Parts V and VI work together to reveal the details of how to implement IPv4 routing in Cisco routers. To that end, Part V focuses on the most common features for Cisco routers, including IP address configuration, connected routes, and static routes. Part VI then goes into some detail about the one IP routing protocol discussed in this book: OSPF Version 2 (OSPFv2).

Part V follows a progression of topics. First, Chapter 16 examines the fundamentals of routers—the physical components, how to access the router command-line interface (CLI), and the configuration process. Chapter 16 makes a close comparison of the switch CLI and its basic administrative commands so that you need to learn only new commands that apply to routers but not to switches.

Chapter 17 then moves on to discuss how to configure routers to route IPv4 packets in the most basic designs. Those designs require a simple IP address/mask configuration on each interface, with the addition of a static route command—a command that directly configures a route into the IP routing table—for each destination subnet.

Chapter 18 continues the progression into more challenging but more realistic configurations related to routing between subnets in a LAN environment. Most LANs use many VLANs, with one subnet per VLAN. Cisco routers and switches can be configured to route packets between those subnets, with more than a few twists in the configuration.

Chapter 19 moves the focus from routers to hosts. Hosts rely on a working internetwork of routers, but the hosts need IP settings as well, with an IP address, mask, default gateway, and DNS server list. This chapter examines how hosts can dynamically learn these IP settings using Dynamic Host Configuration Protocol (DHCP), and the role the routers play in that process. The chapter also shows how to display and understand IP settings on hosts.

Finally, Part V closes with a chapter about troubleshooting IPv4 routing. Chapter 20 features the **ping** and **traceroute** commands, two commands that can help you discover not only whether a routing problem exists but also where the problem exists. The other chapters show how to confirm whether a route has been added to one router's routing table, while the commands discussed in Chapter 20 teach you how to test the end-to-end routes from sending host to receiving host.

Part V

IPv4 Routing

Chapter 16: Operating Cisco Routers

Chapter 17: Configuring IPv4 Addresses and Static Routes

Chapter 18: IP Routing in the LAN

Chapter 19: IP Addressing on Hosts

Chapter 20: Troubleshooting IPv4 Routing

Part V Review

CHAPTER 16

Operating Cisco Routers

This chapter covers the following exam topics:

- **1.0 Network Fundamentals**
  - **1.1 Explain the role and function of network components**
    - **1.1.a Routers**
  - **1.2 Describe characteristics of network topology architectures**
    - **1.2.e Small office/home office (SOHO)**
  - **1.6 Configure and verify IPv4 addressing and subnetting**

This chapter begins a series of chapters that focus on specific Cisco router features. It begins by discussing Cisco routers: hardware, operating system, interfaces, and other components that comprise a router. This first section helps give you concrete examples of interfaces and devices before getting into the many concept and topology drawings to come.

The second section of the chapter then discusses the command-line interface (CLI) on a Cisco router, which has the same look and feel as the Cisco switch CLI. However, unlike switches, routers require some minimal configuration before they will do their primary job: to forward IP packets. The second section of this chapter discusses the concepts and commands to configure a router so it begins forwarding IP packets on its interfaces.

"Do I Know This Already?" Quiz

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

Table 16-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Installing Cisco Routers	1
Enabling IPv4 Support on Cisco Router Interfaces	2–6

1. Which operating systems run on Cisco enterprise routers and use a CLI that works much like the CLI on Cisco LAN switches? (Choose two answers.)
   - **a.** CatOS
   - **b.** IOS
   - **c.** Windows
   - **d.** IOS XE

2. Which action would you expect to be true of a router CLI interaction that is not true when configuring a LAN switch that performs only Layer 2 switching functions?
   a. Moving from global to physical interface configuration mode
   b. Configuring an IP address in physical interface configuration mode
   c. Configuring a 10/100/1000 port's settings related to speed and autonegotiation
   d. Configuring a console password
3. Which answers list a task that could be helpful in making a router interface G0/0 ready to route packets? (Choose two answers.)
   a. Configuring the **ip address** *address mask* command in G0/0 configuration mode
   b. Configuring the **ip address** *address* and **ip mask** *mask* commands in G0/0 configuration mode
   c. Configuring the **no shutdown** command in G0/0 configuration mode
   d. Setting the interface **description** in G0/0 configuration mode
4. The output of the **show ip interface brief** command on R1 lists interface status codes of "down" and "down" for interface GigabitEthernet 0/0. The interface connects to a LAN switch with a UTP straight-through cable. Which of the following could be true?
   a. The **shutdown** command is currently configured for router interface G0/0.
   b. The **shutdown** command is currently configured for the switch interface on the other end of the cable.
   c. The router was never configured with an **ip address** command on the interface.
   d. The router was configured with the **no ip address** command.
5. Which of the following commands list the IP address but not the subnet mask of an interface?
   a. **show running-config**
   b. **show protocols** *type number*
   c. **show ip interface brief**
   d. **show interfaces**
6. Which of the following is different on the Cisco switch CLI for a Layer 2 switch as compared with the Cisco router CLI?
   a. The commands used to configure simple password checking for the console
   b. The number of IP addresses configured
   c. The configuration of the device's hostname
   d. The configuration of an interface description

Foundation Topics

Installing Cisco Routers

Routers collectively provide the main feature of the network layer—the capability to forward packets end to end through a network. As introduced in Chapter 3, "Fundamentals of WANs and IP Routing," routers forward packets by connecting to various physical network links, like Ethernet LAN, Ethernet WAN, and serial WAN links, then using Layer 3 routing

logic to choose where to forward each packet. As a reminder, Chapter 2, "Fundamentals of Ethernet LANs," covered the details of making those physical connections to Ethernet networks, while Chapter 3 covered the basics of cabling with WAN links.

This section examines some of the details of router installation and cabling, first from the enterprise perspective and then from the perspective of connecting a typical small office/home office (SOHO) to an ISP using high-speed Internet.

Installing Enterprise Routers

A typical enterprise network has a few centralized sites as well as lots of smaller remote sites. To support devices at each site (the computers, IP phones, printers, and other devices), the network includes at least one LAN switch at each site. In addition, each site has a router, which connects to the LAN switch and to some WAN link. The WAN link provides connectivity from each remote site, back to the central site, and to other sites through the connection to the central site.

Figures 16-1 and 16-2 show a couple of different kinds of network diagrams that might be used to represent an enterprise network. The style of Figure 16-1 supports discussions about Layer 3 topics, showing the subnet IDs, masks, and interface IP addresses in shorthand. The figure also keeps the physical and data-link details to a minimum with these conventions:

Ethernet LAN: Simple straight lines with one or more LAN switches implied but not shown.

Ethernet WAN: Shown as a straight line, often with a cloud over it, with some kind of Ethernet interface identifier shown by the router (in this case, G0/1/0 and G0/0/0, which refers to GigabitEthernet interfaces).

Serial WAN: A line with a crooked part in the middle (a "lightning bolt") represents a typical point-to-point serial link as introduced in Chapter 3.

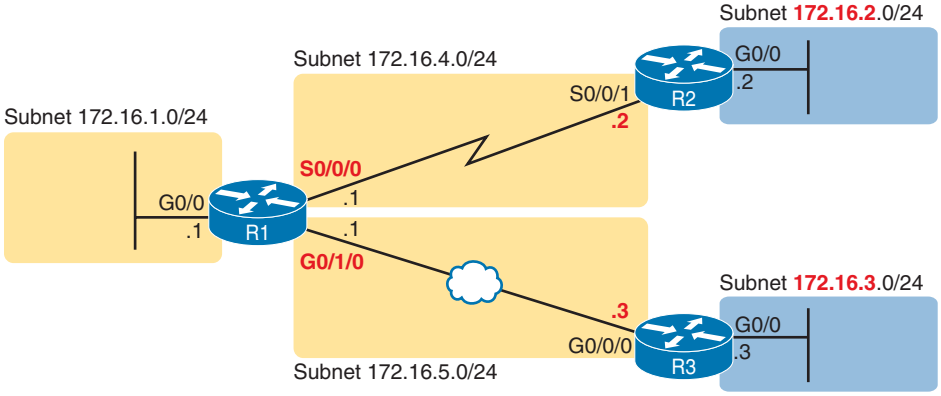

Figure 16-1 *Generic Enterprise Network Diagram*

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

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

In comparison, Figure 16-2 shows the same network, with more detail about the physical cabling but with IP details removed. Focusing on the LANs, all the lines connected to the LAN switches could be the standard UTP cabling with RJ-45 connectors.

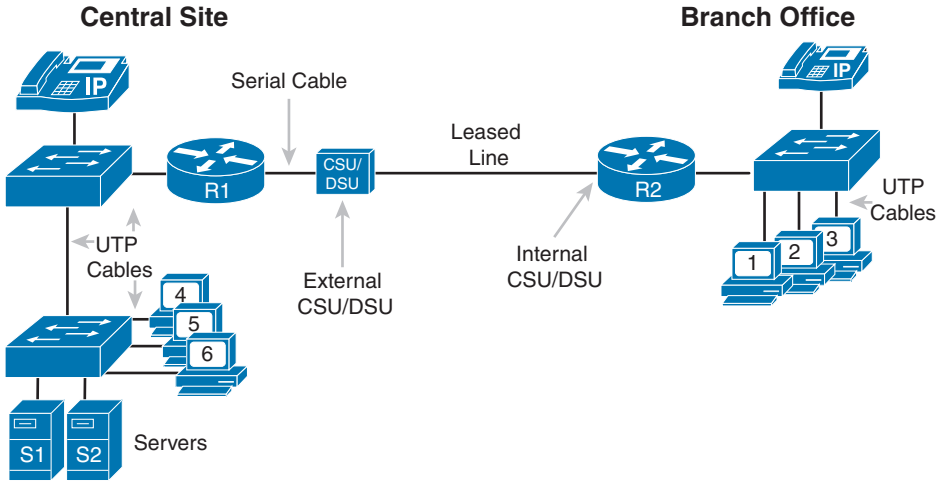

Figure 16-2 *More Detailed Cabling Diagram for the Same Enterprise Network*

Next, consider the hardware on the ends of the serial link between the two routers. In a real serial link that runs through a service provider, the link terminates at a channel service unit/data service unit (CSU/DSU). The CSU/DSU can either sit outside the router as a separate device (as shown on the left at Router R1) or integrated into the router's serial interface hardware (as shown on the right).

As for cabling, the service provider will run the cable into the enterprise's wiring closet and often put an RJ-48 connector (same size as an RJ-45 connector) on the end of the cable. That cable should connect to the CSU/DSU. With an internal CSU/DSU (as with Router R2 in Figure 16-2), the router serial port has an RJ-48 port to which the serial cable should connect. With an external CSU/DSU, the CSU/DSU must be connected to the router's serial card via a short serial cable.

The Cisco Router Operating Systems

All routers have the usual components found in a computer: a CPU, RAM, permanent memory (usually flash memory), and other electronics. They also run an operating system (OS), which goes by the name **IOS**. The original Cisco routers used IOS; even today, some current router products use IOS. However, Cisco has created other enterprise-class router product families that use a different variation of IOS named **IOS XE**.

Cisco created IOS XE in the 2000s to improve the IOS software architecture. Those improvements may not be evident to the casual observer, but to name a few, IOS XE reduces unplanned and planned downtime, better protects against cyberattacks, and aids network automation. For instance, IOS XE devices can support upgrading the OS while continuing to forward frames and packets, while IOS cannot.

Thankfully, IOS XE uses the same familiar CLI as IOS. Both use the same commands, for the most part, the same command syntax, navigation and modes, and so on. If you learned the CLI using an IOS router, you might not even notice when using a router that runs IOS XE later.

Because the differences between IOS and IOS XE do not matter in most cases in this book, the book uses the term *IOS* almost exclusively to refer to the router OS. When differences in IOS versus IOS XE matter, the text will note the differences.

Cisco Integrated Services Routers

Product vendors, including Cisco, typically provide several different types of router hardware. Today, routers often do much more work than simply routing packets; in fact, they serve as a device or platform from which to provide many network services.

As an example, consider the networking functions needed at a typical branch office. A typical enterprise branch office needs a router for WAN/LAN connectivity, and a LAN switch to provide a high-performance local network and connectivity into the router and WAN. Many branches also need voice-over-IP (VoIP) services to support IP phones, and several security services as well. Plus, it is hard to imagine a site with users that does not have Wi-Fi access today. So, rather than require multiple separate devices at one site, as shown in Figure 16-2, Cisco offers single devices that act as both router and switch and provide other functions as well.

For the sake of learning and understanding the different functions, this book focuses on using a separate switch and separate router, which provides a much cleaner path for learning the basics. However, most Cisco products and product families support a variety of functions in one device.

The Cisco **Integrated Services Router (ISR)** product families include routers that perform much more than routing. Cisco introduced the first ISRs in the mid-2000s, with some ISR families still in existence in the early 2020s when this chapter was most recently updated. The name itself emphasizes the role of being a router, while also integrating other functions (services).

Figure 16-3 shows a Cisco 4321 ISR, with some of the more important features highlighted. The top part of the figure shows a full view of the back of the router. This model comes with two built-in Gigabit Ethernet interfaces and two modular slots that allow you to add small cards called Network Interface Modules (NIMs). The bottom of the figure shows one sample NIM (a NIM that provides two serial interfaces). The router has other items as well, including both an RJ-45 and USB console port.

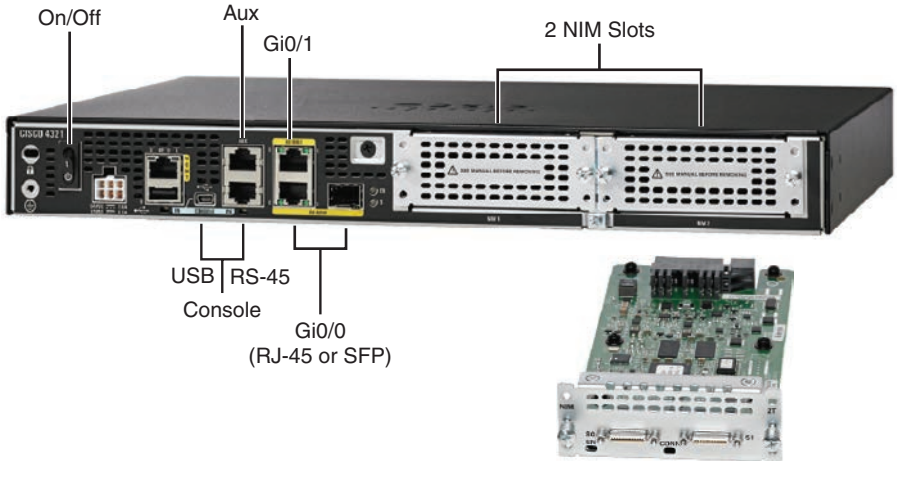

Figure 16-3 *Photos of a Model 4321 Cisco Integrated Services Router (ISR)*

The figure shows an important feature for using routers to connect to both Ethernet LANs and Ethernet WAN services. Look closely at Figure 16-3's Gigabit interfaces. G0/1 refers to interface GigabitEthernet0/1 and is an RJ-45 port that supports UTP cabling only. However, interface G0/0 (short for GigabitEthernet0/0) has some interesting features:

- The router has two ports for one interface (G0/0).
- You can use one or the other at any point in time, but not both.
- One physical port is an RJ-45 port that supports copper cabling (implying that it is used to connect to a LAN).
- The other G0/0 physical port is a small form pluggable (SFP) port that would support various fiber Ethernet standards, allowing the port to be used for Ethernet WAN purposes.

Cisco commonly makes one or more of the Ethernet ports on its Enterprise class routers support SFPs so that the engineer can choose an SFP that supports the type of Ethernet cabling provided by the Ethernet WAN service provider.

NOTE When building a lab network to study for CCNA or CCNP, because your devices will be in the same place, you can create Ethernet WAN links by using the RJ-45 ports and a UTP cable without the need to purchase an SFP for each router.

The Cisco Catalyst Edge Platform

In 2020, Cisco announced a new router family with the **Cisco Catalyst Edge Platform** branding, a departure from the previous ISR branding. The new family includes models in the 8000 series, such as the 8200, 8300, and 8500. These new models replaced some existing Cisco ISR families in the Cisco product lineup, but not all.

If the ISR brand emphasized the multifunction nature of modern routers, the *Catalyst Edge Platform* emphasizes that point even more. First, the name does not even use the word *router*, even though the devices use IOS XE and act primarily as routers. The word *platform* emphasizes that the device serves as a platform to run various network services.

Also, note that the new branding uses the term *Catalyst*, a term historically used only for switches. Before the Catalyst 8000 series, the term *Catalyst* always referred to switches, but now Cisco uses the term more broadly. Just be aware in your travels outside of your CCNA studies that when you come across the Catalyst 8000 series devices, think of them as routers.

As for exam preparation, this chapter provides information about ISRs and Catalyst 8000s to provide some context. However, the exam topics do not mention specific Cisco product families. But knowing something about router models and families can help you connect the more generalized topics in CCNA to the devices in your network at work.

Physical Installation

Armed with the cabling details in images like Figure 16-2 and the router hardware details in photos like Figure 16-3, you can physically install a router. To install a router, follow these steps:

Key Topic

Step 1. For any Ethernet LAN interface, connect the RJ-45 connector of an appropriate copper Ethernet cable between the RJ-45 Ethernet port on the router and one of the LAN switch ports.

Step 2. For any serial WAN ports:

- **Step A.** If using an external CSU/DSU, connect the router's serial interface to the CSU/DSU and the CSU/DSU to the line from the telco.
- **Step B.** If using an internal CSU/DSU, connect the router's serial interface to the line from the telco.

Step 3. For any Ethernet WAN ports:

- **Step A.** When ordering the Ethernet WAN service, confirm the required Ethernet standard and SFP type required to connect to the link, and order the SFPs.
- **Step B.** Install the SFPs into the routers, and connect the Ethernet cable for the Ethernet WAN link to the SFP on each end of the link.

Step 4. Connect the router's console port to a PC (as discussed in Chapter 4, "Using the Command-Line Interface"), as needed, to configure the router.

Step 5. Connect a power cable from a power outlet to the power port on the router.

Step 6. Power on the router using the on/off switch.

Note that Cisco enterprise routers typically have an on/off switch, while switches do not.

Installing SOHO Routers

The terms **enterprise router** and *small office/home office (SOHO) router* act as a pair of contrasting categories for routers, both in terms of how vendors like Cisco provide to the market, and how enterprises use and configure those devices. The term *enterprise router* typically refers to a router that a company would use in a permanent business location, while a **SOHO router** would reside at an employee's home or at a small permanent site with just a few people. However, as you might guess, the line between a router acting as an enterprise router and a SOHO router is blurry, so use these terms as general categories.

Even with that general comparison, SOHO routers typically have two features that an enterprise router would be less likely to have:

- SOHO routers almost always use the Internet and virtual private network (VPN) technology for its WAN connection to send data back and forth to the rest of the enterprise.
- SOHO routers almost always use a multifunction device that does routing, LAN switching, VPN, wireless, and maybe other features.

For instance, at an enterprise business location, the building may contain enterprise routers, separate Ethernet switches, and separate wireless access points (APs), all connected together.

At a permanent business site with four employees and 10 total devices in the network, one SOHO router could provide all those same features in one device.

For instance, Figure 16-4 shows a typical SOHO site. The three icons that represent a router, switch, and access point all exist inside one box; the figure shows them separately to emphasize the fact that the one SOHO router provides several functions. On the left, the SOHO router provides wired and wireless LAN servers, and on the right, it provides WAN access through a cable Internet connection.

