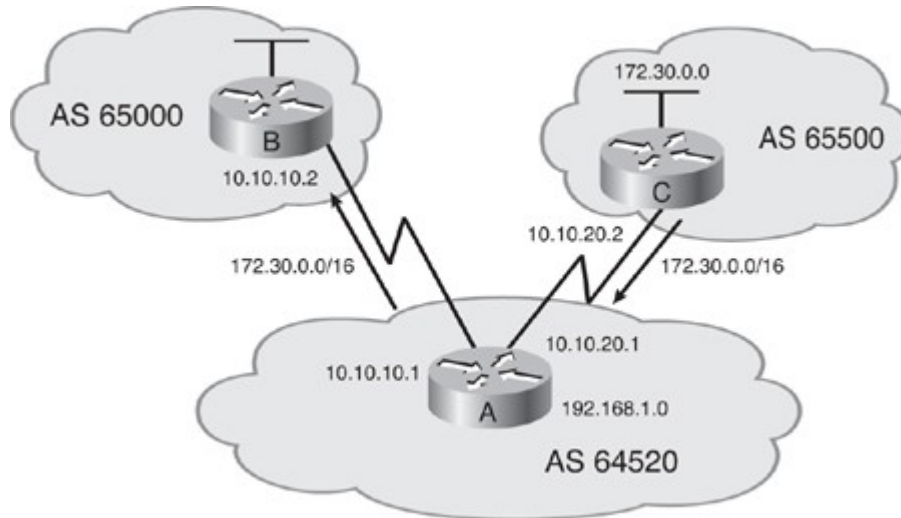
The **match community** *community-list-number* [**exact**] route map configuration command enables you to match a BGP community attribute to a value in a community list. The parameters of this command are described in [Table C-5](#).

Table C-5. *match community* Command Description

Parameter	Description
<i>community-list-number</i>	The community list number, in the range of 1 to 99, that is used to compare the community attribute.
exact	(Optional) Indicates that an exact match is required. All the communities and only those communities in the community list must be present in the community attribute.

In [Figure C-6](#), Router C is sending BGP updates to Router A. Router A sets the weight of these routes based on the community value set by router C.

Figure C-6. Network for BGP Communities Example Using Weight.



Example C-6 shows the configuration of Router C in Figure C-6. Router C has one neighbor, 10.10.20.1 (Router A).

Example C-6. Configuration of Router C in Figure C-6

```
router bgp 65500
 network 172.30.0.0
 neighbor 10.10.20.1 remote-as 64520
 neighbor 10.10.20.1 send-community
 neighbor 10.10.20.1 route-map SETCOMM out
!
route-map SETCOMM permit 10
 match ip address 1
 set community 100 additive
!
access-list 1 permit 0.0.0.0 255.255.255.255
```

In this example, the community attribute is sent to Router A, as specified by the **neighbor send-community** command. The route map SETCOMM is used when sending routes to Router A to set the community attribute. Any route that matches access list 1 has community 100 added to the existing communities in the route's community attribute. In this example, access list 1 permits any routes. Therefore, all routes have 100 added to the list of communities. If the **additive** keyword in the **set community** command is not set, 100 replaces any old community that already exists. Because the keyword **additive** is used, the 100 is added to the list of communities that the route is part of.

Example C-7 shows the configuration of Router A in Figure C-6.

Example C-7. Configuration of Router A in Figure C-6

```
router bgp 64520
 neighbor 10.10.20.2 remote-as 65500
 neighbor 10.10.20.2 route-map CHKCOMM in
!
route-map CHKCOMM permit 10
 match community 1
 set weight 20
```

```
route-map CHKCOMM permit 20
  match community 2
!
ip community-list 1 permit 100
ip community-list 2 permit internet
```

Note

Other **router bgp** configuration commands for Router A are not shown in [Example C-9](#).

In this example, Router A has a neighbor, 10.10.20.2 (Router C). The route map CHKCOMM is used when receiving routes from Router C to check the community attribute. Any route whose community attribute matches community list 1 has its weight attribute set to 20. Community list 1 permits routes with a community attribute of 100. Therefore, all routes from Router C (which all have 100 in their list of communities) have their weight set to 20.