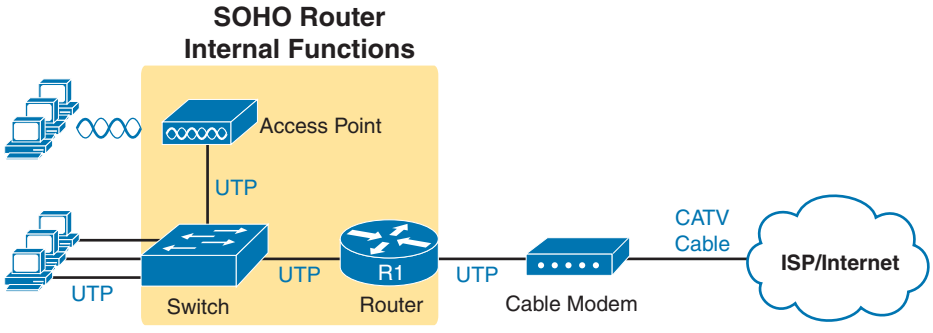

Figure 16-4 *Devices in a SOHO Network with High-Speed CATV Internet*

Figure 16-4 does not reflect the physical reality of a SOHO router, so Figure 16-5 shows one cabling example. The figure shows user devices on the left, connecting to the router via wireless or via Ethernet UTP cabling. On the right in this case, the router uses an external cable modem to connect to the coaxial cable provided by the ISP. Then the router must use a normal UTP Ethernet port to connect a short Ethernet cable between the SOHO router and the cable modem.

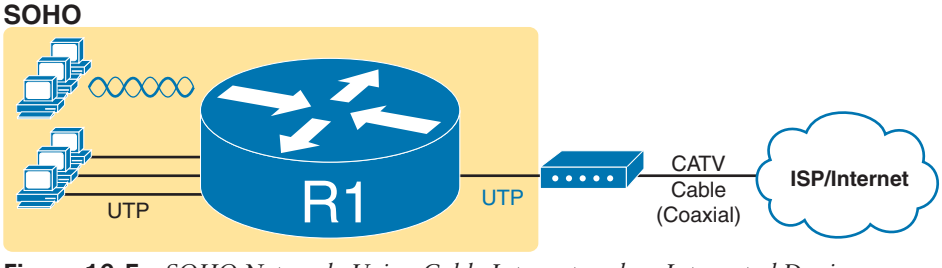

Figure 16-5 *SOHO Network, Using Cable Internet and an Integrated Device*

Enabling IPv4 Support on Cisco Router Interfaces

Routers support a relatively large number of features, with a large number of configuration and EXEC commands to support those features. You will learn about many of these features throughout the rest of this book.

NOTE For perspective, the Cisco router documentation includes a command reference, with an index to every single router command. A quick informal count listed over 5000 CLI commands.

This second section of the chapter focuses on commands related to router interfaces. To make routers work—that is, to route IPv4 packets—the interfaces must be configured. This section introduces the most common commands that configure interfaces, make them work, and give the interfaces IP addresses and masks.

Accessing the Router CLI

The router command-line interface (CLI) works much like a switch. In fact, rather than repeat the detail in Chapter 4, just assume the CLI details from that chapter also apply to routers. If the details from Chapter 4 are not fresh in your memory, it might be worthwhile to spend a few minutes briefly reviewing that chapter as well as Chapter 7, "Configuring and Verifying Switch Interfaces," before reading further. The following list reviews some of the basic CLI features in common between switches and routers:

Key Topic

- User and Privileged (enable) mode
- Entering and exiting configuration mode, using the **configure terminal**, **end**, and **exit** commands and the Ctrl+Z key sequence
- Configuration of console, Telnet (vty), and enable secret passwords
- Configuration of Secure Shell (SSH) encryption keys and username/password login credentials
- Configuration of the hostname and interface description
- Configuration of an interface to be administratively disabled (**shutdown**) and administratively enabled (**no shutdown**)
- Navigation through different configuration mode contexts using commands like **line console 0** and **interface** *type number*
- CLI help, command editing, and command recall features
- The meaning and use of the startup-config (in NVRAM), running-config (in RAM), and external servers (like TFTP), along with how to use the **copy** command to copy the configuration files and IOS images

The most significant differences in the CLI between Cisco routers and switches come from the primary functions of each device: Layer 2 LAN switching by switches and Layer 3 IP routing by routers. LAN switches consider physical interfaces *Layer 2 interfaces*, or *switched interfaces*, meaning the switch should receive Ethernet frames on the interface and pass those into the LAN switching logic. Routers consider their physical interfaces to be *Layer 3 interfaces* or *routed interfaces*. Layer 3 interfaces expect to receive frames, then de-encapsulate the packet found inside the frame, and pass the packet off to the internal routing processes of the router.

The best way to become comfortable with the differences between switches and routers is to learn more and more commands on each. The chapters from here to the end of this book, and several in *CCNA 200-301 Official Cert Guide, Volume 2,* Second Edition, reveal more detail about router features and the related commands. Here are a few of the router CLI topics you will learn about for routers, with one sample command each:

Key Topic

- Configuring interface IP addresses (**ip address** *address mask*)
- Configuring IP routing protocols (e.g., OSPF) (**router ospf** *process-id*)

- Verifying the IP routing table (**show ip route**)
- Configuring static IP routes (**ip route** *subnet mask next-hop-address*)

NOTE Modern routers and switches often support both routing and switching features. Network engineers can then decide whether they need a switch in a particular location in a network design (usually when stronger switching features are required) or a router (when stronger routing features are needed). When studying for CCNA, be aware that routers and switches often support commands for both routing and switching, commands that you might otherwise expect to see only in one or the other.

Cisco routers require the same commands to secure Telnet and SSH, so take some time to review the "Securing the Switch CLI" section in Chapter 6, "Configuring Basic Switch Management." Additionally, take extra care with one command related to Telnet and SSH configuration: the **transport input** line subcommand. The command has different defaults on switches and routers, and the default has migrated to a new setting as well. For instance, older enterprise-class routers using IOS (not IOS XE) often default to use **transport input none**, meaning the router supported neither Telnet nor SSH. Enterprise-class routers running IOS XE often default to **transport input ssh**, supporting SSH but not Telnet.

Router Interfaces

One minor difference between Cisco switches and routers is that routers support a much wider variety of interfaces. Today, LAN switches support only Ethernet interfaces of various speeds. Routers support a variety of other types of interfaces, including serial interfaces, cable TV, 4G/5G wireless, and others not mentioned in this book.

Most Cisco routers have at least one Ethernet interface of some type. Many of those Ethernet interfaces support multiple speeds and use autonegotiation, so for consistency, the router IOS refers to these interfaces based on the fastest speed. For example, a 10-Mbps-only Ethernet interface would be configured with the **interface ethernet** *number* configuration command, a 10/100 interface with the **interface fastethernet** *number* command, and a 10/100/1000 interface with the **interface gigabitethernet** *number* command. However, when discussing these interfaces all together, engineers would simply call them *ethernet interfaces*, regardless of the maximum speed.

Some Cisco routers have serial interfaces. As you might recall from Chapter 3, Cisco routers use serial interfaces to connect to a serial link. Each point-to-point serial link can then use High-Level Data Link Control (HDLC, the default) or Point-to-Point Protocol (PPP).

Router commands refer to interfaces first by the type of interface (Ethernet, Fast Ethernet, Gigabit Ethernet, Serial, and so on) and then with a unique interface identifier (a number) on that router. Depending on the router model, the interface identifiers might be a single number, two numbers separated by a slash, or three numbers separated by slashes. For example, all of the following configuration commands are correct on at least one model of Cisco router:

```
interface ethernet 0
interface fastethernet 0/1
interface gigabitethernet 0/0
interface gigabitethernet 0/1/0
interface serial 1/0/1
interface cellular 0/0/1
```

Two of the most common commands to display the interfaces, and their status, are the **show ip interface brief** and **show interfaces** commands. The first of these commands displays a list with one line per interface, with some basic information, including the interface IP address and interface status. The second command lists the interfaces, but with a large amount of information per interface. Example 16-1 shows a sample of each command. The output comes from an ISR router that has both a GigabitEthernet0/0 interface and a GigabitEthernet0/1/0 interface, showing a case with both two-digit and three-digit interface identifiers.

Example 16-1 *Listing the Interfaces in a Router*

```
R1# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
Embedded-Service-Engine0/0 unassigned      YES NVRAM  administratively down down
GigabitEthernet0/0         172.16.1.1      YES NVRAM  down                  down
GigabitEthernet0/1         unassigned      YES NVRAM  administratively down down
Serial0/0/0                172.16.4.1      YES manual up                    up
Serial0/0/1                unassigned      YES unset  administratively down down
GigabitEthernet0/1/0       172.16.5.1      YES NVRAM  up                    up

R1# show interfaces gigabitEthernet 0/1/0
GigabitEthernet0/1/0 is up, line protocol is up
  Hardware is EHWIC-1GE-SFP-CU, address is 0201.a010.0001 (bia 30f7.0d29.8570)
  Description: Link in lab to R3's G0/0/0
  Internet address is 172.16.5.1/24
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full Duplex, 1Gbps, media type is RJ45
  output flow-control is XON, input flow-control is XON
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:29, output 00:00:08, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     12 packets input, 4251 bytes, 0 no buffer
     Received 12 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     55 packets output, 8098 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
```

```
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

NOTE Commands that refer to router interfaces can be significantly shortened by truncating the words. For example, **sh int gi0/0** or **sh int g0/0** can be used instead of **show interfaces gigabitethernet0/0**. In fact, many network engineers, when looking over someone's shoulder, would say something like "just do a show int G-I-oh-oh command" in this case, rather than speaking the long version of the command.

Also, note that the **show interfaces** command lists a text interface description on about the third line, if configured. In this case, interface G0/1/0 had been previously configured with the **description Link in lab to R3's G0/0/0** command in interface configuration mode for interface G0/1/0. The **description** interface subcommand provides an easy way to keep small notes about what router interfaces connect to which neighboring devices, with the **show interfaces** command listing that information.

Interface Status Codes

Each interface has two *interface status codes*. To be usable, the two interface status codes must be in an "up" state. The first status code refers to whether Layer 1 is working, and the second status code generally (but not always) refers to whether the data-link layer protocol is working. Table 16-2 summarizes these two status codes.

Key Topic

Table 16-2 Interface Status Codes and Their Meanings

Name	Location	General Meaning
Line status	First status code	Refers to the Layer 1 status. (For example, is the cable installed, is it the right/wrong cable, is the device on the other end powered on?)
Protocol status	Second status code	Refers generally to the Layer 2 status. It is always down if the line status is down. If the line status is up, a protocol status of down is usually caused by a mismatched data-link layer configuration.

Several combinations of interface status codes exist, as summarized in Table 16-3. The table lists the status codes in order, from being disabled on purpose by the configuration to a fully working state.

Key Topic

Table 16-3 Typical Combinations of Interface Status Codes

Line Status	Protocol Status	Typical Reasons
Administratively down	Down	The interface has a **shutdown** command configured on it.
Down	Down	The interface is not **shutdown**, but the physical layer has a problem. For example, no cable has been attached to the interface, or with Ethernet, the switch interface on the other end of the cable is shut down, or the switch is powered off, or the devices on the ends of the cable use a different transmission speed.

Line Status	Protocol Status	Typical Reasons
Up	Down	Almost always refers to data-link layer problems, most often configuration problems. For example, serial links have this combination when one router was configured to use PPP and the other defaults to use HDLC.
Up	Up	Both Layer 1 and Layer 2 of this interface are functioning.

For some examples, look back at Example 16-1's **show ip interface brief** command, to the three interfaces in the following list. The interfaces in this list each have a different combination of interface status codes; the list details the specific reasons for this status code in the lab used to create this example for the book.

G0/0: The interface is down/down, in this case because no cable was connected to the interface.

G0/1: The interface is administratively down/down, because the configuration includes the **shutdown** command under the G0/1 interface.

S0/0/0: The interface is up/up because a serial cable is installed, is connected to another router in a lab, and is working.

Router Interface IP Addresses

Cisco enterprise routers require at least some configuration beyond the default configuration before they will do their primary job: routing IP packets. First, default router behavior may place an interface in a disabled (shutdown) state. Second, routers use a default interface command of **no ip address**, meaning the interface has no IP address or mask configured. For a router interface to function for routing packets, the interface must be up and configured with an IP address and mask.

To configure the address and mask, simply use the **ip address** *address mask* interface subcommand and ensure that the interface is enabled by using the **no shutdown** command. Figure 16-6 shows a simple IPv4 network with IPv4 addresses on Router R1, with Example 16-2 showing the matching configuration.

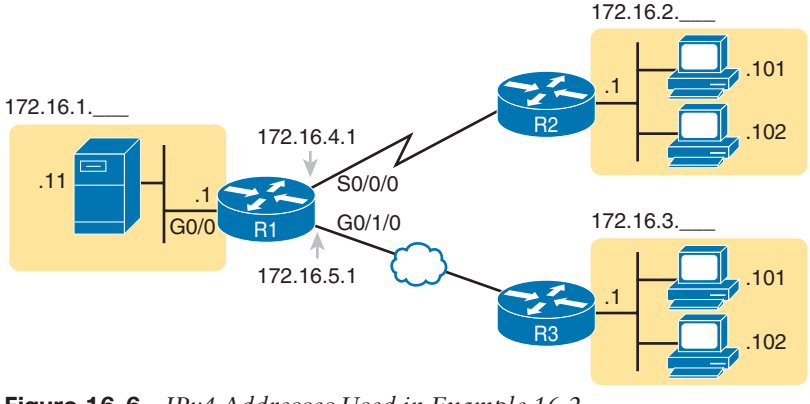

Figure 16-6 *IPv4 Addresses Used in Example 16-2*

Example 16-2 *Configuring IP Addresses on Cisco Routers*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1config)# interface G0/0
R1(config-if)# ip address 172.16.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface S0/0/0
R1(config-if)# ip address 172.16.4.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface G0/1/0
R1(config-if)# ip address 172.16.5.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# ^Z
R1#
```

The **show protocols** command in Example 16-3 confirms the state of each of the three R1 interfaces in Figure 16-6 and the IP address and mask configured on those same interfaces.

Example 16-3 *Verifying IP Addresses on Cisco Routers*

```
R1# show protocols
Global values:
   Internet Protocol routing is enabled
Embedded-Service-Engine0/0 is administratively down, line protocol is down
GigabitEthernet0/0 is up, line protocol is up
   Internet address is 172.16.1.1/24
GigabitEthernet0/1 is administratively down, line protocol is down
Serial0/0/0 is up, line protocol is up
   Internet address is 172.16.4.1/24
Serial0/0/1 is administratively down, line protocol is down
GigabitEthernet0/1/0 is up, line protocol is up
   Internet address is 172.16.5.1/24
```

One of the first actions to take when verifying whether a router is working is to find the interfaces, check the interface status, and check to see whether the correct IP addresses and masks are used. Examples 16-1 and 16-3 showed samples of the key **show** commands, while Table 16-4 summarizes those commands and the types of information they display.

Key Topic

Table 16-4 Key Commands to List Router Interface Status

Command	Lines of Output per Interface	IP Configuration Listed	Interface Status Listed?
show ip interface brief	1	Address	Yes
show protocols [*type number*]	1 or 2	Address/mask	Yes
show interfaces [*type number*]	Many	Address/mask	Yes

Ethernet Interface Autonegotiation

The first step to make a router Ethernet interface reach an up/up state requires the installation of the correct cable. For instance, for Ethernet WAN links, the physical standard may need to support distances of several kilometers, so the interface hardware typically requires fiber-optic cabling. Router LAN interfaces often use UTP cabling that connects to nearby LAN switches, with the expected straight-through cable pinout.

For Ethernet interfaces that allow multiple standards and speeds, you also need to ensure the interfaces use autonegotiation or use correct static settings. IOS routers use the same concepts and commands as Cisco switches, as discussed in the section, "Configuring Autonegotiation, Speed, and Duplex," in Chapter 7. IOS XE routers use the same autonegotiation processes but with different commands, so read the next few pages to work through those differences.

First, for the familiar, the top part of Figure 16-7 shows an example with an IOS Router R1 connected to a Cisco switch. The figure shows all the default settings on both the router and the switch, which cause both devices to perform IEEE autonegotiation.

Key Topic

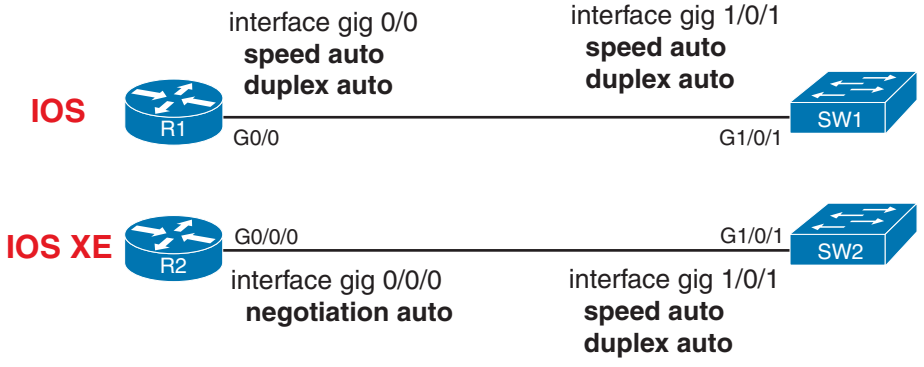

Figure 16-7 *Default Ethernet Autonegotiation Configuration: IOS and IOS XE*

The bottom half of the figure focuses on the different configuration in IOS XE Router R2. The IOS XE default interface subcommand, **negotiation auto**, does just what it appears to do—it enables IEEE autonegotiation. As a result, both devices use autonegotiation to choose the link speed and duplex.

The **show interfaces** EXEC command reveals the chosen duplex and speed. For an IOS (not IOS XE) example, refer to Example 16-1, about nine lines into the **show interfaces** command output, to see the speed (1Gbps) and duplex (full) listed. However, none of the IOS **show** commands specifically note that the router used autonegotiation.

For the IOS XE case, several commands reveal the use of IEEE autonegotiation, as seen in Example 16-4's output from IOS XE Router R2 from Figure 16-7. The **show interfaces** command (in the highlighted line) confirms the speed and duplex. The highlighted line also lists "link type is auto," meaning it autonegotiated the values.

Example 16-4 *Verifying IOS XE Autonegotiation Results (Default)*

```
! Output below is from IOS XE router R2
R2# show interfaces g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Hardware is C1111-2x1GE, address is 0200.1111.1111 (bia 2436.dadf.5680)
  Internet address is 10.1.1.1/24
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
  Full Duplex, 1000Mbps, link type is auto, media type is RJ45
! Lines omitted for brevity
R2# show interfaces g0/0/0 controller | include Autoneg
Admin State Up MTU 1500 Speed 1000mbps Duplex full Autoneg On Stats Interval 5
```

The IOS XE **show interfaces g0/0/0 controller** command at the end of the example filters the output to show only the subset of lines that include the text "Autoneg." That line also confirms the use of autonegotiation.

Whether you are using IOS or IOS XE, the recommendations for using autonegotiation on router Ethernet interfaces mirror the recommendations for switch interfaces, summarized here as follows:

- Use autonegotiation on both ends of an Ethernet link.
- If you must set the speed or duplex, ensure you configure the devices on both ends of the link to use the same speed and duplex.
- Avoid configuring speed and duplex on one end while relying on autonegotiation on the other.

If you must manually configure the settings, IOS and IOS XE differ slightly in how to configure speed and duplex manually. IOS routers use the same conventions as switches: just configure both the **speed** and **duplex** interface subcommands to specific settings, as shown in the left of Figure 16-8. With IOS routers, configuring both speed and duplex also disables autonegotiation.