In this example, any route that does not match community list 1 is checked against community list 2. Any route matching community list 2 is permitted but does not have any of its attributes changed. Community list 2 specifies the **internet** keyword, which means all routes.

The sample output shown in [Example C-8](#) is from Router A in [Figure C-6](#). The output shows the details about the route 172.30.0.0 from Router C, including that its community attribute is 100, and its weight attribute is now 20.

Example C-8. *show ip bgp* Output from Router A in Figure C-6

Code View: Scroll / [Show All](#)

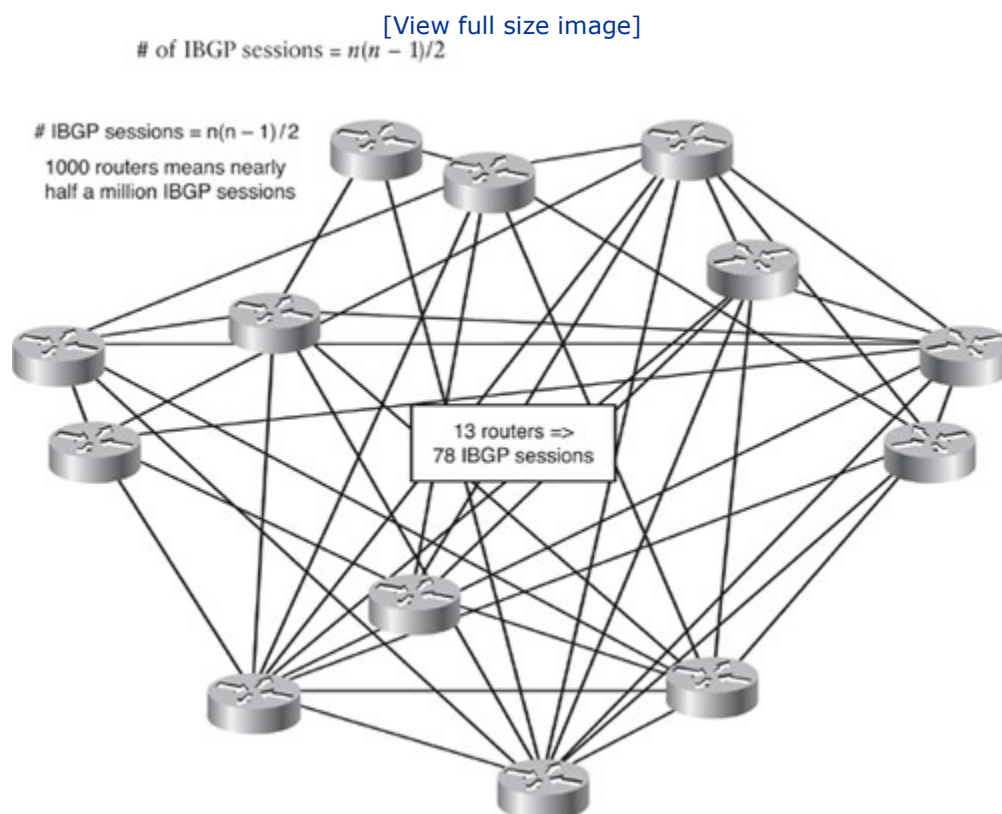
```
RtrA #show ip bgp 172.30.0.0/16
BGP routing Table Cntry for 172.30.0.0/16, version 2
Paths: (1 available, best #1)
  Advertised to non peer-group peers:
    10.10.10.2
  65500
    10.10.20.2 from 10.10.20.2 (172.30.0.1)
      Origin IGP, metric 0, localpref 100, weight 20, valid, external, best, ref 2
    Community: 100
```

Route Reflectors

BGP specifies that routes learned via IBGP are never propagated to other IBGP peers. The result is that a full mesh of IBGP peers is required within an autonomous system. As Figure C-7 illustrates, however, a full mesh of IBGP is not scalable. With only 13 routers, 78 IBGP sessions would need to be maintained. As the number of routers increases, so does the number of sessions required, governed by the following formula, in which n is the number of routers:

$$\# \text{ of IBGP sessions} = n(n - 1)/2$$

Figure C-7. Full-Mesh IBGP Requires Many Sessions and, Therefore, Is Not Scalable.

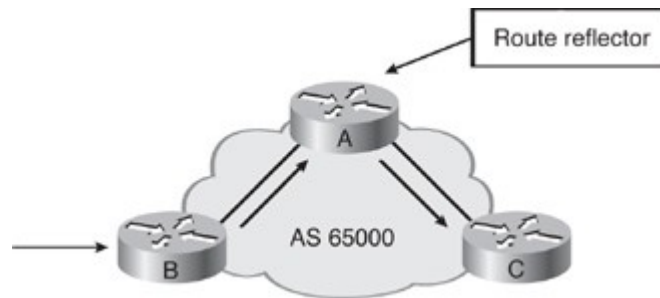


In addition to the number of BGP TCP sessions that must be created and maintained, the amount of routing traffic might also be a problem. Depending on the autonomous system topology, traffic might be replicated many times on some links as it travels to each IBGP peer. For example, if the physical topology of a large autonomous system includes some WAN links, the IBGP sessions running over those links might consume a significant amount of bandwidth.

A solution to this problem is the use of route reflectors (RRs). This section describes what an RR is, how it works, and how to configure it.

RRs modify the BGP rule by allowing the router configured as the RR to propagate routes learned by IBGP to other IBGP peers, as illustrated in Figure C-8.

Figure C-8. When Router A Is a Route Reflector, It Can Propagate Routes Learned via IBGP from Router B to Router C.



This saves on the number of BGP TCP sessions that must be maintained and also reduces the BGP routing traffic.

Route Reflector Benefits

With a BGP RR configured, a full mesh of IBGP peers is no longer required. The RR is allowed to propagate IBGP routes to other IBGP peers. RRs are used mainly by ISPs when the number of internal **neighbor** statements becomes excessive. Route reflectors reduce the number of BGP neighbor relationships in an autonomous system (thus, saving on TCP connections) by having key routers replicate updates to their RR clients.

Route reflectors do not affect the paths that IP packets follow. Only the path that routing information is distributed on is affected. However, if RRs are configured incorrectly, routing loops might result, as shown in the example later in this appendix in the “[Route Reflector Migration Tips](#)” section.

An autonomous system can have multiple RRs, both for redundancy and for grouping to further reduce the number of IBGP sessions required.

Migrating to RRs involves a minimal configuration and does not have to be done all at one time, because routers that are not RRs can coexist with RRs within an autonomous system.

Route Reflector Terminology

A *route reflector* is a router that is configured to be the router allowed to advertise (or reflect) routes it learned via IBGP to other IBGP peers. The RR has a partial IBGP peering with other routers, which are called *clients*. Peering between the clients is not needed, because the route reflector passes advertisements between the clients.

The combination of the RR and its clients is called a *cluster*.

Other IBGP peers of the RR that are not clients are called *nonclients*.

The *originator ID* is an optional, nontransitive BGP attribute that is created by the RR. This attribute carries the router ID of the route’s originator in the local autonomous system. If the update comes back to the originator because of poor configuration, the originator ignores it.

Usually a cluster has a single RR, in which case the cluster is identified by the RR’s router ID. To increase redundancy and avoid single points of failure, a cluster might have more than one RR. When this occurs, all the RRs in the cluster need to be configured with a *cluster ID*. The cluster ID allows route reflectors to recognize updates from other RRs in the same cluster.

A *cluster list* is a sequence of cluster IDs that the route has passed. When an RR reflects a route from its clients to nonclients outside the cluster, it appends the local cluster ID to the cluster list. If the update has an empty cluster list, the RR creates one. Using this attribute, an RR can tell whether the routing information is looped back to the same cluster because of poor configuration. If the local cluster ID is found in an advertisement’s cluster list, the advertisement is ignored.

The originator ID, cluster ID, and cluster list help prevent routing loops in RR configurations.

Route Reflector Design

When using RRs in an autonomous system, you can divide the autonomous system into multiple clusters, each having at least one RR and a few clients. Multiple route reflectors can exist in one cluster for redundancy.

The RRs must be fully meshed with IBGP to ensure that all routes learned are propagated throughout the autonomous system.

An IGP is still used, just as it was before RRs were introduced, to carry local routes and next-hop addresses.

Split-horizon rules still apply between an RR and its clients. Thus an RR that receives a route from a client does not advertise that route back to that client.