Key Topic

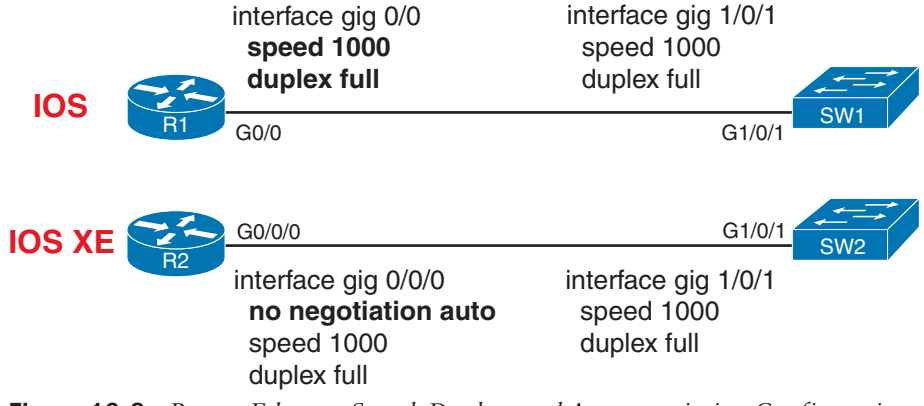

Figure 16-8 *Router Ethernet Speed, Duplex, and Autonegotiation Configuration*

IOS XE requires the commands shown by Router R2 in the figure, with Example 16-5 demonstrating those same configuration steps. The example begins by showing how IOS XE does not allow the configuration of the **speed** or **duplex** commands until you disable autonegotiation with the **no negotiation auto** interface subcommand. After disabling autonegotiation, IOS XE allows setting the specific **speed** and **duplex** values. The example then shows log messages informing you that the interface failed and then recovered, which is common when using new speed and duplex settings, with approximately ten seconds to work through the process.

Example 16-5 *IOS XE Router: Setting Speed and Duplex*

```
R2# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)# interface g0/0/0
R2(config-if)# speed 1000
Auto-negotiation is enabled. Speed cannot be set

R2(config-if)# no negotiation auto
R2(config-if) # speed 1000
R2(config-if) # duplex full
*Oct 14 12:24:16.014: %LINEPROTO-5-UPDOWN: Line protocol on Interface
GigabitEthernet0/0/0, changed state to down
*Oct 14 12:24:24.207: %LINEPROTO-5-UPDOWN: Line protocol on Interface
GigabitEthernet0/0/0, changed state to up
R2(config-if)# ^Z
*Oct 14 12:19:10.210: %LINK-3-UPDOWN: Interface GigabitEthernet0/0/0, changed state
to up
R2#
R2# show interfaces g0/0/0 controller
GigabitEthernet0/0/0 is up, line protocol is up
  Hardware is C1111-2x1GE, address is 0200.1111.1111 (bia 2436.dadf.5680)
  Internet address is 10.1.1.1/24
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
  Full Duplex, 1000Mbps, link type is force-up, media type is RJ45
  output flow-control is on, input flow-control is on
! Lines omitted for brevity
Driver Configuration Block:
Admin State Up MTU 1500 Speed 1000mbps Duplex full Autoneg Off Stats Interval 5
! More lines omitted for brevity
```

The highlighted text near the end of Example 16-5 confirms the resulting speed and duplex. It also identifies that autonegotiation was not used, with the phrase "link type is force-up," in contrast to the "link type is auto" text in the autonegotiation in Example 16-4. Later in the output, one line explicitly lists the fact that autonegotiation is off.

NOTE On IOS XE routers, the [**no**] **negotiation auto** command controls autonegotiation. In IOS routers, configuring **speed** and **duplex** to specific values disables autonegotiation.

Bandwidth and Clock Rate on Serial Interfaces

Cisco included serial WAN topics in the CCNA exam topic from its inception in 1998 until the CCNA 200-301 Version 1.0 blueprint in early 2020. Even though Cisco no longer mentions serial links in the exam topics, the exam might show them with the expectation that you at least understand the basics, such as the fact that two routers can send data over a serial link if the router interfaces on both ends are up/up and the routers have IP addresses in the same subnet.

However, some of you will want to make serial links work in a lab because you have some serial interface cards in your lab. If so, take the time to look at a few pages in the section titled "Bandwidth and Clock Rate on Serial Interfaces," in Appendix K, "Topics from Previous Editions," which shows how to cable and configure a WAN serial link in the lab.

Router Auxiliary Port

Both routers and switches have a console port to allow administrative access, but most Cisco routers have an extra physical port called an auxiliary (Aux) port. The Aux port typically serves as a means to make a phone call to connect into the router to issue commands from the CLI.

The Aux port works like the console port, except that the Aux port is typically connected through a cable to an external analog modem, which in turn connects to a phone line. Then, the engineer uses a PC, terminal emulator, and modem to call the remote router. After being connected, the engineer can use the terminal emulator to access the router CLI, starting in user mode as usual.

Aux ports can be configured beginning with the **line aux 0** command to reach aux line configuration mode. From there, all the commands for the console line, covered mostly in Chapter 6, can be used. For example, the **login** and **password** *password* subcommands on the aux line could be used to set up simple password checking when a user dials in.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 16-5 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 16-5 Chapter Review Tracking

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

Review Element	Review Date(s)	Resource Used
Review command tables		Book
Review memory tables		Website
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 16-6 Key Topics for Chapter 16

Key Topic	Description	Page Number
List	Steps required to install a router	412
List	Similarities between a router CLI and a switch CLI	414
List	IOS features typical of routers (and all devices that perform routing)	414
Table 16-2	Router interface status codes and their meanings	417
Table 16-3	Combinations of the two interface status codes and the likely reasons for each combination	417
Table 16-4	Commands useful to display interface IPv4 addresses, masks, and interface status	419
Figure 16-7	Comparison of default autonegotiation configuration, routers with IOS and IOS XE	420
Figure 16-8	Comparison of configuration to disable autonegotiation and use a set speed/duplex, routers with IOS and IOS XE	421

Key Terms You Should Know

Cisco Catalyst Edge Platform, enterprise router, Integrated Services Router (ISR), IOS, IOS XE, SOHO router

Command References

Tables 16-7 and 16-8 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 16-7 Chapter 16 Configuration Command Reference

Command	Description
interface *type number*	Global command that moves the user into configuration mode of the named interface.
ip address *address mask*	Interface subcommand that sets the router's IPv4 address and mask.
[no] shutdown	Interface subcommand that enables (**no shutdown**) or disables (**shutdown**) the interface.

Command	Description
duplex {**full** | **half** | **auto**}	IOS (not XE) interface command that sets the duplex, or sets the use of IEEE autonegotiation, for router LAN interfaces that support multiple speeds.
speed {**10** | **100** | **1000** | **auto**}	IOS (not XE) interface command for router Gigabit (10/100/1000) interfaces that sets the speed at which the router interface sends and receives data, or sets the use of IEEE autonegotiation.
description *text*	An interface subcommand with which you can type a string of text to document information about that particular interface.
[no] negotiation auto	IOS XE (not IOS) interface command that enables or disables (**y**) the use of IEEE autonegotiation on the interface. There is no equivalent for routers that use IOS.
duplex {**full** | **half** }	IOS XE (not IOS) interface command that sets the duplex, allowed only if the interface is already configured with the **no negotiation auto** subcommand.
speed {**10** | **100** | **1000**}	IOS XE (not IOS) interface command that sets the speed, allowed only if the interface is already configured with the **no negotiation auto** subcommand.

Table 16-8 Chapter 16 EXEC Command Reference

Command	Purpose
show interfaces [*type number*]	Lists a large set of informational messages about each interface, or about the one specifically listed interface.
show ip interface brief	Lists a single line of information about each interface, including the IP address, line and protocol status, and the method with which the address was configured (manual or Dynamic Host Configuration Protocol [DHCP]).
show protocols [*type number*]	Lists information about the listed interface (or all interfaces if the interface is omitted), including the IP address, mask, and line/protocol status.
show interfaces [*type number*] **controller**	IOS XE (not IOS) command that lists detail about interface hardware and driver behavior, including detail about Ethernet autonegotiation processes and results.

CHAPTER 17

Configuring IPv4 Addresses and Static Routes

This chapter covers the following exam topics:

- **1.0 Network Fundamentals**
  - **1.6 Configure and verify IPv4 addressing and subnetting**
- **3.0 IP Connectivity**
  - **3.1 Interpret the components of routing table**
    - **3.1.a Routing protocol code**
    - **3.1.b Prefix**
    - **3.1.c Network mask**
    - **3.1.d Next hop**
    - **3.1.e Administrative distance**
    - **3.1.f Metric**
    - **3.1.g Gateway of last resort**
  - **3.2 Determine how a router makes a forwarding decision by default**
    - **3.2.a Longest prefix match**
    - **3.2.b Administrative distance**
  - **3.3 Configure and verify IPv4 and IPv6 static routing**
    - **3.3.a Default route**
    - **3.3.b Network route**
    - **3.3.c Host route**
    - **3.3.d Floating static**

Routers route IPv4 packets. That simple statement actually carries a lot of hidden meaning. For routers to route packets, routers follow a routing process, and that routing process relies on information called IP routes. Each IP route lists a destination—an IP network, IP subnet, or some other group of IP addresses. Each route also lists instructions that tell the router where to forward packets sent to addresses in that IP network or subnet. For routers to do a good job of routing packets, routers need to have a detailed, accurate list of IP routes.

Routers use three methods to add IPv4 routes to their IPv4 routing tables. Routers first learn **connected routes**, which are routes for subnets attached to a router interface. Routers can also use **static routes**, which are routes created through a configuration command (**ip route**) that tells the router what route to put in the IPv4 **routing table**. And routers can

use a routing protocol, in which routers tell each other about all their known routes, so that all routers can learn and build dynamic routes to all networks and subnets.

This chapter examines IP routing in depth with the most straightforward routes that can be added to a router's routing table. The router starts with a detailed look at the IP packet routing (forwarding process)—a process that relies on each router having useful IP routes in their routing tables. The second section then examines connected routes, which are routes to subnets that exist on the interfaces connected to the local router. The third section then examines static routes, which are routes the network engineer configures directly.

"Do I Know This Already?" Quiz

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

Table 17-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
IP Routing	1, 2
Configuring IP Addresses and Connected Routes	3
Configuring Static Routes	4–6

1. Router R1 lists a route in its routing table. Which of the following answers list a fact from a route that the router uses when matching the packet's destination address? (Choose two answers.)
   a. Mask
   b. Next-hop router
   c. Subnet ID
   d. Outgoing interface
2. PC1 sends an IP packet to PC2. To do so, PC1 sends the packet to Router R1, which routes it to Router R2, which routes it to Router R3, which routes it to the final destination (PC2). All links use Ethernet. How many routers de-encapsulate the IP packet from an Ethernet frame during its journey from PC1 to PC2?
   a. 0
   b. 1
   c. 2
   d. 3
3. After configuring a working router interface with IP address/mask 10.1.1.100/26, which of the following routes would you expect to see in the output of the **show ip route** command? (Choose two answers.)
   a. A connected route for subnet 10.1.1.64 255.255.255.192
   b. A connected route for subnet 10.1.1.0 255.255.255.0

c. A local route for host 10.1.1.100 255.255.255.192
d. A local route for host 10.1.1.100 255.255.255.255
e. A local route for host 10.1.1.64 255.255.255.255

4. Which of the following pieces of information could be listed in a correct **ip route** command on a local router? (Choose two answers.)
   a. The local router's IP address on the link between the two routers
   b. The next-hop router's IP address on the link between the two routers
   c. The next-hop router's interface ID on the link between the two routers
   d. The local router's interface ID on the link between the two routers
5. Which of the following commands correctly configures a static route?
   a. **ip route 10.1.3.0 255.255.255.0 10.1.130.253**
   b. **ip route 10.1.3.0 serial 0**
   c. **ip route 10.1.3.0 /24 10.1.130.253**
   d. **ip route 10.1.3.0 /24 serial 0**
6. A network engineer configures the **ip route 10.1.1.0 255.255.255.0 s0/0/0** command on a router and then issues a **show ip route** command from enable mode. No routes for subnet 10.1.1.0/24 appear in the output. Which of the following could be true?
   a. The **ip route** command has incorrect syntax and was rejected in config mode.
   b. Interface s0/0/0 is down.
   c. The router has no up/up interfaces in Class A network 10.0.0.0.
   d. The **ip route** command is missing a next-hop router IP address.

Foundation Topics

IP Routing

IP routing—the process of forwarding IP packets—delivers packets across entire TCP/IP networks, from the device that originally builds the IP packet to the device that is supposed to receive the packet. In other words, IP routing delivers IP packets from the sending host to the destination host.

The complete end-to-end routing process relies on network layer logic on hosts and on routers. The sending host uses Layer 3 concepts to create an IP packet, forwarding the IP packet to the host's default gateway (default router). The process requires Layer 3 logic on the routers as well, by which the routers compare the destination address in the packet to their routing tables, to decide where to forward the IP packet next.

The routing process also relies on data-link and physical details at each link. IP routing relies on serial WAN links, Ethernet WAN links, Ethernet LANs, wireless LANs, and many other networks that implement data-link and physical layer standards. These lower-layer devices and protocols move the IP packets around the TCP/IP network by encapsulating and transmitting the packets inside data-link layer frames.

The previous two paragraphs summarize the key concepts about IP routing as introduced back in Chapter 3, "Fundamentals of WANs and IP Routing." Next, this section reviews IP routing, while taking the discussion another step or two deeper, taking advantage of the additional depth of knowledge discussed in all the earlier chapters in this book.

IPv4 Routing Process Reference

Because you already saw the basics back in Chapter 3, this section collects the routing process into steps for reference. The steps use many specific Ethernet LAN terms discussed in Parts II and III of this book and some IP addressing terms discussed in Part IV. The upcoming descriptions and example then discuss these summaries of routing logic to make sure that each step is clear.

The routing process starts with the host that creates the IP packet. First, the host asks the question: Is the destination IP address of this new packet in my local subnet? The host uses its own IP address/mask to determine the range of addresses in the local subnet. Based on its own opinion of the range of addresses in the local subnet, a LAN-based host acts as follows:

Key Topic

Step 1. If the destination is local, send directly:

- **a.** Find the destination host's MAC address. Use the already-known Address Resolution Protocol (ARP) table entry, or use ARP messages to learn the information.
- **b.** Encapsulate the IP packet in a data-link frame, with the destination data-link address of the destination host.

Step 2. If the destination is not local, send to the default gateway:

- **a.** Find the default gateway's MAC address. Use the already-known Address Resolution Protocol (ARP) table entry, or use ARP messages to learn the information.
- **b.** Encapsulate the IP packet in a data-link frame, with the destination data-link address of the default gateway.

17

Figure 17-1 summarizes these same concepts. In the figure, host A sends a local packet directly to host D. However, for packets to host B, on the other side of a router and therefore in a different subnet, host A sends the packet to its default router (R1). (As a reminder, the terms *default gateway* and *default router* are synonyms.)

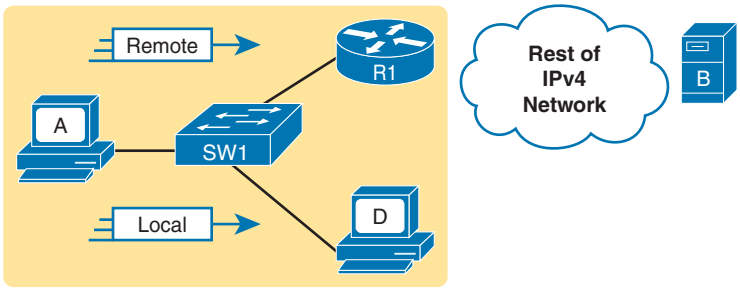

Figure 17-1 *Host Routing Logic Summary*

Routers have a little more routing work to do as compared with hosts. While the host logic began with an IP packet sitting in memory, a router has some work to do before getting to that point. With the following five-step summary of a router's routing logic, the router takes the first two steps just to receive the frame and extract the IP packet, before thinking about the packet's destination address at Step 3. The steps are as follows:

Key Topic

1. For each received data-link frame, choose whether or not to process the frame. Process it if
   a. The frame has no errors (per the data-link trailer Frame Check Sequence [FCS] field).
   b. The frame's destination data-link address is the router's address (or an appropriate multicast or broadcast address).
2. If choosing to process the frame at Step 1, de-encapsulate the packet from inside the data-link frame.
3. Make a routing decision. To do so, compare the packet's destination IP address to the routing table and find the route that matches the destination address. This route identifies the **outgoing interface** of the router and possibly the **next-hop router**.
4. Encapsulate the packet into a data-link frame appropriate for the outgoing interface. When forwarding out LAN interfaces, use ARP as needed to find the next device's MAC address.
5. Transmit the frame out the outgoing interface, as listed in the matched IP route.

This routing process summary lists many details, but sometimes you can think about the routing process in simpler terms. For example, leaving out some details, this paraphrase of the step list details the same big concepts:

> The router receives a frame, removes the packet from inside the frame, decides where to forward the packet, puts the packet into another frame, and sends the frame.

To give you a little more perspective on these steps, Figure 17-2 breaks down the same five-step routing process as a diagram. The figure shows a packet arriving from the left, entering a router Ethernet interface, with an IP destination of host C. The figure shows the packet arriving, encapsulated inside an Ethernet frame (both header and trailer).

Key Topic

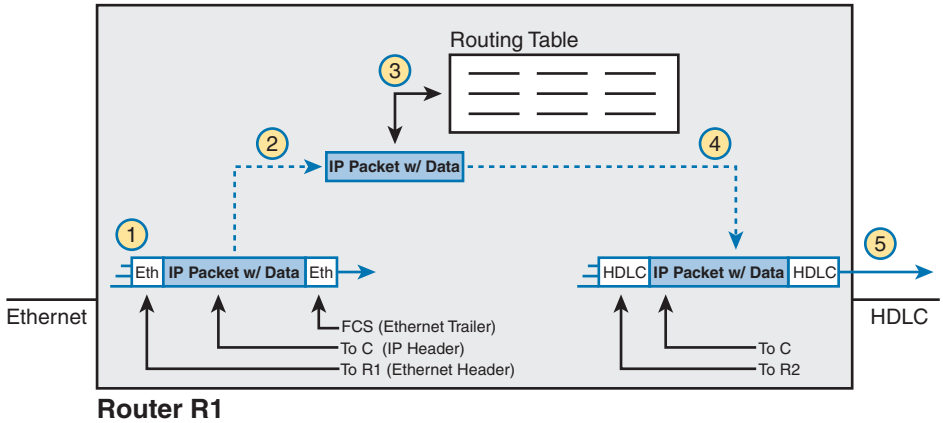

Figure 17-2 *Router Routing Logic Summary*

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

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

Router R1 processes the frame and packet as shown with the numbers in the figure, matching the same five-step process described just before the figure, as follows:

1. Router R1 notes that the received Ethernet frame passes the FCS check and that the destination Ethernet MAC address is R1's MAC address, so R1 processes the frame.
2. R1 de-encapsulates the IP packet from inside the Ethernet frame's header and trailer.
3. R1 compares the IP packet's destination IP address to R1's IP routing table.
4. R1 encapsulates the IP packet inside a new data-link frame, in this case, inside a High-Level Data Link Control (HDLC) header and trailer.
5. R1 transmits the IP packet, inside the new HDLC frame, out the serial link on the right.

NOTE This chapter uses several figures that show an IP packet encapsulated inside a data-link layer frame. These figures often show both the data-link header as well as the data-link trailer, with the IP packet in the middle. The IP packets all include the IP header, plus any encapsulated data.

An Example of IP Routing

The next several pages walk you through an example that discusses each routing step, in order, through multiple devices. The example uses a case in which host A (172.16.1.9) sends a packet to host B (172.16.2.9), with host routing logic and the five steps showing how R1 forwards the packet.

Figure 17-3 shows a typical IP addressing diagram for an IPv4 network with typical address abbreviations. The diagram can get a little too messy if it lists the full IP address for every router interface. When possible, these diagrams usually list the subnet and then the last octet or two of the individual IP addresses—just enough so that you know the IP address but with less clutter. For example, host A uses IP address 172.16.1.9, taking from subnet 172.16.1.0/24 (in which all addresses begin 172.16.1) and the .9 beside the host A icon. As another example, R1 uses address 172.16.1.1 on its LAN interface, 172.16.4.1 on one serial interface, and 172.16.5.1 on an Ethernet WAN interface.