Note

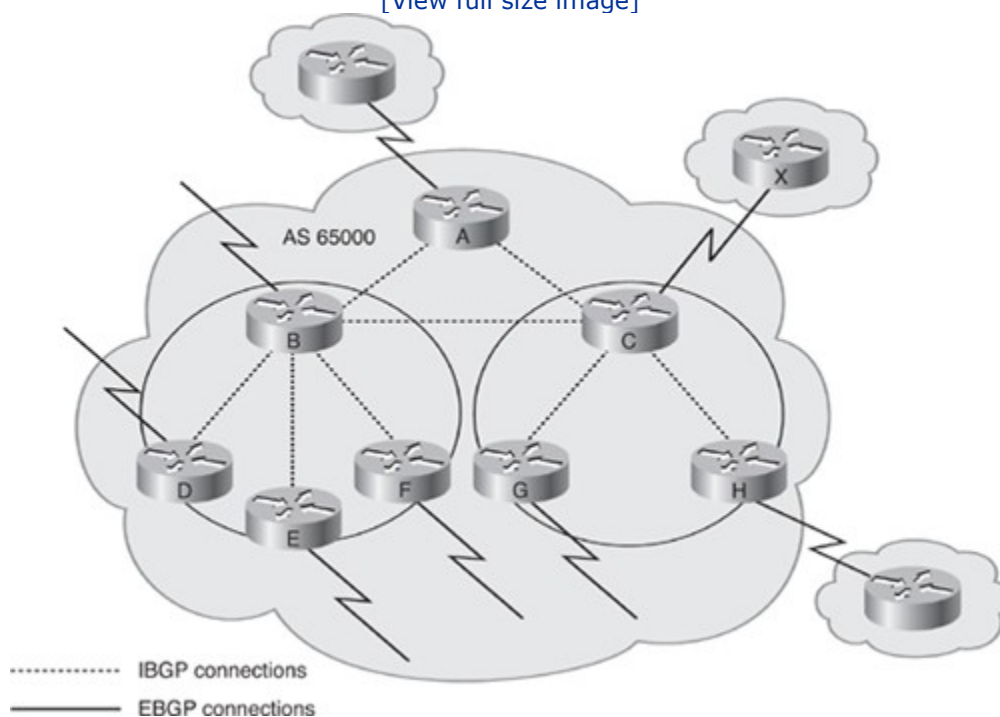
No defined limit applies to the number of clients an RR might have. It is constrained by the amount of router memory.

Route Reflector Design Example

Figure C-9 provides an example of a BGP RR design.

Figure C-9. Example of a Route Reflector Design.

[\[View full size image\]](#)



Note

The physical connections within autonomous system 65000 are not shown in [Figure C-12](#).

In [Figure C-9](#), Routers B, D, E, and F form one cluster. Routers C, G, and H form another cluster. Routers B and C are RRs. Routers A, B, and C are fully meshed with IBGP. Note that the routers within a cluster are not fully meshed.

Route Reflector Operation

When an RR receives an update, it takes the following actions, depending on the type of peer that sent the update:

- If the update is from a client peer, it sends the update to all nonclient peers and to all client peers (except the route's originator).
- If the update is from a nonclient peer, it sends the update to all clients in the cluster.
- If the update is from an EBGp peer, it sends the update to all nonclient peers and to all client peers.

For example, in [Figure C-9](#), the following happens:

- If Router C receives an update from Router H (a client), it sends it to Router G, and to Routers A and B.
- If Router C receives an update from Router A (a nonclient), it sends it to Routers G and H.
- If Router C receives an update from Router X (via EBGp), it sends it to Routers G and H, and to Routers A and B.

Note

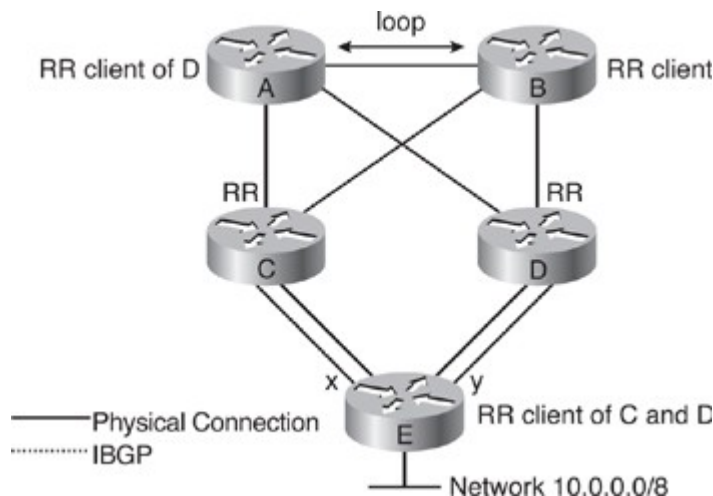
Routers also send updates to their EBGp neighbors as appropriate.