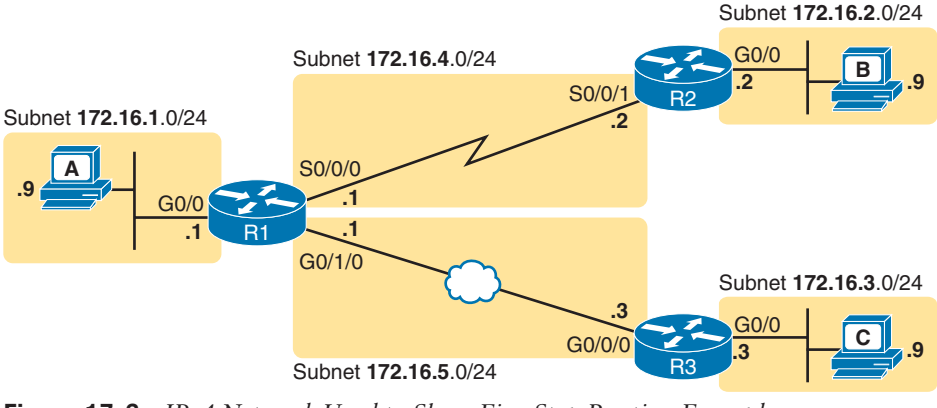

Figure 17-3 *IPv4 Network Used to Show Five-Step Routing Example*

Now on to the example, with host A (172.16.1.9) sending a packet to host B (172.16.2.9).

Host Forwards the IP Packet to the Default Router (Gateway)

In this example, host A uses some application that sends data to host B (172.16.2.9). After host A has the IP packet sitting in memory, host A's logic reduces to the following:

- My IP address/mask is 172.16.1.9/24, so my local subnet contains numbers 172.16.1.0–172.16.1.255 (including the subnet ID and subnet broadcast address).
- The destination address is 172.16.2.9, which is clearly not in my local subnet.
- Send the packet to my default gateway, which is set to 172.16.1.1.
- To send the packet, encapsulate it in an Ethernet frame. Make the destination MAC address be R1's G0/0 MAC address (host A's default gateway).

Figure 17-4 pulls these concepts together, showing the destination IP address and destination MAC address in the frame and packet sent by host A in this case. Note that the figure uses a common drawing convention in networking, showing an Ethernet as a few lines, hiding all the detail of the Layer 2 switches.

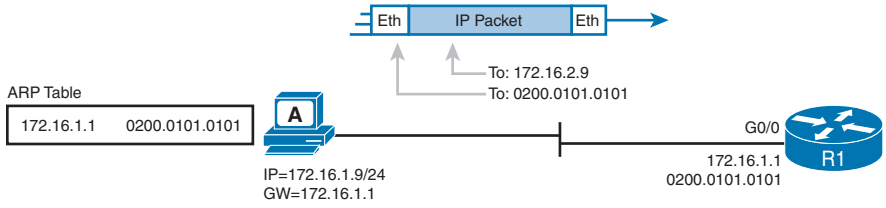

Figure 17-4 *Host A Sends Packet Through Router R1 to Host B*

Routing Step 1: Decide Whether to Process the Incoming Frame

Routers receive many frames in an interface, particularly LAN interfaces. However, a router can and should ignore some of those frames. So, the first step in the routing process begins with a decision of whether a router should process the frame or silently discard (ignore) the frame.

First, the router does a simple but important check (Step 1A in the process summary) so that the router ignores all frames that had bit errors during transmission. The router uses the data-link trailer's FCS field to check the frame, and if errors occurred in transmission, the router discards the frame. (The router makes no attempt at error recovery; that is, the router does not ask the sender to retransmit the data.)

The router also checks the destination data-link address (Step 1B in the summary) to decide whether the frame is intended for the router. For example, frames sent to the router's unicast MAC address for that interface are clearly sent to that router. However, a router can actually receive a frame sent to some other unicast MAC address, and routers should ignore these frames.

For example, think back to how LAN switches forward unknown unicast frames—frames for which the switch does not list the destination MAC address in the MAC address table. The LAN switch floods those frames. The result? Routers sometimes receive frames destined for

some other device, with some other device's MAC address listed as the destination MAC address. Routers should ignore those frames.

Figure 17-5 shows a working example in which host A sends a frame destined for R1's MAC address. After receiving the frame, R1 confirms with the FCS that no errors occurred, and R1 confirms that the frame is destined for R1's MAC address (0200.0101.0101 in this case). All checks have been passed, so R1 will process the frame. (Note that the large rectangle in the figure represents the internals of Router R1.)

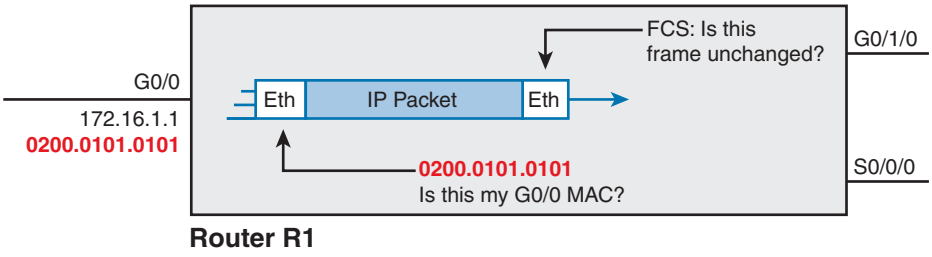

Figure 17-5 *Routing Step 1, on Router R1: Checking FCS and Destination MAC*

Routing Step 2: De-encapsulation of the IP Packet

After the router knows that it ought to process the received frame (per Step 1), the next step is relatively simple: de-encapsulating the packet. In router memory, the router no longer needs the original frame's data-link header and trailer, so the router removes and discards them, leaving the IP packet, as shown in Figure 17-6. Note that the destination IP address remains unchanged (172.16.2.9).

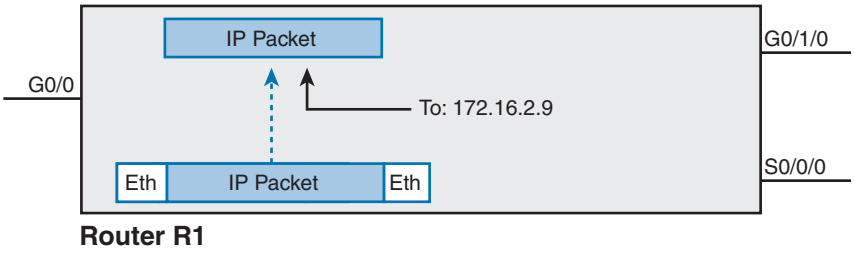

Figure 17-6 *Routing Step 2 on Router R1: De-encapsulating the Packet*

Routing Step 3: Choosing Where to Forward the Packet

While routing Step 2 required little thought, Step 3 requires the most thought of all the steps. At this point, the router needs to make a choice about where to forward the packet next. That process uses the router's IP routing table, with some matching logic to compare the packet's destination address with the table.

First, an IP routing table lists multiple routes. Each individual route contains several facts, which in turn can be grouped as shown in Figure 17-7. Part of each route is used to match the destination address of the packet, while the rest of the route lists forwarding instructions: where to send the packet next.

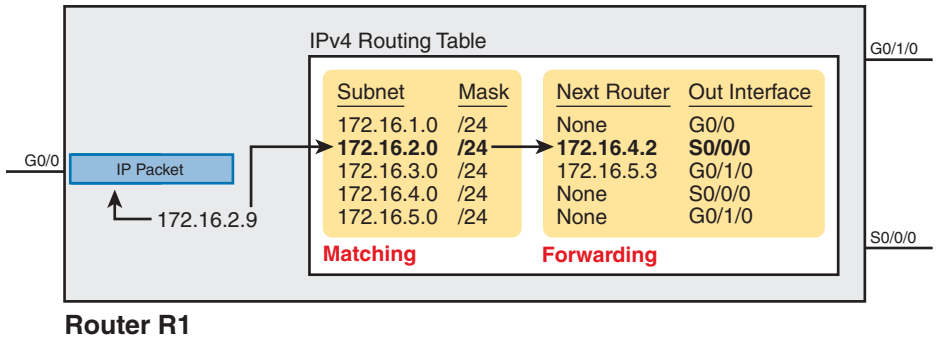

Figure 17-7 *Routing Step 3 on Router R1: Matching the Routing Table*

Focus on the entire routing table for a moment, and notice the fact that it lists five routes. Earlier, Figure 17-3 showed the entire example network, with five subnets, so R1 has a route for each of the five subnets.

Next, look at the part of the five routes that Router R1 will use to match packets. To fully define each subnet, each route lists both the subnet ID and the subnet mask. When matching the IP packet's destination with the routing table, the router looks at the packet's destination IP address (172.16.2.9) and compares it to the range of addresses defined by each subnet. Specifically, the router looks at the subnet and mask information; with a little math, the router can figure out in which of these subnets 172.16.2.9 resides (the route for subnet 172.16.2.0/24).

Finally, look to the right side of the figure, to the forwarding instructions for these five routes. After the router matches a specific route, the router uses the forwarding information in the route to tell the router where to send the packet next. In this case, the router matched the route for subnet 172.16.2.0/24, so R1 will forward the packet out its own interface S0/0/0, to Router R2 next, listed with its next-hop router IP address of 172.16.4.2.

NOTE Routes for remote subnets typically list both an outgoing interface and next-hop router IP address. Routes for subnets that connect directly to the router list only the outgoing interface because packets to these destinations do not need to be sent to another router.

Routing Step 4: Encapsulating the Packet in a New Frame

At this point, the router knows how it will forward the packet. However, routers cannot forward a packet without first wrapping a data-link header and trailer around it (encapsulation).

Encapsulating packets for serial links does not require a lot of thought, but the current CCNA exam does not require a lot from us. Point-to-point serial WAN links use either HDLC (the default) or PPP as the data-link protocol. However, we can ignore any data-link logic, even ignoring data-link addressing, because serial links have only two devices on the link: the sender and the then-obvious receiver; the data-link addressing does not matter. In this example, R1 forwards the packet out S0/0/0, after encapsulating the packet inside an HDLC frame, as shown in Figure 17-8.

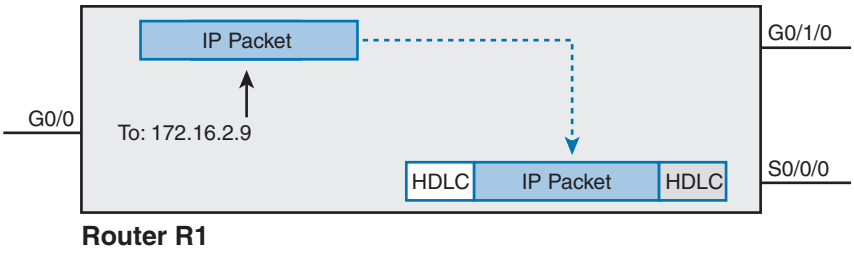

Figure 17-8 *Routing Step 4 on Router R1: Encapsulating the Packet*

Note that with some other types of data links, the router has a little more work to do at this routing step. For example, sometimes a router forwards packets out an Ethernet interface. To encapsulate the IP packet, the router would need to build an Ethernet header, and that Ethernet header's destination MAC address would need to list the correct value.

For example, consider a packet sent by that same PC A (172.16.1.9) in Figure 17-3 but with a destination of PC C (172.16.3.9). When R1 processes the packet, R1 matches a route that tells R1 to forward the packet out R1's G0/1/0 Ethernet interface to 172.16.5.3 (R3) next. R1 needs to put R3's MAC address in the header, and to do that, R1 uses its IP ARP table information, as shown in Figure 17-9. If R1 did not have an ARP table entry for 172.16.5.3, R1 would first have to use ARP to learn the matching MAC address.

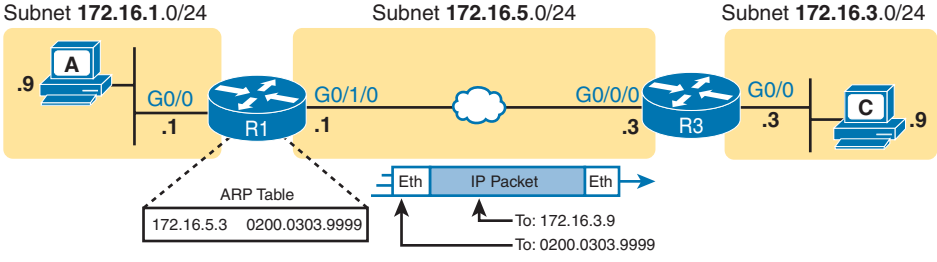

Figure 17-9 *Routing Step 4 on Router R1 with a LAN Outgoing Interface*

Routing Step 5: Transmitting the Frame

After the frame has been prepared, the router simply needs to transmit the frame. The router might have to wait, particularly if other frames are already waiting their turn to exit the interface.

Configuring IP Addresses and Connected Routes

Cisco routers enable IPv4 routing globally, by default. Then, to make the router be ready to route packets on a particular interface, the interface must be configured with an IP address and the interface must be configured such that it comes up, reaching a "line status up, line protocol up" state. Only at that point can routers route IP packets in and out a particular interface.

After a router can route IP packets out one or more interfaces, the router needs some routes. Routers can add routes to their routing tables through three methods:

Key Topic

Connected routes: Added because of the configuration of the **ip address** interface subcommand on the local router

Static routes: Added because of the configuration of the **ip route** global command on the local router

Routing protocols: Added as a function by configuration on all routers, resulting in a process by which routers dynamically tell each other about the network so that they all learn routes

This second of three sections discusses several variations on how to configure connected routes, while the next major section discusses static routes.

Connected Routes and the ip address Command

A Cisco router automatically adds a route to its routing table for the subnet connected to each interface, assuming that the following two facts are true:

- The interface is in a working state. In other words, the interface status in the **show interfaces** command lists a line status of up and a protocol status of up.
- The interface has an IP address assigned through the **ip address** interface subcommand.

The concept of connected routes is relatively basic. The router, of course, needs to know the subnet number connected to each of its interfaces, so the router can route packets to that subnet. The router does the math, taking the interface IP address and mask and calculating the subnet ID. However, the router only needs that route when the interface is up and working, so the router includes a connected route in the routing table only when the interface is working.

Example 17-1 shows the connected routes on Router R1 in Figure 17-10. The first part of the example shows the configuration of IP addresses on all three of R1's interfaces. The end of the example lists the output from the **show ip route** command, which lists these routes with a C as the route code, meaning *connected*.

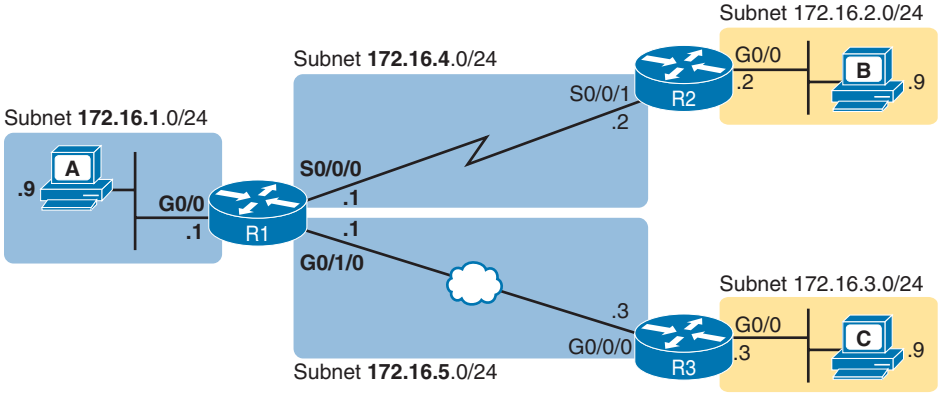

Figure 17-10 *Sample Network to Show Connected Routes*

Example 17-1 *Connected and Local Routes on Router R1*

```
! Excerpt from show running-config follows...
!
interface GigabitEthernet0/0
 ip address 172.16.1.1 255.255.255.0
!
interface Serial0/0/0
 ip address 172.16.4.1 255.255.255.0
!
interface GigabitEthernet0/1/0
 ip address 172.16.5.1 255.255.255.0

R1# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR

Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C        172.16.1.0/24 is directly connected, GigabitEthernet0/0
L        172.16.1.1/32 is directly connected, GigabitEthernet0/0
C        172.16.4.0/24 is directly connected, Serial0/0/0
L        172.16.4.1/32 is directly connected, Serial0/0/0
C        172.16.5.0/24 is directly connected, GigabitEthernet0/1/0
L        172.16.5.1/32 is directly connected, GigabitEthernet0/1/0
```


Each time you do a lab or see an example with output from the **show ip route** command, look closely at the heading line for each classful network and the indented lines that follow. The indented lines list routes to specific subnets. In Example 17-1, the heading line shows Class B network 172.16.0.0 as "172.16.0.0/16", with /16 representing the default mask for a Class B network. Why? IOS groups the output by Class A, B, or C network. In this case, the output shows a heading line that tells us that the following indented lines have to do with network 172.16.0.0/16.

Take a moment to look closely at each of the three highlighted routes below the heading line that references Class B network 172.16.0.0. Each lists a C in the first column, and each has text that says "directly connected"; both identify the route as connected to the router. The early part of each route lists the matching parameters (subnet ID and mask), as shown in the earlier example in Figure 17-7. The end of each of these routes lists the outgoing interface.

Note that the router also automatically produces a different kind of route, called a *local route*. The local routes define a route for the one specific IP address configured on the router interface. Each local route has a /32 prefix length, defining a *host route*, which defines a route just for that one IP address. For example, the last local route, for 172.16.5.1/32, defines a route that matches only the IP address of 172.16.5.1. Routers use these local routes that list their own local IP addresses to more efficiently forward packets sent to the router itself.

The **show ip route** command in the example reveals a few of the specific subitems within exam topic 3.1 (per CCNA 200-301 V1.1), with later examples in this chapter revealing even more details. This section shows details related to the following terms from the exam topics:

- **Routing Protocol Code:** The legend at the top of the **show ip route** output (about nine lines) lists all the routing protocol codes (exam topic 3.1.a). This book references the codes for connected routes (C), local (L), static (S), and OSPF (O).
- **Prefix:** The word *prefix* (exam topic 3.1.b) is just another name for subnet ID.
- **Mask:** Each route lists a prefix (subnet ID) and network mask (exam topic 3.1.c) in prefix format, for example, /24.

Common Mistakes with the ip address Subcommand

If you follow a correct IP address plan, the **ip address** commands on your routers should be accepted. They should also work correctly to support endpoint hosts in LANs in their roles as the default gateway. However, several mistakes are possible with this command, mistakes that might not be obvious at first glance.

This next topic examines a few of those mistakes, using Figure 17-11 as a backdrop. The figure shows a more detailed view of the LAN connected to Router R1 in Figure 17-10, now with two PCs in it.

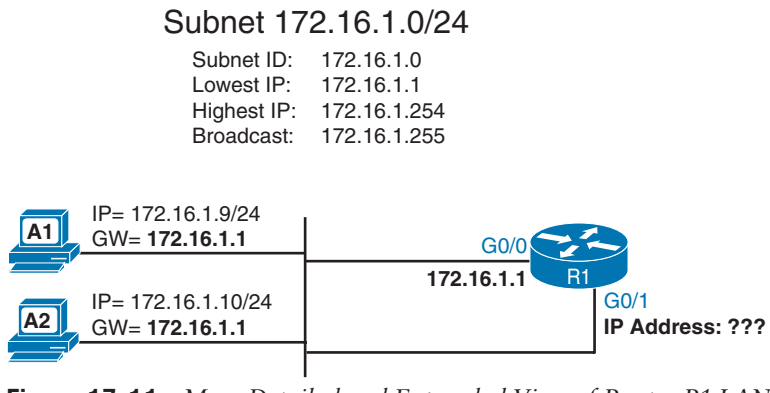

Figure 17-11 *More Detailed and Expanded View of Router R1 LAN*

First, IOS rejects the **ip address** command in configuration mode if it uses a reserved number in the subnet, such as the subnet ID or the subnet broadcast address. Example 17-2 shows three rejected **ip address** commands. The first attempts to configure the subnet ID as the address, while the second attempts to configure the subnet broadcast address. The final

example shows the command with an invalid subnet mask (which is also rejected). Note that the error message does not reveal the specific reason, instead giving a cryptic reference to the idea that the values have a problem when using that mask.

Example 17-2 *IOS Rejects* **ip address** *Commands with Reserved Addresses*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# ip address 172.16.1.0 255.255.255.0
Bad mask /24 for address 172.16.1.0
R1(config-if)# ip address 172.16.1.255 255.255.255.0
Bad mask /24 for address 172.16.1.255
R1(config-if)# ip address 172.16.1.1 255.0.255.0
Bad mask 0xFF00FF00 for address 172.16.1.1
R1(config-if)#
```

You can also configure the **ip address** command with values that IOS accepts but that are incorrect per your address plan—resulting in problems in the network. For example, looking at Figure 17-11, the default gateway setting on the two PCs implies that router R1's address should be 172.16.1.1. Router R1 would accept the **ip address 172.16.1.2 255.255.255.0** command on its G0/0 interface, setting the wrong address value—but the router has no mechanism to know that its address does not match the PC's default gateway settings. Instead, the PCs cannot communicate outside the subnet. The solution: Configure the correct address on Router R1.

As a third issue, one router may connect only one interface to a subnet. In Figure 17-11, Router R1 has two interfaces (G0/0 and G0/1) connected to the same LAN. If you attempted to assign both interfaces an IP address in that same 172.16.1.0/24 subnet, and both interfaces were in an up/up state, IOS would reject the second **ip address** command. Example 17-3 shows that sequence, with R1's G0/0 configured first and R1's G0/1 configured second (with the rejection).

Example 17-3 *IOS Rejects* **ip address** *Command for Second Interface in the Same Subnet*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# ip address 172.16.1.1 255.255.255.0
R1(config)# interface gigabitEthernet 0/1
R1(config-if)# ip address 172.16.1.2 255.255.255.0
% 172.16.1.0 overlaps with GigabitEthernet0/0
R1(config-if)#
```

The ARP Table on a Cisco Router

After a router has added these connected routes, the router can route IPv4 packets between those subnets. To do so, the router makes use of its IP ARP table.

The IPv4 **ARP table** lists the IPv4 address and matching MAC address of hosts connected to the same subnet as the router. When forwarding a packet to a host on the same subnet, the router encapsulates the packet, with a destination MAC address as found in the ARP table. If the router wants to forward a packet to an IP address on the same subnet as the router but does not find an ARP table entry for that IP address, the router will use ARP messages to learn that device's MAC address.

Example 17-4 shows R1's ARP table based on the previous example. The output lists R1's own IP address of 172.16.1.1, with an age of -, meaning that this entry does not time out. Dynamically learned ARP table entries have an upward counter, like the 35-minute value for the ARP table entry for IP address 172.16.1.9. By default, IOS will time out (remove) an ARP table entry after 240 minutes in which the entry is not used. (IOS resets the timer to 0 when an ARP table entry is used.) Note that to experiment in the lab, you might want to empty all dynamic entries (or a single entry for one IP address) using the **clear ip arp** [*ip-address*] EXEC command.

Example 17-4 *Displaying a Router's IP ARP Table*

```
R1# show ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  172.16.1.1             -   0200.0101.0101  ARPA   GigabitEthernet0/0
Internet  172.16.1.9            35   0200.aaaa.aaaa  ARPA   GigabitEthernet0/0
```

Thinking about how Router R1 forwards a packet to host A (172.16.1.9), over that final subnet, R1 does the following:

1. R1 looks in the ARP table for an entry for 172.16.1.9.
2. R1 encapsulates the IP packet in an Ethernet frame, adding destination 0200.aaaa.aaaa to the Ethernet header (as taken from the ARP table).
3. R1 transmits the frame out interface G0/0.

Configuring Static Routes

In real networks, you will find connected routes on every router, which the routers create for subnets connected to their interfaces. You will use dynamic routing protocols like OSPF and EIGRP to learn routes for the rest of the subnets in the enterprise, routes for subnets remote from the local router.

Enterprises use static routes—that is, routes added to a routing table through direct configuration of the **ip route** command—much less often than connected routes and routes learned with dynamic routing protocols. However, static routes can be useful at times. Studying static routes also happens to be a wonderful learning tool as well. And, of course, the exam topics list several variations of static routes, so you need to learn the topic for the exam.

NOTE The CCNA 200-301 exam V1.1, exam topic 3.3, subdivides IPv4 (and IPv6) static routes into four subtopics: network routes, default routes, host routes, and floating static routes. This section explains all four types for IPv4 as noted in the upcoming headings.

Static Network Routes

IOS allows the definition of individual static routes using the **ip route** global configuration command. Every **ip route** command defines a destination subnet with a subnet ID and mask. The command also lists the forwarding instructions, listing the outgoing interface or the next-hop router's IP address (or both). IOS then takes that information and adds that route to the IP routing table.

A static **network route** defines either a subnet or an entire Class A, B, or C network in the **ip route** command. In contrast, a **default route** defines the set of all destination IP addresses, while a **host route** defines a single IP address (that is, an address of one host).

Figure 17-12 shows a new network diagram to be used in the upcoming examples. Note that the design uses the same subnets as some previous figures in this chapter, but it uses two GigabitEthernet WAN links and no serial links.

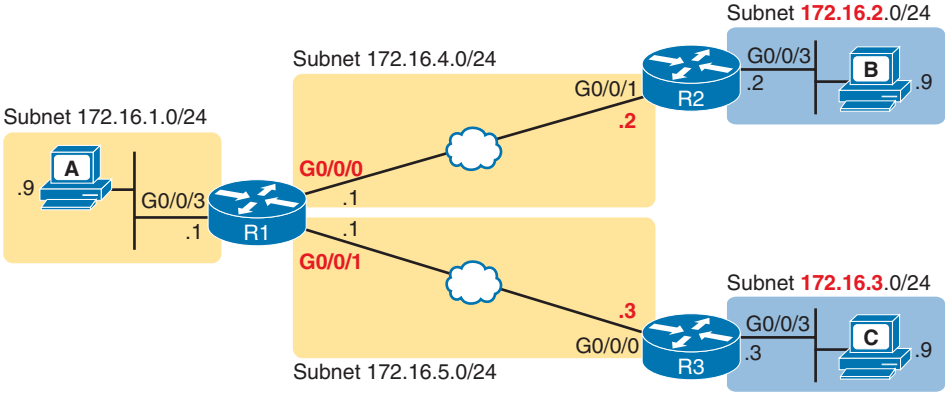

Figure 17-12 *Sample Network Used in Static Route Configuration Examples*

Figure 17-13 shows the concepts behind the potential network routes on Router R1 for the subnet to the right of Router R2 (subnet 172.16.2.0/24). To create that static network route on R1, R1 will configure the subnet ID and mask. For the forwarding instructions, R1 will configure either R1's outgoing interface (G0/0/0) or R2's IP address as the next-hop router IP address (172.16.4.2).

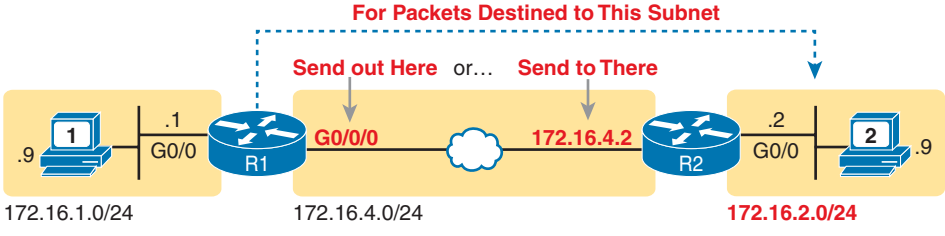

Figure 17-13 *Static Route Configuration Concept*

Example 17-5 shows the configuration of a couple of sample static routes, using the three-router topology in Figure 17-12. The example shows static routes on R1, one route for each of the two subnets on the right side of the figure. The first command defines a route for subnet 172.16.2.0 255.255.255.0, the subnet off Router R2. The forwarding instruction parameters list only R1's outgoing interface, G0/0/0. This route basically states: To send packets to the subnet off Router R2, send them out my local G0/0/0 interface.

Example 17-5 *Static Routes Added to R1*

```
ip route 172.16.2.0 255.255.255.0 G0/0/0
ip route 172.16.3.0 255.255.255.0 172.16.5.3
```

The second route has the same kind of logic, except listing the neighboring router's IP address on the WAN link as the next-hop router. This route basically says this: To send packets to the subnet off Router R3 (172.16.3.0 255.255.255.0), send the packets to R3's WAN IP address next (172.16.5.3).

Verifying Static Network Routes

Static routes, when seen in the IP routing table, list the same parameters included in the configuration command. To see how, closely compare the output in Example 17-6's **show ip route static** command. This command lists only static routes, in this case listing the two static routes configured in Example 17-5. First, the two lines listing the static routes begin with legend code S, meaning static. Both list the subnet and mask defined in the respective routes, although the output uses prefix-style masks. But note the key difference: for the route configured with only the outgoing interface, the route lists only the outgoing interface, and for the route configured with only the next-hop IP address, the output lists only the next-hop address.

Example 17-6 *Static Routes Added to R1*

```
R1# show ip route static
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! Legend lines omitted for brevity

      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
S        172.16.2.0/24 is directly connected, GigabitEthernet0/0/0S
S        172.16.3.0/24 [1/0] via 172.16.5.3
```

Although statically configured, IOS adds and removes static routes from the routing table over time based on whether the outgoing interface is working or not. For example, in this case, if R1's G0/0/0 interface fails, R1 removes the static route to 172.16.2.0/24 from the IPv4 routing table. Later, when the interface comes up again, IOS adds the route back to the routing table.

Examples 17-3 and 17-4 show how to configure static network routes to support left-to-right packet flow in Figure 17-12, but routers need routes for all subnets to support packet flow in all directions. Even in a small lab used for CCNA learning, if you use only static routes, you need static routes for all subnets for both directions of packet flow. For instance, in Figure 17-13, to support packet flow from PC A to PC B, you need:

A route on Router R1 for PC B's subnet 172.16.2.0/24

A route on Router R2 for PC A's subnet 172.16.1.0/24

Ethernet Outgoing Interfaces and Proxy ARP

While both styles of static routes in Example 17-5 work—with either next-hop IP address or outgoing interface—using the next-hop address is better. Both work, but using a next-hop address causes much less confusion when the staff begin to wonder about how those static routes work behind the scenes. In short, using the next-hop address does not force the use of proxy ARP on the neighboring router, but using an outgoing interface on an Ethernet link does require proxy ARP.

This section attempts to explain how proxy ARP works and how static routes make use of it. To begin, consider this general definition of proxy ARP:

1. A router receives in interface X an ARP request, whose target is not in the subnet connected to interface X.
2. The router has a route to forward packets to that target address, and that route should not forward the packet back out onto the same interface causing a loop. In other words, the router has a useful route to forward packets to the target.
3. The router is therefore willing and useful to act as a proxy for the target host. To do so, the router supplies its own MAC address in the ARP Reply.

While true, it helps to work through an example. Consider Example 17-5 again with the **ip route 172.16.2.0 255.255.255.0 G0/0/0** command. Imagine PC A sends a packet to PC B at address 172.16.2.9. The packet arrives at Router R1, R1 matches the packet to the static route for subnet 172.16.2.0/24, and now Router R1 must decide from the forwarding instructions how to forward the packet. Router R1 uses logic as follows:

1. The route's forwarding instructions show the destination subnet as connected to interface G0/0/0. That causes R1 to send an ARP Request looking for 172.16.2.9 as the target address. In effect, R1's route makes R1 behave as if the destination subnet 172.16.2.0/24 is connected to port G0/0/0. Of course, PC B will not receive the ARP Request, as it resides on the other side of a router.
2. R2 receives R1's ARP for target 172.16.2.9. The ARP meets the requirements of proxy ARP (an address not in the local subnet, but R2 has a useful route matching address 172.16.2.9). R2 sends an ARP reply listing R2's interface G0/0/1 MAC—a proxy ARP Reply.
3. R1 receives the ARP Reply, so R1 can now forward packets meant for 172.16.2.9. R1 forwards these packets encapsulated in an Ethernet frame, with R2's G0/0/1 MAC as the destination MAC address.
4. R2 receives the frame and routes the encapsulated packet to PC B, as with normal routing processes.

As you can understand from the previous page or so, the underlying logic behind using an outgoing Ethernet interface on a route requires quite a lot of analysis. For real-world applications, use static routes with a next-hop address instead.

Static Default Routes

When a router tries to route a packet, the router might not match the packet's destination IP address with any route. When that happens, the router normally just discards the packet.

Routers can use a *default route* to match all packets, so that if a packet does not match any other more specific route in the routing table, the router will forward the packet based on

the default route. Routers can learn default routes with a routing protocol, or you can configure a default static route.

Figure 17-14 shows one classic example in which a company has many branch office routers, like R2 and R3 in the figure, each of which has a single WAN connection. No matter how many routes a router might learn for specific subnets in those isolated branch offices, all the forwarding details in those routes would be the same for each route. For instance, all routes on Router R2, for remote subnets, would use outgoing interface G0/0/1 and next-hop address 172.16.4.1. Router R2 could instead use a default route, which matches all destinations, with the same forwarding instructions.

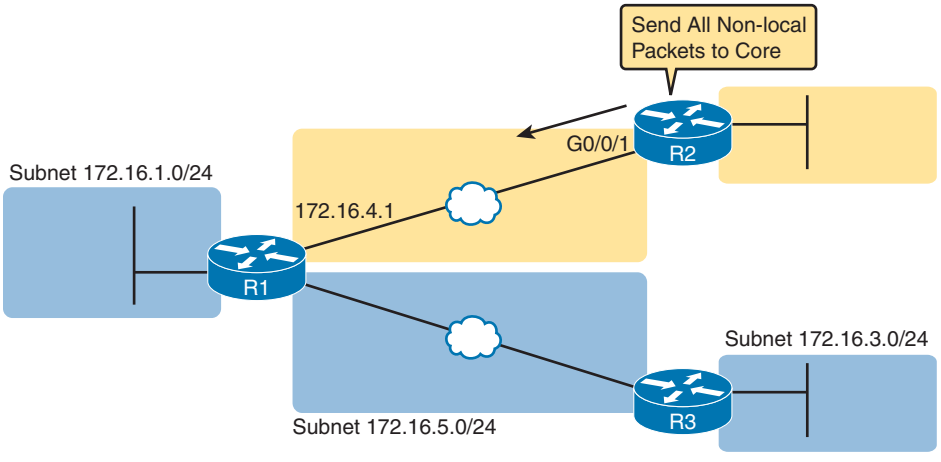

Figure 17-14 *Example Use of Static Default Routes*

IOS allows the configuration of a static default route by using special values for the subnet and mask fields in the **ip route** command: 0.0.0.0 and 0.0.0.0. For example, the command **ip route 0.0.0.0 0.0.0.0 172.16.4.1** creates the static route depicted in Figure 17-14 on Router R2—a route that matches all IP packets—and sends those packets to R1 (172.16.4.1). Example 17-7 shows that static default route, but take care to notice the unexpected location of the route for 0.0.0.0/0.

Example 17-7 *Adding a Static Default Route on R2 (Figure 17-14)*

```
R2# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)# ip route 0.0.0.0 0.0.0.0 172.16.4.1
R2(config)# ^Z
R2# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
```

```
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is 172.16.4.1 to network 0.0.0.0S*

S*    0.0.0.0/0 [1/0] via 172.16.4.1
      172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        172.16.2.0/24 is directly connected, GigabitEthernet0/0/0
L        172.16.2.2/32 is directly connected, GigabitEthernet0/0/0
C        172.16.4.0/24 is directly connected, GigabitEthernet0/0/1
L        172.16.4.2/32 is directly connected, GigabitEthernet0/0/1
```

The output of the **show ip route** command lists a few new and interesting facts. First, it lists the route as 0.0.0.0/0, another notation for a subnet and mask of 0.0.0.0 and 0.0.0.0. The output lists a code of S, meaning static, but also with a *, meaning it is a *candidate default route*. (A router can learn about more than one default route, and the router then has to choose which one to use; the * means that it is at least a candidate to become the default route.) Just above the list of routes, the "Gateway of Last Resort" line refers to the chosen default route; in this case, R2 has only one candidate, so the output shows the just-configured static default route with next-hop address 172.16.4.1.

NOTE If you attempt to re-create some of the examples from the book as part of your lab practice, note that if you configure the network as shown in Figures 17-13 and 17-14, with the configuration in Examples 17-5 and 17-7, PCs A and B should be able to ping each other. As an exercise, you can think about what network or default static routes to add to R3 so that PC C can ping the other two PCs as well.

Static Host Routes

The term *network route* refers to a route that matches addresses in an entire IP network or subnet, while a default route refers to a route that matches all destinations. In contrast, the term *host route* defines a route to a single host address. To configure such a static route, the **ip route** command uses an IP address plus a mask of 255.255.255.255 so that the matching logic matches just that one address.

Why use a static host route? Honestly, it is the least likely style of static route to use because network engineers seldom need to solve some routing problem for which a static host route will help. However, just to get the gist of what it can do, consider the topology in Figure 17-15, which expands the topology shown in the previous few figures.

First, imagine you run a dynamic routing protocol like OSPF, and Router R1 learns a route for subnet 172.16.10.0/26. Depending on the conditions of the network, R1's route for 172.16.10.0/26 may send packets to R2 next or R3 next. Most enterprises would be content in either case. The figure shows that route running through the lower router, R3.

Now imagine there was some unusual business reason such that all packets meant for host D must flow through Router R2 if it is working. The figure also shows that route, for host D only (172.16.10.4).

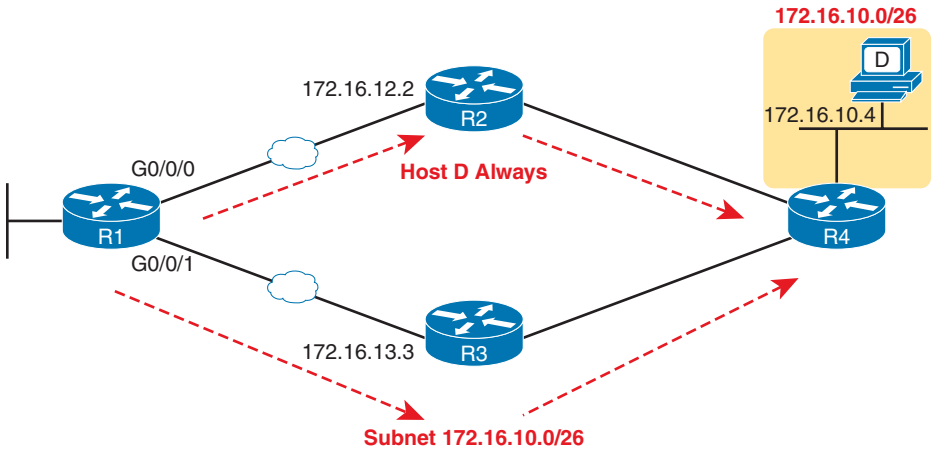

Figure 17-15 *An Example Possible Use for a Static Host Route*

The preceding scenario creates a need for a static host route. For instance, all the routers use OSPF to dynamically learn network routes for all subnets in the design. However, the engineer also configures the one host route on R1 as discussed in the last few paragraphs, as seen at the top of Example 17-8.

Example 17-8 *Host Route Configuration Versus an Overlapping Static Network Route*

```
R1# configure terminal
! The static host route for one IP address within that same subnet
R1(config)# ip route 172.16.10.4 255.255.255.255 172.16.4.2
R1(config)# ^Z
R1# show ip route
! Irrelevant portions omitted for brevity

      172.16.0.0/16 is variably subnetted, 12 subnets, 4 masks
O IA     172.16.10.0/26 [110/3] via 172.16.5.3, 01:52:58, GigabitEthernet0/0/1
S        172.16.10.4/32 [1/0] via 172.16.4.2
```

Note that it is normal for a host route to overlap with some other network route as seen here. A router always attempts to place the best route for each prefix (subnet) into the IP routing table. However, the host route uses a mask of 255.255.255.255 (/32), and the dynamic route for the subnet that address resides in uses a different mask. Router R1 does not view those routes as routes for the same subnet, but as different routes, and places both into its routing table.