Route Reflector Migration Tips

When migrating to using RRs, the first consideration is which routers should be the reflectors and which should be the clients. Following the physical topology in this design decision ensures that the packet-forwarding paths are not affected. Not following the physical topology (for example, configuring RR clients that are not physically connected to the route reflector) might result in routing loops.

[Figure C-10](#) demonstrates what can happen if RRs are configured without following the physical topology. In this figure, the lower router, Router E, is an RR client for both RRs, Routers C and D.

Figure C-10. Bad Route Reflector Design That Does Not Follow the Physical Topology.



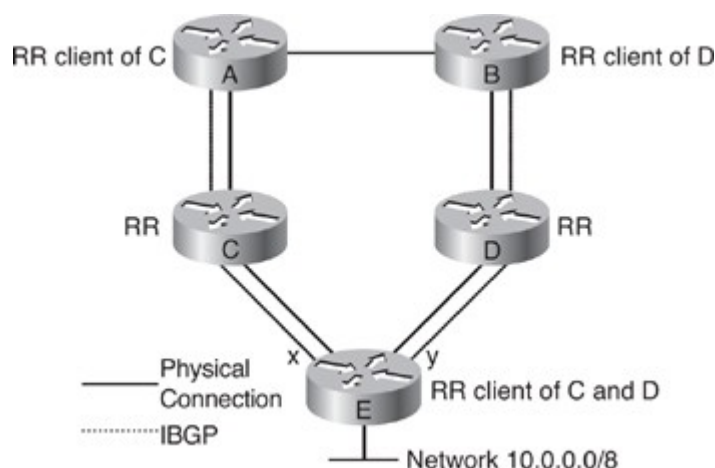
In this *bad design*, which does not follow the physical topology, the following happens:

- Router B knows that the next hop to get to 10.0.0.0 is x (because it learns this from its RR, Router C).

- Router A knows that the next hop to get to 10.0.0.0 is *y* (because it learns this from its RR, Router D).
- For Router B to get to *x*, the best route might be through Router A, so Router B sends a packet destined for 10.0.0.0 to Router A.
- For Router A to get to *y*, the best route might be through Router B, so Router A sends a packet destined for 10.0.0.0 to Router B.
- This is a routing loop.

Figure C-11 shows a better design (better because it follows the physical topology). Again, in this figure, the lower router, Router E, is an RR client for both route reflectors.

Figure C-11. Good Route Reflector Design That Does Follow the Physical Topology.



In this *good design*, which follows the physical topology, the following are true:

- Router B knows that the next hop to get to 10.0.0.0 is *y* (because it learns this from its RR, Router D).
- Router A knows that the next hop to get to 10.0.0.0 is *x* (because it learns this from its RR, Router C).
- For Router A to get to *x*, the best route is through Router C, so Router A sends a packet destined for 10.0.0.0 to Router C, and Router C sends it to Router E.
- For Router B to get to *y*, the best route is through Router D, so Router B sends a packet destined for 10.0.0.0 to Router D, and Router D sends it to Router E.
- There is no routing loop.

When migrating to using RRs, configure one RR at a time, and then delete the redundant IBGP sessions between the clients. It is recommended that you configure one RR per cluster.

Route Reflector Configuration

The **neighbor *ip-address* route-reflector-client** router configuration command enables you to configure the router as a BGP RR and to configure the specified neighbor as its client. The *ip-address* is the IP address of the BGP neighbor being identified as a client.

Configuring the Cluster ID

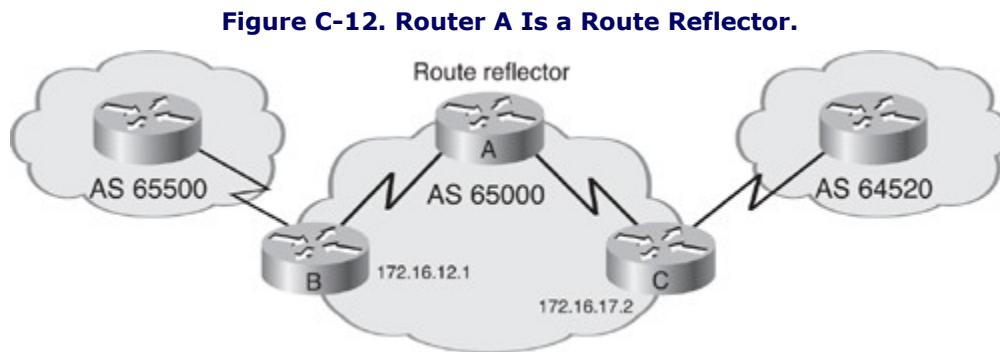
To configure the cluster ID if the BGP cluster has more than one RR, use the **bgp cluster-id cluster-id** router configuration command on all the RRs in a cluster. You cannot change the cluster ID after the RR clients have been configured.

RRs cause some restrictions on other commands, including the following:

- When used on RRs, the **neighbor next-hop-self** command affects only the next hop of EBGP learned routes, because the next hop of reflected IBGP routes should not be changed.
- RR clients are incompatible with peer groups. This is because a router configured with a peer group must send any update to *all* members of the peer group. If an RR has all of its clients in a peer group and then one of those clients sends an update, the RR is responsible for sharing that update with all *other* clients. The RR must not send the update to the originating client, because of the split-horizon rule.

Route Reflector Example

Figure C-12 illustrates a network, with Router A configured as an RR in autonomous system 65000. Example C-9 shows the configuration for Router A, the RR.



Example C-9. Configuration of Router A in Figure C-12

```
RTRA(config)#router bgp 65000
RTRA(config-router)#neighbor 172.16.12.1 remote-as 65000
RTRA(config-router)#neighbor 172.16.12.1 route-reflector-client
RTRA(config-router)#neighbor 172.16.17.2 remote-as 65000
RTRA(config-router)#neighbor 172.16.17.2 route-reflector-client
```

The **neighbor route-reflector-client** commands define which neighbors are RR clients. In this example, both Routers B and C are RR clients of Router A, the RR.

Verifying Route Reflectors

The **show ip bgp neighbors** command output indicates that a particular neighbor is an RR client. The sample partial output for this command, shown in Example C-10, is from Router A in Figure C-12 and shows that 172.16.12.1 (Router B) is an RR client of Router A.

Example C-10. show ip bgp neighbors Output from Router A in Figure C-12

```
RTRA#show ip bgp neighbors
```

```
BGP neighbor is 172.16.12.1, remote AS 65000, internal link
Index 1, Offset 0, Mask 0x2
Route-Reflector Client
BGP version 4, remote router ID 192.168.101.101
BGP state = Established, table version = 1, up for 00:05:42
Last read 00:00:42, hold time is 180, keepalive interval is 60 seconds
Minimum time between advertisement runs is 5 seconds
Received 14 messages, 0 notifications, 0 in queue
Sent 12 messages, 0 notifications, 0 in queue
Prefix advertised 0, suppressed 0, withdrawn 0
Connections established 2; dropped 1
Last reset 00:05:44, because of User reset
1 accepted prefixes consume 32 bytes
0 history paths consume 0 bytes
—More—
```

Acronyms and Abbreviations

This appendix identifies abbreviations, acronyms, and initialisms used in this book and in the internetworking industry.

Acronym	Expanded Term
3DES	Triple DES
6-to-4	IPv6-to-IPv4
AAA	Authentication, authorization, accounting
ABR	Area Border Router
ACK	1. acknowledge 2. acknowledgment 3. acknowledgment bit in a TCP segment
ACL	access control list
ACS	Access Control Server
AD	advertised distance
ADSL	asymmetric DSL
AES	Advanced Encryption Standard
AfriNIC	African Network Information Centre
AH	Authentication Header
ALG	application layer gateway
ANSI	American National Standards Institute
AON	Application-Oriented Networking
API	application programming interface
APNIC	Asia Pacific Network Information Center
ARCNET	Attached Resource Computer Network