Once in the routing table, when a packet arrives for that one special host address (172.16.10.4), R1 uses the host route. Why? Routers use the most specific route (that is, the route with the longest prefix length) when the packet's destination address matches multiple routes in the routing table. So, a packet sent to 172.16.10.4 would match the host route because a /32 mask is longer and more specific than the route with a /26 mask. Router R1 will forward that packet to next-hop router 172.16.4.2 per the host route. Packets sent to other destinations in subnet 172.16.10.0/26 would be sent to next-hop router 172.16.5.3, because those match only one route.

Note that the section "IP Forwarding with the Longest Prefix Match" near the end of Chapter 25, "Fundamentals of IP Version 6," gets into more detail about how a router chooses a route when multiple routes match a packet destination.

Floating Static Routes

Floating static routes exist in the configuration of a router, but the router floats the route into the routing table and out of the routing table based on certain conditions. Why? As part of a broader configuration that lets the router create a temporary WAN link when the primary WAN link fails.

To see how that works, consider the example illustrated in Figure 17-16, which shows routers R1 and R2 from the previous few figures. The Ethernet WAN link serves as the only working WAN link most of the time, but now both routers have a cellular interface. The routers can connect to each other using a 4G/LTE/5G network when the primary link goes down.

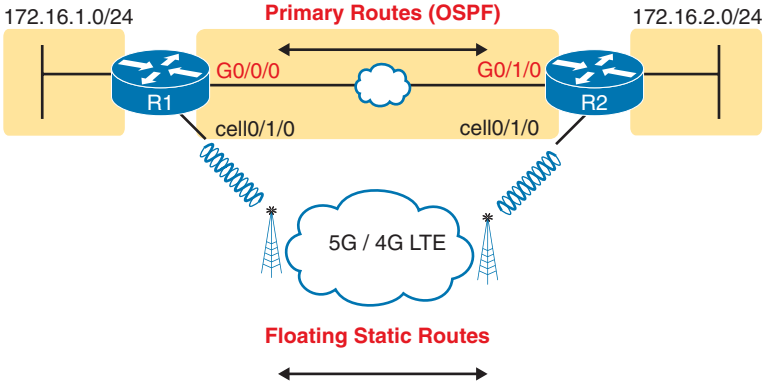

Figure 17-16 *Using a Floating Static Route to Key Subnet 172.16.2.0/24*

Interestingly, the configuration to use the cellular network to back up the Ethernet WAN link hinges on the routes. Routers use a routing protocol like OSPF on the permanent links, so in this case, R1 learns an OSPF route to subnet 172.16.2.0/24 that uses the Ethernet WAN link. R1 also defines a floating static route for that same subnet that directs packets over the cellular interface to Router R2. Although configured, Router R1 then chooses to add that floating static route to the routing table *only when the OSPF route is not available*. If the Ethernet link is up, R1 routes packets over that link. If not, it routes packets over the cellular network.

To create a floating static route, you configure a route but assign it a higher (worse) administrative distance than the competing dynamically learned route. When a router learns of more than one route to the same subnet/mask combination, the router must first decide which routing source has the better **administrative distance**, with lower being better, and then use the route learned from the better source. Default settings make routers treat static routes as better than any routes learned by a dynamic routing protocol. A floating static route reverses that logic.

To implement a floating static route, you need to use a parameter on the **ip route** command that sets the administrative distance for just that route, making the value larger than the

default administrative distance of the routing protocol. For example, the **ip route 172.16.2.0 255.255.255.0 cell0/1/0 130** command on R1 (in Figure 17-15) would do exactly that—setting the static route's administrative distance to 130. As long as the primary link stays up, and OSPF on R1 learns a route for 172.16.2.0/24, with a default OSPF administrative distance of 110, R1 ignores the floating static route. When the Ethernet WAN link fails, R1 loses its OSPF-learned route for that subnet, so R1 places the floating static route into the routing table. (The rest of the configuration, not shown here, tells the router how to make the call to create the backup link over the cellular network.)

Finally, note that while the **show ip route** command lists the administrative distance of most routes, as the first of two numbers inside two brackets, the **show ip route** *subnet* command plainly lists the administrative distance. Example 17-9 shows a sample, matching this most recent example.

Example 17-9 *Displaying the Administrative Distance of the Static Route*

```
R1# show ip route static
! Legend omitted for brevity
       172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
S         172.16.2.0/24 is directly connected, Cellular0/1/0

R1# show ip route 172.16.2.0
Routing entry for 172.16.2.0/24
  Known via "static", distance 130, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Cellular0/1/0
      Route metric is 0, traffic share count is 1
```

Troubleshooting Static Routes

These final few pages about IPv4 static routes examine some issues that can occur with static routes, both reviewing some reasons mentioned over the last few pages, while adding more detail. This topic breaks static route troubleshooting into three perspectives:

- The route is in the routing table but is incorrect.
- The route is not in the routing table.
- The route is in the routing table and is correct, but the packets do not arrive at the destination host.

Incorrect Static Routes That Appear in the IP Routing Table

This first troubleshooting item can be obvious, but it is worth pausing to think about. A static route is only as good as the input typed into the **ip route** command. IOS checks the syntax, and as mentioned earlier, makes a few other checks that this section reviews in the next heading. But once those checks are passed, IOS puts the route into the IP routing table, even if the route had poorly chosen parameters.

For instance, if you wanted to configure a static route for subnet 172.16.2.0/24, but configured the route for subnet 172.16.222.0/24, the router will accept the command—but it is for the wrong subnet. Or, when choosing the next-hop router address, you looked at a network diagram, and used the next-hop address of a router to the left—but the destination subnet

was on the right side of the figure. Or you could look at documentation and choose the next-hop address for the command, and it is in the correct destination subnet—but it is not the address of the next-hop router. In all these cases, the router would accept the command and even add the route to the routing table, but the route would not be useful.

As another example, IOS will reject the **ip route** command if the first two parameters reveal the configuration of an IP address within the subnet rather than the subnet ID. For instance, back in Example 17-5, the commands used mask 255.255.255.0, with subnet IDs 172.16.2.0 and 172.16.3.0. Example 17-10 shows a repeat of those same commands, but now with addresses from within those subnets: 172.16.2.1 and 172.16.3.1. In both cases, IOS shows an error message and does not add the command to the configuration.

Example 17-10 *IOS Rejects* **ip route** *Commands That Use an Address in the Subnet*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1-8200(config)# ip route 172.16.2.1 255.255.255.0 G0/0/0
%Inconsistent address and mask
R1-8200(config)# ip route 172.16.3.1 255.255.255.0 172.16.5.3
%Inconsistent address and mask
R1-8200(config)#
```

When you see an exam question that has static routes, and you see them in the output of **show ip route**, remember to check on these items:

- Are the subnet ID and mask correct?
- Is the next-hop IP address correct and referencing an IP address on a neighboring router?
- Does the next-hop IP address identify the correct router?
- Is the outgoing interface correct and referencing an interface on the local router (that is, the same router where the static route is configured)?

The Static Route Does Not Appear in the IP Routing Table

After configuring an **ip route** command, IOS might or might not add the route to the IP routing table. IOS also considers the following before adding the route to its routing table:

- For **ip route** commands that list an outgoing interface, that interface must be in an up/up state.
- For **ip route** commands that list a next-hop IP address, the local router must have a route to reach that next-hop address.

For example, earlier in Example 17-5, R1's command **ip route 172.16.3.0 255.255.255.0 172.16.5.3** defines a static route. Before adding the route to the IP routing table, R1 looks for an existing IP route to reach 172.16.5.3. In that case, R1 will find a connected route for subnet 172.16.5.0/24 as long as its Ethernet WAN link is up. As a result, R1 adds the static route to subnet 172.16.3.0/24. Later, if R1's G0/0/1 were to fail, R1 would remove its connected route to 172.16.5.0/24 from the IP routing table—an action that would also then cause R1 to remove its static route to 172.16.3.0/24.

You can configure a static route so that IOS ignores these basic checks, always putting the IP route in the routing table. To do so, just use the **permanent** keyword on the **ip route** command. For example, if you add the **permanent** keyword to the end of the two commands as demonstrated in Example 17-11, R1 would now add these routes, regardless of whether the two WAN links were up.

Example 17-11 *Permanently Adding Static Routes to the IP Routing Table (Router R1)*

```
ip route 172.16.2.0 255.255.255.0 G0/0/0 permanent
ip route 172.16.3.0 255.255.255.0 172.16.5.3 permanent
```

Note that although the **permanent** keyword lets the router keep the route in the routing table without checking the outgoing interface or route to the next-hop address, it does not magically fix a broken route. For example, if the outgoing interface fails, the route will remain in the routing table, but the router cannot forward packets because the outgoing interface is down.

The Correct Static Route Appears but Works Poorly

This last section is a place to make two points—one mainstream and one point to review a bit of trivia.

First, on the mainstream point, the static route can be perfect, but the packets might not arrive because of other problems. An incorrect static route is just one of many items to check when you're troubleshooting problems like "host A cannot connect to server B." The root cause may be the static route, or it may be something else. Chapter 20, "Troubleshooting IPv4 Routing," goes into some depth about troubleshooting these types of problems.

On the more specific point, be wary of any **ip route** command with the **permanent** keyword. IOS puts these routes in the routing table with no checks for accuracy. You should check whether the outgoing interface is down and/or whether the router has a route to reach the next-hop address.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 17-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 17-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 17-3 Key Topics for Chapter 17

Key Topic Element	Description	Page Number
List	Steps taken by a host when forwarding IP packets	429
List	Steps taken by a router when forwarding IP packets	430
Figure 17-2	Diagram of five routing steps taken by a router	430
Figure 17-7	Breakdown of IP routing table with matching and forwarding details	434
List	Three common sources from which routers build IP routes	435
List	Rules regarding when a router creates a connected route	436
Figure 17-13	Static route configuration concept	441
List	Troubleshooting checklist for routes that do appear in the IP routing table	449
List	Troubleshooting checklist for static routes that do not appear in the IP routing table	449

Key Terms You Should Know

administrative distance, ARP table, connected route, default route, floating static route, host route, network route, next-hop router, outgoing interface, routing table, static route

Command References

Tables 17-4 and 17-5 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 17-4 Chapter 17 Configuration Command Reference

Command	Description
ip address *ip-address mask*	Interface subcommand that assigns the interface's IP address
interface *type number.subint*	Global command to create a subinterface and to enter configuration mode for that subinterface
[**no**] **ip routing**	Global command that enables (**ip routing**) or disables (**no ip routing**) the routing of IPv4 packets on a router or Layer 3 switch
ip route *prefix mask* {*ip-address* | *interface-type interface-number*} [*distance*] [**permanent**]	Global configuration command that creates a static route

Table 17-5 Chapter 17 EXEC Command Reference

Command	Description
show ip route	Lists the router's entire routing table
show ip route [connected | static | ospf]	Lists a subset of the IP routing table
show ip route *ip-address*	Lists detailed information about the route that a router matches for the listed IP address
show arp, show ip arp	Lists the router's IPv4 ARP table
clear ip arp [ip-address]	Removes all dynamically learned ARP table entries, or if the command lists an IP address, removes the entry for that IP address only

This page intentionally left blank

CHAPTER 18

IP Routing in the LAN

This chapter covers the following exam topics:

- **1.0 Network Fundamentals**
  - **1.1 Explain the role and function of network components**
    - **1.1.a Routers**
    - **1.1.b Layer 2 and Layer 3 switches**
  - **1.6 Configure and verify IPv4 addressing and subnetting**
- **2.0 Network Access**
  - **2.1 Configure and verify VLANs (normal range) spanning multiple switches**
    - **2.1.c InterVLAN connectivity**
  - **2.4 Configure and verify (Layer 2/Layer 3) EtherChannel (LACP)**

The preceding two chapters showed how to configure an IP address and mask on a router interface, making the router ready to route packets to/from the subnet implied by that address/mask combination. While true and useful, all the examples so far ignored the LAN switches and the possibility of VLANs. In fact, the examples so far show the simplest possible cases: the attached switches as Layer 2 switches, using only one VLAN, with the router configured with one **ip address** command on its physical interface. This chapter takes a detailed look at how to configure routers so that they route packets to/from the subnets that exist on each and every VLAN.

Because Layer 2 switches do not forward Layer 2 frames between VLANs, a network must use a device that performs IP routing to route IP packets between subnets. That device can be a router, or it can be a switch that also includes routing features.

To review, Ethernet defines the concept of a VLAN, while IP defines the concept of an IP subnet, so a VLAN is not equivalent to a subnet. However, the devices in one VLAN typically use IP addresses in one subnet. By the same reasoning, devices in two different VLANs are normally in two different subnets. For two devices in different VLANs to communicate with each other, routers must connect to the subnets that exist on each VLAN, and then the routers forward IP packets between the devices in those subnets.

This chapter discusses multiple methods of routing between VLANs, all of which can be useful in different scenarios:

- **VLAN Routing with Router 802.1Q Trunks:** The first section discusses configuring a router to use VLAN trunking on a link connected to a Layer 2 switch. The router does the routing, with the switch creating the VLANs. The link between the router and switch uses VLAN trunking so that the router has an interface connected to each

VLAN/subnet. This feature is known as routing over a VLAN trunk and also known as **router-on-a-stick (ROAS)**.

- **VLAN Routing with Layer 3 Switch SVIs:** The second section discusses using a LAN switch that supports both Layer 2 switching and Layer 3 routing (called a **Layer 3 switch** or **multilayer switch**). To route, the Layer 3 switch configuration uses interfaces called switched virtual interfaces (SVIs), which are also called VLAN interfaces.
- **VLAN Routing with Layer 3 Switch Routed Ports:** The third major section of the chapter discusses an alternative to SVIs called **routed ports**, in which the physical switch ports are made to act like interfaces on a router. This third section also introduces the concept of an EtherChannel as used as a routed port in a feature called **Layer 3 EtherChannel (L3 EtherChannel)**.
- **VLAN Routing on a Router's LAN Switch Ports:** The final major section discusses Cisco routers that include integrated LAN switch ports. For instance, rather than installing a small router plus a separate small switch at a branch office, you could install a router that has a small set of integrated switch ports. This section shows how to configure the router's switch ports and the internal logic, so the router routes packets for the subnets on those switch ports.

"Do I Know This Already?" Quiz

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

Table 18-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
VLAN Routing with Router 802.1Q Trunks	1, 2
VLAN Routing with Layer 3 Switch SVIs	3, 4
VLAN Routing with Layer 3 Switch Routed Ports	5, 6
VLAN Routing on a Router's LAN Switch Ports	7

1. Router 1 has a Fast Ethernet interface 0/0 with IP address 10.1.1.1. The interface is connected to a switch. This connection is then migrated to use 802.1Q trunking. Which of the following commands could be part of a valid configuration for Router 1's Fa0/0 interface, assuming the trunk uses VLAN 4 as one of the supported VLANs? (Choose two answers.)
   a. interface fastethernet 0/0.4
   b. dot1q enable
   c. dot1q enable 4
   d. trunking enable
   e. trunking enable 4
   f. encapsulation dot1q 4

2. Router R1 has a router-on-a-stick (ROAS) configuration with two subinterfaces of interface G0/1: G0/1.1 and G0/1.2. Physical interface G0/1 is currently in a down/down state. The network engineer then configures a **shutdown** command when in interface configuration mode for G0/1.1 and a **no shutdown** command when in interface configuration mode for G0/1.2. Which answers are correct about the interface state for the subinterfaces? (Choose two answers.)
   a. G0/1.1 will be in a down/down state.
   b. G0/1.2 will be in a down/down state.
   c. G0/1.1 will be in an administratively down state.
   d. G0/1.2 will be in an up/up state.
3. A Layer 3 switch has been configured to route IP packets between VLANs 1, 2, and 3 using SVIs, which connect to subnets 172.20.1.0/25, 172.20.2.0/25, and 172.20.3.0/25, respectively. The engineer issues a **show ip route connected** command on the Layer 3 switch, listing the connected routes. Which of the following answers lists a piece of information that should be in at least one of the routes?
   a. Interface Gigabit Ethernet 0/0.3
   b. Next-hop router 172.20.2.1
   c. Interface VLAN 2
   d. Mask 255.255.255.0
4. An engineer has successfully configured a Layer 3 switch with SVIs for VLANs 2 and 3 with autostate enabled by default. Hosts in the subnets using VLANs 2 and 3 can ping each other with the Layer 3 switch routing the packets. The next week, the network engineer receives a call that those same users can no longer ping each other. If the problem is with the Layer 3 switching function, which of the following could have caused the problem? (Choose two answers.)
   a. Six (or more) out of ten working VLAN 2 access ports failing due to physical problems
   b. A **shutdown** command issued from interface VLAN 4 configuration mode
   c. A **no vlan 2** command issued from global configuration mode
   d. A **shutdown** command issued from VLAN 2 configuration mode
5. A LAN design uses a Layer 3 EtherChannel between two switches SW1 and SW2, with port-channel interface 1 used on both switches. SW1 uses ports G0/1, G0/2, G0/3, and G0/4 in the channel. Which of the following is true about SW1's configuration to enable the channel to route IPv4 packets correctly? (Choose two answers.)
   a. The **ip address** command must be on the port-channel 1 interface.
   b. The **ip address** command must be on interface G0/1 (lowest numbered port).
   c. The port-channel 1 interface must be configured with the **no switchport** command.
   d. Interface G0/1 must be configured with the **routedport** command.

6. A LAN design uses a Layer 3 EtherChannel between two switches: SW1 and SW2. An engineer adds SW1 port G0/1 to the channel successfully but later fails when adding SW1 port G0/2 to the channel. Which answers list a configuration setting on port G0/2 that would cause this issue? (Choose two answers.)
   a. A different STP cost (**spanning-tree cost** *value*)
   b. A different speed (**speed** *value*)
   c. A default setting for switchport (**switchport**)
   d. A different access VLAN (**switchport access vlan** *vlan-id*)
7. A router has some routed and some switched physical ports. Which interface subcommands would you expect to be supported only on the switched ports?
   a. The **switchport access vlan** *vlan-id* subcommand
   b. The **ip address** *address mask* subcommand
   c. The **description** *text* subcommand
   d. The **hostname** *name* subcommand

Foundation Topics

VLAN Routing with Router 802.1Q Trunks

Almost all enterprise networks use VLANs. To route IP packets in and out of those VLANs, some devices (either routers or Layer 3 switches) need to have an IP address in each subnet and have a connected route to each of those subnets. Then the IP addresses on those routers or Layer 3 switches can serve as the default gateways in those subnets.

This chapter breaks down the LAN routing options into five categories:

- Use a router, with one router LAN interface and cable connected to the switch for each and every VLAN (typically not used)
- Use a router, with a VLAN trunk connecting to a LAN switch (known as router-on-a-stick, or ROAS)
- Use a Layer 3 switch with switched virtual interfaces (SVIs)
- Use a Layer 3 switch with routed interfaces (which may or may not be Layer 3 EtherChannels)
- Use a router with integrated switch ports, configuring it much like a Layer 3 switch with SVIs

Of the items in the list, the first option works, but to be practical, it requires far too many interfaces. It is mentioned here only to make the list complete.

As for the other options, this chapter discusses each in turn. Real networks use these features, with the choice to use one or the other driven by the design and needs for a particular part of the network. Figure 18-1 shows cases in which these options could be used.

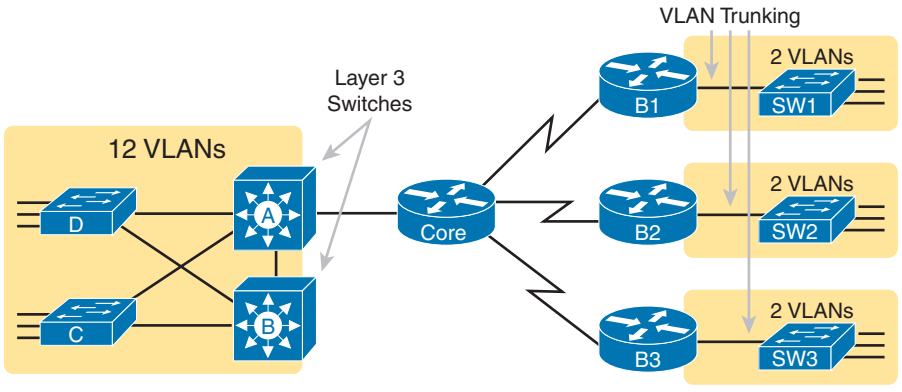

Figure 18-1 *Layer 3 Switching at the Central Site, ROAS at Branch Offices*

Figure 18-1 shows two switches, labeled A and B, which could act as Layer 3 switches—both with SVIs and routed interfaces. The figure shows a central site campus LAN on the left, with 12 VLANs. At the central site, two of the switches act as Layer 3 switches, combining the functions of a router and a switch, routing between all 12 subnets/VLANs. Those Layer 3 switches could use SVIs, routed interfaces, or both.

Figure 18-1 also shows a classic case for using a router with a VLAN trunk. Sites like the remote sites on the right side of the figure may have a WAN-connected router and a LAN switch. These sites might use ROAS to take advantage of the router's ability to route over an 802.1Q trunk.

Note that Figure 18-1 just shows an example. The engineer could use Layer 3 switching at each site or routers with VLAN trunking at each site.

Configuring ROAS

This next topic discusses how routers route packets to subnets associated with VLANs connected to a router 802.1Q trunk. That long description can be a bit of a chore to repeat each time someone wants to discuss this feature, so over time, the networking world has instead settled on a shorter and more interesting name for this feature: router-on-a-stick (ROAS).

ROAS uses router VLAN trunking configuration to give the router a logical interface connected to each VLAN. Because the router then has an interface connected to each VLAN, the router can also be configured with an IP address in the subnet that exists on each VLAN.

The router needs to have an IP address/mask associated with each VLAN on the trunk. However, the router has only one physical interface for the link connected to the trunk. Cisco solves this problem by creating multiple virtual router interfaces, one for each supported VLAN on that trunk. Cisco calls these virtual interfaces **subinterfaces**, and the router configuration includes an **ip address** command for each subinterface.

Figure 18-2 shows the concept with Router B1, one of the branch routers from Figure 18-1. Because this router needs to route between only two VLANs, the figure also shows two

---

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

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

subinterfaces, named G0/0/0.10 and G0/0/0.20, which create a new place in the configuration where the per-VLAN configuration settings can be made. The router treats frames tagged with VLAN 10 as if they came in or out of G0/0/0.10 and frames tagged with VLAN 20 as if they came in or out G0/0/0.20.

Key Topic

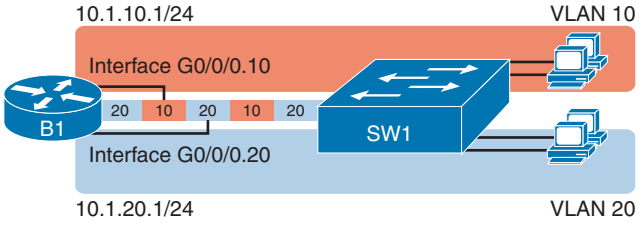

Figure 18-2 *Subinterfaces on Router B1*

In addition, note that most Cisco routers do not attempt to negotiate trunking, so both the router and switch need to manually configure trunking. This chapter discusses the router side of that trunking configuration; the matching switch interface would need to be configured with the **switchport mode trunk** command.

Example 18-1 shows a full example of the 802.1Q trunking configuration required on Router B1 in Figure 18-2. More generally, these steps detail how to configure 802.1Q trunking on a router:

Config Checklist

Step 1. Use the **interface** *type number.subint* command in global configuration mode to create a unique subinterface for each VLAN that needs to be routed.

Step 2. Use the **encapsulation dot1q** *vlan_id* command in subinterface configuration mode to enable 802.1Q and associate one specific VLAN with the subinterface.

Step 3. Use the **ip address** *address mask* command in subinterface configuration mode to configure IP settings (address and mask).

18

Example 18-1 *Router 802.1Q Configuration per Figure 18-2*