Acronym	Expanded Term
ARIN	American Registry for Internet Numbers
ARP	Address Resolution Protocol
AS	autonomous system
ASA	Adaptive Security Appliance
ASBR	Autonomous System Boundary Router
ATM	Asynchronous Transfer Mode
AToM	Any Transport over MPLS
BDR	Backup Designated Router
BGP	Border Gateway Protocol
BGPv4 or BGP-4	BGP Version 4
BIS	Bump-in-the-Stack
BPDU	bridge protocol data unit
bps	bits per second
BRI	Basic Rate Interface
BSCI	Building Scalable Cisco Internetworks
CATV	community antenna television
CBAC	context-based access control
CCDP	Cisco Certified Design Professional
CCNA	Cisco Certified Network Associate
CCNP	Cisco Certified Network Professional
CCSP	Cisco Certified Security Professional
CDP	Cisco Discovery Protocol
CE	Customer Edge
CEF	Cisco Express Forwarding
CEFv6	Cisco Express Forwarding for IPv6
CIDR	classless interdomain routing
CIR	committed information rate
CMTS	Cable modem termination system
CO	Central office
CoS	class of service
CPE	customer provider edge customer premise equipment
CPU	central processing unit
CSM	Cisco Security Manager
CSMA/CD	carrier sense multiple access collision detect
DAD	Duplicate address detection

Acronym	Expanded Term
DBD	database description packets
DDP	database description packets
DES	Data Encryption Standard
DESGN	Designing for Cisco Internetwork Solutions
DHCP	Dynamic Host Configuration Protocol
DHCPv6	DHCP for IPv6
DiffServ	Differentiated Services
DLCI	data-link connection identifier
DMVPN	Dynamic multipoint VPN
DNA	DoNotAge
DNS	Domain Name Service or Domain Name System
DoD	Department of Defense
DR	designated router
DSL	digital subscriber line
DSLAM	DSL access multiplexer
DUAL	Diffusing Update Algorithm
E1	External Type 1
E2	External Type 2
EAP	Extensible Authentication Protocol
EBGP	External BGP
e-bit	external bit
EGP	Exterior Gateway Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
EoMPLS	Ethernet over MPLS
ESP	Encapsulating Security Payload
EUI-64	extended universal identifier 64-bit
FD	feasible distance
FCAPS	Fault, Configuration, Accounting, Performance, and Security
FDDI	Fiber Distributed Data Interface
FEC	forwarding equivalence class
FIB	Forwarding Information Base
FLSM	fixed-length subnet mask
FS	feasible successor
FTP	File Transfer Protocol
Gbps	gigabits per second

Acronym	Expanded Term
GET	Group encrypted transport
GRE	Generic Routing Encapsulation
HDLC	High-Level Data Link Control
HFC	hybrid fiber-coaxial
HSRP	Hot Standby Router Protocol
HTTP	Hypertext Transfer Protocol
Hz	hertz
IANA	Internet Assigned Numbers Authority
IBGP	Internal BGP
ICMP	Internet Control Message Protocol
ID	identifier
IDP	initial domain part
IDRP	Interdomain Routing Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IGP	Interior Gateway Protocol
IGRP	Interior Gateway Routing Protocol
IIN	Intelligent Information Network
IKE	Internet Key Exchange
IntServ	Integrated Services
IOS	Internet Operating System
IP	Internet Protocol
IPCP	Internet Protocol Control Protocol
IPM	Internetwork Performance Monitor
IPsec	IP security
IPv4	IP Version 4
IPv6	IP Version 6
IPX	Internetwork Packet Exchange
IS	1. information systems 2. intermediate system
ISAKMP	Internet Security Association and Key Management Protocol
ISATAP	Intra-Site Automatic Tunnel Addressing
IS-IS	Intermediate System-to-Intermediate System
IS-ISv6	IS-IS for IPv6