```
B1# show running-config
! Only pertinent lines shown
interface gigabitethernet 0/0/0
! No IP address or encapsulation up here!
!
interface gigabitethernet 0/0/0.10
 encapsulation dot1q 10
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

First, look at the subinterface numbers. The subinterface number begins with the period, like .10 and .20 in this case. These numbers can be any number from 1 up through a very

large number (over 4 billion). The number just needs to be unique among all subinterfaces associated with this one physical interface. In fact, the subinterface number does not even have to match the associated VLAN ID. (The **encapsulation** command, and not the subinterface number, defines the VLAN ID associated with the subinterface.)

NOTE Although not required, most sites do choose to make the subinterface number match the VLAN ID, as shown in Example 18-1, just to avoid confusion.

Each subinterface configuration lists two subcommands. One command (**encapsulation**) enables trunking and defines the VLAN the router associates with the frames entering and exiting the subinterface. The **ip address** command works the same way it does on any other router interface.

Now that the router has a working interface, with IPv4 addresses configured, the router can route IPv4 packets on these subinterfaces. That is, the router treats these subinterfaces like any physical interface in terms of adding connected routes, matching those routes, and forwarding packets to/from those connected subnets.

Example 18-1 uses two VLANs, 10 and 20, neither of which is the native VLAN on the trunk. ROAS can make use of the native VLAN, with two variations in the configuration, so it requires a little extra thought. The native VLAN can be configured on a subinterface, or on the physical interface, or ignored as in Example 18-1. Each 802.1Q trunk has one native VLAN, and if the router needs to route packets for a subnet that exists in the native VLAN, then the router needs some configuration to support that subnet. The two options to define a router interface for the native VLAN are

- Configure the **ip address** command on the physical interface, but without an **encapsulation** command; the router considers this physical interface to be using the native VLAN.
- Configure the **ip address** command on a subinterface and use the **encapsulation dot1q** *vlan-id* **native** subcommand to tell the router both the VLAN ID and the fact that it is the native VLAN.

Example 18-2 shows both native VLAN configuration options with a small change to the same configuration in Example 18-1. In this case, VLAN 10 becomes the native VLAN. The top part of the example shows the option to configure the router physical interface to use native VLAN 10. The second half of the example shows how to configure that same native VLAN on a subinterface. In both cases, the switch configuration also needs to be changed to make VLAN 10 the native VLAN.

Example 18-2 *Two Configuration Options for Native VLAN 10 on Router B1*

```
! First option: put the native VLAN IP address on the physical interface
interface gigabitethernet 0/0/0
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

```
! Second option: like Example 18-1, but add the native keyword
interface gigabitethernet 0/0/0.10
 encapsulation dot1q 10 native
 ip address 10.1.10.1 255.255.255.0
!
interface gigabitethernet 0/0/0.20
 encapsulation dot1q 20
 ip address 10.1.20.1 255.255.255.0
```

Verifying ROAS

Beyond using the **show running-config** command, ROAS configuration on a router can be best verified with two commands: **show ip route** [**connected**] and **show vlans**. As with any router interface, as long as the interface is in an up/up state and has an IPv4 address configured, IOS will put a connected (and local) route in the IPv4 routing table. So, a first and obvious check would be to see if all the expected connected routes exist. Example 18-3 lists the connected routes per the configuration shown in Example 18-1.

Example 18-3 *Connected Routes Based on Example 18-1 Configuration*

```
B1# show ip route connected
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! Legend omitted for brevity

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.10.0/24 is directly connected, GigabitEthernet0/0/0.10
L        10.1.10.1/32 is directly connected, GigabitEthernet0/0/0.10
C        10.1.20.0/24 is directly connected, GigabitEthernet0/0/0.20
L        10.1.20.1/32 is directly connected, GigabitEthernet0/0/0.20
```


As for interface and subinterface state, note that the ROAS subinterface state does depend to some degree on the physical interface state. In particular, the subinterface state cannot be better than the state of the matching physical interface. For instance, so far, the physical interface and related subinterfaces on Router B1 rested in an up/up state. If you unplugged the cable from that physical port, the physical and subinterfaces all fail to a down/down state. Or, if you shut down the physical interface, the physical and subinterfaces move to an administratively down state, as seen in Example 18-4.

Example 18-4 *Subinterface State Tied to Physical Interface State*

```
B1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
B1(config)# interface g0/0/0
B1(config-if)# shutdown
B1(config-if)# ^Z
B1# show ip interface brief | include 0/0/0
GigabitEthernet0/0/0         unassigned      YES manual administratively down down
GigabitEthernet0/0/0.10      10.1.10.1       YES manual administratively down down
GigabitEthernet0/0/0.20      10.1.20.1       YES manual administratively down down
```

Additionally, the subinterface state can also be enabled and disabled independently from the physical interface, using the **no shutdown** and **shutdown** commands in subinterface configuration mode. For instance, the physical interface and subinterface .10 can remain up/up, while subinterface .20 can be independently shut down.

Another useful ROAS verification command, **show vlans**, spells out which router trunk interfaces use which VLANs, which VLAN is the native VLAN, plus some packet statistics. The fact that the packet counters are increasing can be useful when verifying whether traffic is happening or not. Example 18-5 shows a sample, based on the Router B1 configuration in Example 18-2 (bottom half), in which native VLAN 10 is configured on subinterface G0/0/0.10. Note that the output identifies VLAN 1 associated with the physical interface, VLAN 10 as the native VLAN associated with G0/0/0.10, and VLAN 20 associated with G0/0/0.20. It also lists the IP addresses assigned to each interface/subinterface.

Example 18-5 *Router* **show vlans** *Command Reveals Router ROAS Configuration*

```
B1# show vlans
VLAN ID: 1 (IEEE 802.1Q Encapsulation)

   Protocols Configured:          Received:        Transmitted:

VLAN trunk interfaces for VLAN ID 1:
GigabitEthernet0/0/0

GigabitEthernet0/0/0 (1)
      Total 5 packets, 330 bytes input
      Total 20 packets, 3134 bytes output

VLAN ID: 10 (IEEE 802.1Q Encapsulation)

 This is configured as native Vlan for the following interface(s) :
GigabitEthernet0/0/0    Native-vlan Tx-type: Untagged

   Protocols Configured:          Received:        Transmitted:
                      IP                  0                    0

VLAN trunk interfaces for VLAN ID 10:
GigabitEthernet0/0/0.10

GigabitEthernet0/0/0.10 (10)
                      IP: 10.1.10.1
      Total 38 packets, 5696 bytes input
      Total 2 packets, 128 bytes output

VLAN ID: 20 (IEEE 802.1Q Encapsulation)
```

```
   Protocols Configured:          Received:        Transmitted:
                        IP                   0                   0

VLAN trunk interfaces for VLAN ID 20:
GigabitEthernet0/0/0.20

GigabitEthernet0/0/0.20 (20)
                        IP: 10.1.20.1
      Total 0 packets, 0 bytes input
      Total 2 packets, 128 bytes output
```

Troubleshooting ROAS

The biggest challenge when troubleshooting ROAS has to do with the fact that if you misconfigure only the router or misconfigure only the switch, the other device on the trunk has no way to know that the other side is misconfigured. That is, the router configuration might be correct, but routing might still fail because of problems on the attached switch. So, troubleshooting ROAS often begins with checking the configuration on both the router and switch because there is no status output on either device that tells you where the problem might be.

First, to check ROAS on the router, you need to start with the intended configuration and ask questions about the configuration:

Key Topic

1. Is each non-native VLAN configured on the router with an **encapsulation dot1q** *vlan-id* command on a subinterface?
2. Do those same VLANs exist on the trunk on the neighboring switch (**show interfaces trunk**), and are they in the allowed list, not VTP pruned, and not STP blocked?
3. Does each router ROAS subinterface have an IP address/mask configured per the planned configuration?
4. If using the native VLAN, is it configured correctly on the router either on a subinterface (with an **encapsulation dot1q** *vlan-id* **native** command) or implied on the physical interface?
5. Is the same native VLAN configured on the neighboring switch's trunk in comparison to the native VLAN configured on the router?
6. Are the router physical or ROAS subinterfaces configured with a **shutdown** command?

For some of these steps, you need to be ready to investigate possible VLAN trunking issues on the LAN switch. Many Cisco router interfaces do not negotiate trunking. As a result, ROAS relies on static trunk configuration on both the router and switch. If the switch has any problems with VLANs or the VLAN trunking configuration on its side of the trunk, the router has no way to realize that the problem exists.

For example, imagine you configured ROAS on a router just like in Example 18-1 or Example 18-2. However, the switch on the other end of the link had no matching configuration. For instance, maybe the switch did not even define VLANs 10 and 20. Maybe the switch did not configure trunking on the port connected to the router. Even with blatant

misconfiguration or missing configuration on the switch, the router still shows up/up ROAS interfaces and subinterfaces, IP routes in the output of **show ip route**, and meaningful configuration information in the output of the **show vlans** command. The router will forward packets (encapsulated inside frames) to the switch, but the switch's configuration does not give it enough information to forward the frame correctly.

VLAN Routing with Layer 3 Switch SVIs

Using a router with ROAS to route packets makes sense in some cases, particularly at small remote sites. In sites with a larger LAN, network designers choose to use Layer 3 switches for most inter-VLAN routing.

A Layer 3 switch (also called a multilayer switch) is one device, but it executes logic at two layers: Layer 2 LAN switching and Layer 3 IP routing. The Layer 2 switch function forwards frames inside each VLAN, but it will not forward frames between VLANs. The Layer 3 forwarding (routing) logic forwards IP packets between VLANs by applying IP routing logic to IP packets sent by the devices in those VLANs.

Layer 3 switches typically support two configuration options to enable IPv4 routing inside the switch, specifically to enable IPv4 on switch interfaces. This section explains one option, an option that uses switched virtual interfaces (SVI). The following major section of the chapter deals with the other option for configuring IPv4 addresses on Layer 3 switches: routed interfaces.

Configuring Routing Using Switch SVIs

The configuration of a Layer 3 switch mostly looks like the Layer 2 switching configuration shown back in Parts II and III of this book, with a small bit of configuration added for the Layer 3 functions. The Layer 3 switching function needs a virtual interface connected to each VLAN internal to the switch. These **VLAN interfaces** act like router interfaces, with an IP address and mask. The Layer 3 switch has an IP routing table, with connected routes off each of these VLAN interfaces. Cisco refers to these virtual interfaces as **switched virtual interfaces (SVIs)**.

To show the concept of Layer 3 switching with SVIs, the following example uses the same branch office with two VLANs shown in the earlier examples, but now the design will use Layer 3 switching in the LAN switch. Figure 18-3 details the physical connections, subnets, and VLAN IDs. Figure 18-4 then shows the internal routing logic and SVIs.

Key Topic

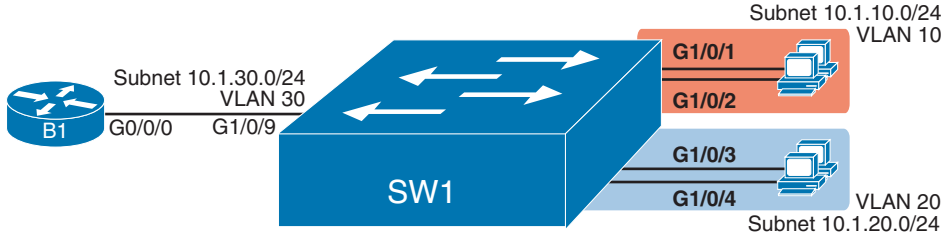

Figure 18-3 *Physical Interfaces and Subnets for Layer 3 Switching Example*

Key Topic

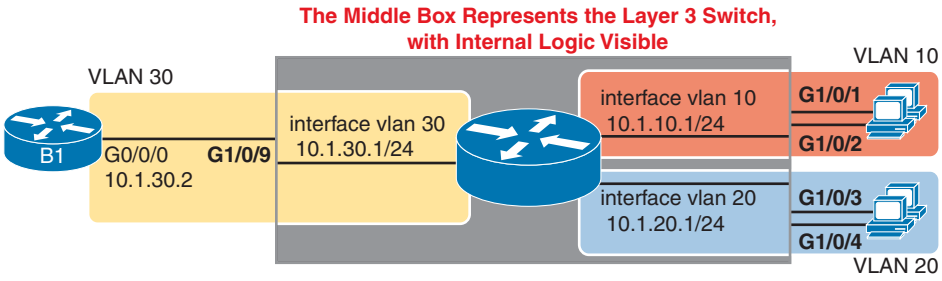

Figure 18-4 *Internal VLAN Interfaces for Layer 3 Switching Example*

Note that Figure 18-4 represents the internals of the Layer 3 switch within the box in the middle of the figure. The branch still has two user VLANs (10 and 20), so the Layer 3 switch needs one VLAN interface (SVI) for each VLAN. The figure shows a router icon inside the gray box to represent the Layer 3 switching (routing) function, with two VLAN interfaces on the right side of that icon. In addition, the traffic still needs to get to Router B1 (a physical router) to access the WAN, so the switch uses a third VLAN (VLAN 30 in this case) for the link to Router B1. The switch treats the link between the Layer 3 switch and Router B1 as an access link; the router would be unaware of any VLANs and would not need VLAN trunking configuration.

The following steps show how to configure Layer 3 switching using SVIs.

Config Checklist

Step 1. Enable IP routing on the switch, as needed:

- **a.** (As needed) Use a model-specific command to change the switch forwarding ASIC settings to make space for IPv4 routes and reload the switch to make those settings take effect.
- **c.** Use the **ip routing** command in global configuration mode to enable the IPv4 routing function in IOS software and to enable key commands like **show ip route**.

Step 2. Configure each SVI interface, one per VLAN for which routing should be done by this Layer 3 switch:

- **a.** Use the **interface vlan** *vlan_id* command in global configuration mode to create a VLAN interface and to give the switch's routing logic a Layer 3 interface connected into the VLAN of the same number.
- **b.** Use the **ip address** *address mask* command in VLAN interface configuration mode to configure an IP address and mask on the VLAN interface, enabling IPv4 routing on that VLAN interface.
- **c.** (As needed) Use the **no shutdown** command in interface configuration mode to enable the VLAN interface (if it is currently in a shutdown state).

NOTE Regarding Step 1A, some older switch models do not support IP routing until you reprogram the switch's forwarding ASIC. For instance, the 2960 and 2960-XR switches popularly sold by Cisco in the 2010s required the **sdm prefer** global command followed by a reload of the switch. Newer Cisco switch models default to support IP routing and do not require similar steps.

Example 18-6 shows the configuration to match Figure 18-4. In this case, the switch defaulted to support IP routing in the forwarding ASIC without any special commands. The example shows the related configuration on all three VLAN interfaces.

Example 18-6 *VLAN Interface Configuration for Layer 3 Switching*

```
ip routing
!
interface vlan 10
 ip address 10.1.10.1 255.255.255.0
!
interface vlan 20
 ip address 10.1.20.1 255.255.255.0
!
interface vlan 30
 ip address 10.1.30.1 255.255.255.0
```

Verifying Routing with SVIs

With the VLAN configuration shown in the previous section, the switch is ready to route packets between the VLANs as shown in Figure 18-4. To support the routing of packets, the switch adds connected IP routes as shown in Example 18-7; note that each route is listed as being connected to a different VLAN interface.

Example 18-7 *Connected Routes on a Layer 3 Switch*