Acronym	Expanded Term
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISP	Internet service provider
ISR	integrated services router
ITIL	IT Infrastructure Library
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
kbps	kilobits per second
L3	Level 3
LACNIC	Latin American and Caribbean IP Address Regional Registry
LAN	local-area network
LANE	LAN Emulation
LS	link state
LSA	link-state advertisement
LSAck	link-state acknowledgment
LSDB	link-state database
LSP	link-state packet
LSR	link-state request
LSU	link-state update
M	metric
MAC	Media Access Control
MAN	metropolitan-area network
MB	megabyte
MBGP	Multiprotocol BGP
Mbps	megabits per second
MD5	message digest algorithm 5
MED	Multi-Exit-Discriminator
MIB	Management Information Base
MLD	Multicast Listener Discovery
MOSPF	Multicast OSPF
MP-BGP4	Multiprotocol Border Gateway Protocol Version 4
MPLS	Multiprotocol Label Switching
ms	Millisecond
MTU	maximum transmission unit
NA	Neighbor advertisement
NAC	Network Admission Control

Acronym	Expanded Term
NAT	Network Address Translation
NAT-T	NAT Traversal
NAT-PT	NAT-Protocol Translation
NBMA	nonbroadcast multiaccess
NCP	Network control protocol
ND	Neighbor discovery
NLRI	Network Layer Reachability Information
NMS	Network Management System
NS	Neighbor solicitation
NSSA	not-so-stubby area
NTP	Network Time Protocol
ODR	on-demand routing
OER	Optimized Edge Routing
OS	operating system
OSI	Open System Interconnection
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
OUI	organizationally unique identifier
P2P	Point-to-point
PAT	Port Address Translation
PBR	policy-based routing
PBX	Private Branch Exchange
PDM	protocol-dependent module
PDU	protocol data unit
PE	Provider Edge
POP	point of presence
POTS	plain old telephone service
PPDIOO	Prepare, Plan, Design, Implement, Operate, Optimize
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM
PPPoE	PPP over Ethernet
pps	packets per second
PPTP	Point-to-Point Tunneling Protocol
PRI	Primary Rate Interface

Acronym	Expanded Term
PSTN	public switched telephone network
PVC	permanent virtual circuit
QoS	quality of service
RA	Router advertisement
RF	Radio frequency
RFC	Request For Comments
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPE-NCC	Réseaux IP Européens-Network Coordination Center
RIPng	Routing Information Protocol new generation
RIPv1	Routing Information Protocol Version 1
RIPv2	Routing Information Protocol Version 2
RIR	Regional Internet registries
RPF	Reverse Path Forwarding
RR	route reflector
RRI	reverse route injection
RS	Router solicitation
RSVP	Resource Reservation Protocol
RTO	retransmit timeout
RTP	Reliable Transport Protocol
RTT	round-trip time
RTTMON	Round-Trip Time Monitor
SA	security association
SDSL	Symmetric DSL
SHA	Secure hash
SHDSL	single-pair high-speed DSL
SIA	stuck in active
SIN	ships in the night
SLAs	Service level agreements
SM	source metric
SMTP	Simple Mail Transfer Protocol
SNAP	Subnetwork Access Protocol
SNMP	Simple Network Management Protocol
SOHO	small office/home office
SONA	Service-Oriented Network Architecture

Acronym	Expanded Term
SP	Service Provider
SPF	shortest path first
SPT	shortest path tree
SPX	Sequenced Packet Exchange
SRTT	smooth round-trip time
ssh	secure shell
SSL	Secure socket layer
STP	1. shielded twisted-pair 2. Spanning Tree Protocol
SVC	switched virtual circuit
SYN	Synchronize
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TFTP	Trivial File Transfer Protocol
TLV	Type, Length, Value
TMN	Telecommunications Management Network
ToS	type of service
TTL	Time To Live
UDP	User Datagram Protocol
U/L	Universal/Local
URL	Uniform Resource Locator
UTP	unshielded twisted-pair
VC	virtual circuit
VDSL	very high bitrate DSL
VLAN	virtual LAN
VLSM	variable-length subnet mask
VoD	video on demand
VoIP	Voice over IP
VPN	virtual private network
VRF	VPN routing and forwarding
VT	video telephony
VTI	Virtual tunnel interface
VTP	VLAN Trunking Protocol
vty	virtual terminal
WAN	wide-area network

Acronym**Expanded Term**

WAAS	Wide Area Application Services
WCCP	Web Cache Communications Protocol
WFQ	weighted fair queuing
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	wireless LAN
WRED	weighted random early detection
WWW	World Wide Web
ZBF	zone-based firewall