```
SW1# show ip route connected
! legend omitted for brevity

       10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C         10.1.10.0/24 is directly connected, Vlan10
L         10.1.10.1/32 is directly connected, Vlan10
C         10.1.20.0/24 is directly connected, Vlan20
L         10.1.20.1/32 is directly connected, Vlan20
C         10.1.30.0/24 is directly connected, Vlan30
L         10.1.30.1/32 is directly connected, Vlan30
```

The switch would also need additional routes to the rest of the network (not shown in the figures in this chapter). The Layer 3 switch could use static routes or a routing protocol, depending on the capabilities of the switch. For instance, if you then enabled OSPF on the Layer 3 switch, the configuration and verification would work the same as it does on a router, as discussed in Part VI, "OSPF." The routes that IOS adds to the Layer 3 switch's IP routing table list the VLAN interfaces as outgoing interfaces.

NOTE Some models of Cisco enterprise switches, based on the specific model, IOS version, and IOS feature set, support different capabilities for IP routing and routing protocols. For use in real networks, check the capabilities of the switch model by using the Cisco Feature Navigator (CFN) tool at www.cisco.com/go/cfn.

Troubleshooting Routing with SVIs

On a physical router, when using physical interfaces, the router can assign a state to the interface based on physical factors, like whether the cable is installed or if the router receives any electricity or light over the cable. However, SVIs act as a virtual interface for the routing function in a Layer 3 switch. The switch must use some other logic to decide whether to place the SVI into a working (up/up) state.

Layer 3 switches use one of two interface configuration settings to dictate the logic to determine interface state: either **autostate** or **no autostate**. The default setting on a VLAN interface, **autostate**, checks several factors about the underlying VLAN to determine the state of the VLAN interface. By configuring the **no autostate** VLAN interface subcommand, the switch instead uses much simpler logic with fewer checks. The following few pages explain both.

SVI Interface State with Autostate Enabled

The VLAN interface acts as the interface between the switch's routing function and the VLAN. For that VLAN interface to work properly, the VLAN must work properly. In particular, for a VLAN interface to be in an up/up state when using the autostate setting:

Key Topic

Step 1. The VLAN must be defined on the local switch (either explicitly or learned with VTP).

Step 2. The switch must have at least one up/up interface using the VLAN, including:

- **a.** An up/up access interface assigned to that VLAN
- **b.** A trunk interface for which the VLAN is in the allowed list, is STP forwarding, and is not VTP pruned

Step 3. The VLAN must be administratively enabled (that is, not **shutdown**).

Step 4. The VLAN interface must be administratively enabled (that is, not **shutdown**).

Do not miss this point: VLAN and the VLAN interface are related but separate ideas, each configured separately in the CLI. A VLAN interface, configured with the **interface vlan** *vlan-id* global command, creates a switch's Layer 3 interface connected to the VLAN. A VLAN, created with the **vlan** *vlan-id* global command, creates the VLAN. If you want to route packets for the subnets on VLANs 11, 12, and 13, using SVIs, you must configure the VLAN interfaces with those same VLAN IDs 11, 12, and 13. The VLANs with those same VLAN IDs must also exist.

IOS supports the function to disable and enable both a VLAN and a VLAN interface with the **shutdown** and **no shutdown** commands (as mentioned in Steps 3 and 4 in the preceding list). As part of the configuration checklist tasks, check the status to ensure that all the configuration enables all the related VLANs and VLAN interfaces.

Example 18-8 shows three scenarios, each of which leads to one of the VLAN interfaces in the previous configuration example (Figure 18-4, Example 18-6) to fail. At the beginning of the example, all three VLAN interfaces are up/up. To begin the example, VLANs 10, 20, and 30

each have at least one access interface up and working. The example works through three scenarios:

- **Scenario 1:** The last access interface in VLAN 10 is shut down (G1/0/1), so IOS shuts down the VLAN 10 interface.
- **Scenario 2:** VLAN 20 (not VLAN interface 20, but VLAN 20) is deleted, which results in IOS then bringing down (not shutting down) the VLAN 20 interface.
- **Scenario 3:** VLAN 30 (not VLAN interface 30, but VLAN 30) is shut down, which results in IOS then bringing down (not shutting down) the VLAN 30 interface.

Example 18-8 *Three Examples That Cause VLAN Interfaces to Fail*

```
SW1# show interfaces status
! Only ports related to the example are shown
Port      Name               Status       Vlan       Duplex  Speed Type
Gi1/0/1                      connected    10         a-full  a-100 10/100/1000BaseTX
Gi1/0/2                      notconnect   10           auto   auto 10/100/1000BaseTX
Gi1/0/3                      connected    20         a-full  a-100 10/100/1000BaseTX
Gi1/0/4                      connected    20         a-full  a-100 10/100/1000BaseTX
Gi1/0/9                      connected    30         a-full a-1000 10/100/1000BaseTX

SW1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.

! Case 1: Interface G1/0/1, the last up/up access interface in VLAN 10, is shutdown
SW1(config)# interface GigabitEthernet 1/0/1
SW1(config-if)# shutdown
SW1(config-if)#
*Apr 2 19:54:08.784: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan10, changed
state to down
*Apr 2 19:54:10.772: %LINK-5-CHANGED: Interface GigabitEthernet1/0/1, changed state
to administratively down
*Apr 2 19:54:11.779: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net1/0/1, changed state to down

! Case 2: VLAN 20 is deleted
SW1(config)# no vlan 20
SW1(config)#
*Apr 2 19:54:39.688: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan20, changed
state to down

! Case 3: VLAN 30, the VLAN from the switch to the router, is shutdown
SW1(config)# vlan 30
SW1(config-vlan)# shutdown
SW1(config-vlan)# exit
SW1(config)#
*Apr 2 19:55:25.204: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan30, changed
state to down
```

```
! Final status of all three VLAN interfaces is below
SW1# show ip interface brief | include Vlan
Vlan1                  unassigned      YES manual administratively down down
Vlan10                 10.1.10.1       YES manual up                    down
Vlan20                 10.1.20.1       YES manual up                    down
Vlan30                 10.1.30.1       YES manual up                    down
```

Note that the example ends with the three VLAN interfaces in an up/down state per the **show ip interface brief** command.

SVI Interface State with Autostate Disabled

With autostate disabled, the switch checks only whether the VLAN is defined on the switch, either explicitly or learned by VTP. It ignores all the other checks performed when using autostate. If no matching VLAN exists, the switch places the VLAN interface in an up/down state.

Example 18-9 shows how to determine whether autostate is enabled from the **show interfaces vlan** command output. In the example, the engineer already configured the **no autostate** command under interface VLAN 10, with interface VLAN 20 using the default setting of **autostate**.

Example 18-9 *Recognizing the Autostate Setting on VLAN Interfaces*

```
SW1# show interfaces vlan 10
Vlan10 is up, line protocol is up , Autostate Disabled
! Lines omitted for brevity
SW1# show interfaces vlan 20
Vlan10 is up, line protocol is up , Autostate Enabled
! Lines omitted for brevity
```


VLAN Routing with Layer 3 Switch Routed Ports

When Layer 3 switches use SVIs, the physical interfaces on the switches act like they always have: as Layer 2 interfaces. That is, the physical interfaces receive Ethernet frames, the switch learns the source MAC address of the frame, and the switch forwards the frame based on the destination MAC address. That logic occurs independently from any configured routing logic.

When using a Layer 3 switch, the switch acts as the default router for endpoint hosts. As usual, to send a packet to a default router, a host uses ARP to learn the default router's MAC address and then encapsulates the packet in a frame destined to the default router's MAC address. As a result, when using a Layer 3 switch with SVIs, hosts send their frames to the SVI's MAC address. Those frames arrive in a physical switch port, which forwards the frame based on the destination MAC address, but to the internal destination of the VLAN interface. That process triggers internal routing actions like stripping data-link headers, making a routing decision, and so on.

Alternately, the Layer 3 switch configuration can make a physical port act like a router interface instead of a switch interface. To do so, the switch configuration makes that port a

routed port. On a *routed port*, the switch does not perform Layer 2 switching logic on that frame. Instead, frames arriving in a routed port trigger the Layer 3 routing logic, including

1. Stripping off the incoming frame's Ethernet data-link header/trailer
2. Making a Layer 3 forwarding decision by comparing the destination IP address to the IP routing table
3. Adding a new Ethernet data-link header/trailer to the packet
4. Forwarding the packet, encapsulated in a new frame

This third major section of the chapter examines routed interfaces as configured on Cisco Layer 3 switches, but with a particular goal in mind: to also discuss Layer 3 EtherChannels. L3 EtherChannels use routed ports, so before learning about L3 EtherChannels you must first understand routed ports.

Implementing Routed Interfaces on Switches

When a Layer 3 switch needs a Layer 3 interface connected to a subnet, and only one physical interface connects to that subnet, the design can use a routed port instead of an SVI. Stated differently, when routing over a point-to-point link connected to one other device only, using a routed port makes sense. Conversely, when the Layer 3 switch needs a Layer 3 interface connected to a subnet, and many physical interfaces on the switch connect to that subnet, the design must use an SVI.

To see why, consider the design in Figure 18-5, which repeats the same design from Figure 18-4, which was used in the SVI examples. In that design, the gray rectangle on the right represents the switch and its internals. On the right of the switch, at least two access ports sit in both VLAN 10 and VLAN 20. The Layer 3 switch must use SVIs as an interface into those VLANs because two or more ports connect to the VLAN.

Key Topic

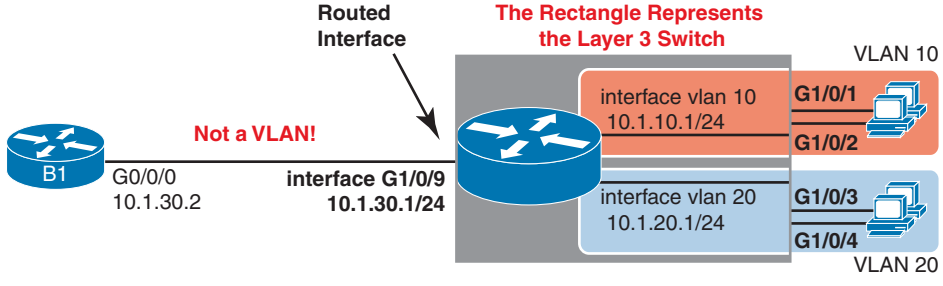

Figure 18-5 *Routing on a Routed Interface on a Switch*

The link on the left of the figure connects from the switch to Router B1. The design needs routing between Router B1 and the switch. While earlier Example 18-6 and Example 18-7 show how to accomplish that routing with an access port and an SVI on the switch, using a routed port works as well, given that the design creates a point-to-point topology between the two devices.

Enabling a switch interface to be a routed interface instead of a switched interface is simple: just use the **no switchport** subcommand on the physical interface. Cisco switches capable

of being a Layer 3 switch use a default of the **switchport** command to each switch physical interface. Think about the word *switchport* for a moment. With that term, Cisco tells the switch to treat the port like it is a port on a switch—that is, a Layer 2 port on a switch. To make the port stop acting like a switch port and instead act like a router port, use the **no switchport** command on the interface.

Once the port is acting as a routed port, think of it like a router interface. That is, configure the IP address on the physical port, as implied in Figure 18-5. Example 18-10 shows a completed configuration for the interfaces configured on the switch in Figure 18-5. Note that the design uses the exact same IP subnets as the example that showed SVI configuration in Example 18-6, but now, the port connected to subnet 10.1.30.0 has been converted to a routed port. All you have to do is add the **no switchport** command to the physical interface and configure the IP address on the physical interface.

Example 18-10 *Configuring Interface G0/1 on Switch SW1 as a Routed Port*

```
ip routing
!
interface vlan 10
 ip address 10.1.10.1 255.255.255.0
!
interface vlan 20
 ip address 10.1.20.1 255.255.255.0
!
interface gigabitethernet 1/0/9
 no switchport
 ip address 10.1.30.1 255.255.255.0
```

Once configured, the routed interface will show up differently in command output in the switch. In particular, for an interface configured as a routed port with an IP address, like interface GigabitEthernet1/0/9 in the previous example:

Key Topic

show interfaces: Similar to the same command on a router, the output will display the IP address of the interface. (Conversely, for switch ports, this command does not list an IP address.)

show interfaces status: Under the "VLAN" heading, instead of listing the access VLAN or the word *trunk*, the output lists the word *routed*, meaning that it is a routed port.

show ip route: Lists the routed port as an outgoing interface in routes.

show interfaces *type number* **switchport:** If a routed port, the output is short and confirms that the port is not a switch port. (If the port is a Layer 2 port, this command lists many configuration and status details.)

Example 18-11 shows samples of all four of these commands as taken from the switch as configured in Example 18-10.

Example 18-11 *Verification Commands for Routed Ports on Switches*

```
SW1# show interfaces g1/0/9
GigabitEthernet1/0/9 is up, line protocol is up (connected)
  Hardware is Gigabit Ethernet, address is 4488.165a.f277 (bia 4488.165a.f277)
  Internet address is 10.1.30.1/24
! lines omitted for brevity

SW1# show interfaces status
! Only ports related to the example are shown; the command lists physical only
Port      Name               Status       Vlan       Duplex  Speed Type
Gi1/0/1                      connected    10         a-full a-1000 10/100/1000BaseTX
Gi1/0/2                      connected    10         a-full a-1000 10/100/1000BaseTX
Gi1/0/3                      connected    20         a-full a-1000 10/100/1000BaseTX
Gi1/0/4                      connected    20         a-full a-1000 10/100/1000BaseTX
Gi1/0/9                      connected    routed     a-full a-1000 10/100/1000BaseTX

SW1# show ip route
! legend omitted for brevity

       10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C         10.1.10.0/24 is directly connected, Vlan10
L         10.1.10.1/32 is directly connected, Vlan10
C         10.1.20.0/24 is directly connected, Vlan20
L         10.1.20.1/32 is directly connected, Vlan20
C         10.1.30.0/24 is directly connected, GigabitEthernet1/0/9
L         10.1.30.1/32 is directly connected, GigabitEthernet1/0/9

SW1# show interfaces g0/1 switchport
Name: Gi1/0/9
Switchport: Disabled
```

So, with two options—SVI and routed ports—where should you use each?

> For any topologies with a point-to-point link between two devices that do routing, a routed interface works better. For any other topology, you must use SVIs.

Figure 18-6 shows an example of where to use SVIs and where to use routed ports in a typical core/distribution/access design. In this design, the core (Core1, Core2) and distribution (D11, D12, D21, D22) switches perform Layer 3 switching. The access switches (labeled A11, A12, and so on) perform only Layer 2 switching. All the ports that are links directly between the Layer 3 switches can be routed interfaces. For VLANs for which many interfaces (access and trunk) connect to the VLAN, SVIs make sense because the SVIs can send and receive traffic out multiple ports in the same VLAN on the same switch. In this design, all the ports on Core1 and Core2 will be routed ports, while the four distribution switches will use some routed ports and some SVIs.

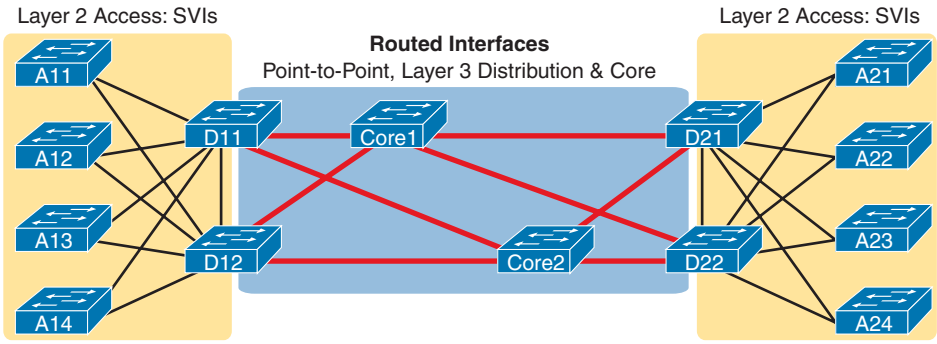

Figure 18-6 *Using Routed Interfaces for Core and Distribution Layer 3 Links*

Implementing Layer 3 EtherChannels

So far, this section has stated that routed interfaces can be used with a single point-to-point link between pairs of Layer 3 switches, or between a Layer 3 switch and a router. However, in most designs, the network engineers use at least two links between each pair of distribution and core switches, as shown in Figure 18-7.

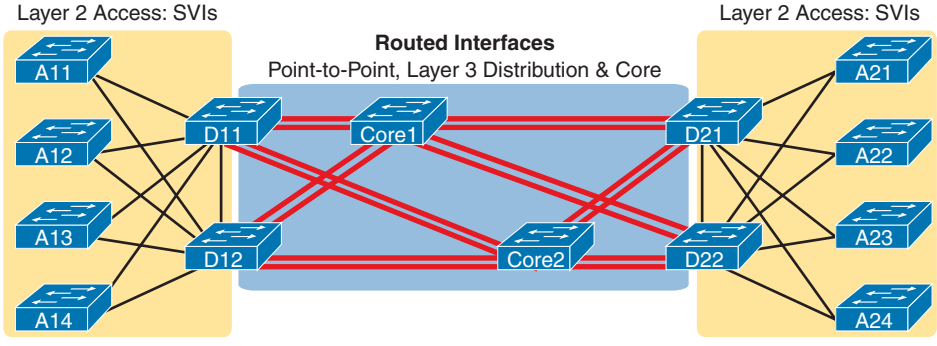

Figure 18-7 *Two Links Between Each Distribution and Core Switch*

While each individual port in the distribution and core could be treated as a separate routed port, it is better to combine each pair of parallel links into a Layer 3 EtherChannel. Without using EtherChannel, you can still make each port on each switch in the center of the figure be a routed port. It works. However, once you enable a routing protocol but don't use EtherChannels, each Layer 3 switch will now learn two IP routes with the same neighboring switch as the next hop—one route over one link, another route over the other link.

Using a Layer 3 EtherChannel makes more sense with multiple parallel links between two switches. By doing so, each pair of links acts as one Layer 3 link. So, each pair of switches has one routing protocol neighbor relationship with the neighbor, and not two. Each switch learns one route per destination per pair of links, and not two. IOS then balances the traffic, often with better balancing than the balancing that occurs with the use of multiple IP routes to the same subnet. Overall, the Layer 3 EtherChannel approach works much better than leaving each link as a separate routed port and using Layer 3 balancing.

Compared to what you have already learned, configuring a Layer 3 EtherChannel takes only a little more work. Chapter 10, "RSTP and EtherChannel Configuration," already showed you how to configure an EtherChannel. This chapter has already shown how to make a port a Layer 3 routed port. Next, you have to combine the two ideas by combining both the EtherChannel and routed port configuration. The following checklist shows the steps, assuming a static definition for the EtherChannel.

Config Checklist

Step 1. Configure the physical interfaces as follows, in interface configuration mode:

- **a.** Add the **no switchport** command to make each physical port a routed port.
- **b.** Add the **channel-group** *number* **mode on** command to add it to the channel. Use the same number for all physical interfaces on the same switch, but the number used (the channel-group number) can differ on the two neighboring switches.

Step 2. Configure the PortChannel interface:

- **a.** Use the **interface port-channel** *number* command to move to port-channel configuration mode for the same channel number configured on the physical interfaces.
- **b.** Add the **no switchport** command to make sure that the port-channel interface acts as a routed port. (IOS may have already added this command.)
- **c.** Use the **ip address** *address mask* command to configure the address and mask.

NOTE Cisco uses the term *EtherChannel* in concepts discussed in this section and then uses the term *PortChannel*, with command keyword **port-channel**, when verifying and configuring EtherChannels. For the purposes of understanding the technology, you may treat these terms as synonyms. However, it helps to pay close attention to the use of the terms *PortChannel* and *EtherChannel* as you work through the examples in this section because IOS uses both.

Example 18-12 shows an example of the configuration for a Layer 3 EtherChannel for switch SW1 in Figure 18-8. The EtherChannel defines port-channel interface 12 and uses subnet 10.1.12.0/24. The design makes use of TenGigabit interfaces between the two switches.

Key Topic

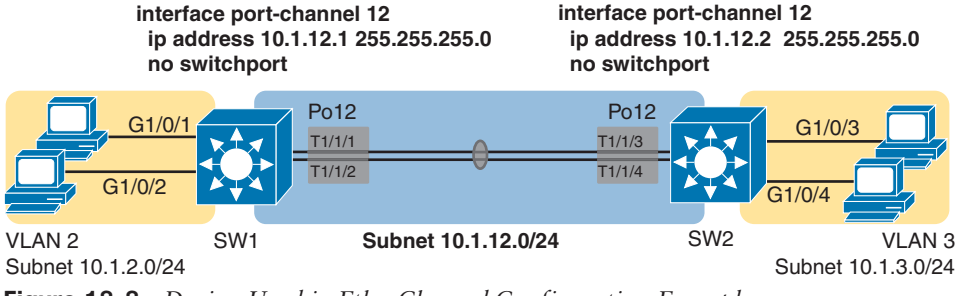

Figure 18-8 *Design Used in EtherChannel Configuration Examples*

Example 18-12 *Layer 3 EtherChannel Configuration on Switch SW1*

```
interface TenGigabit1/1/1
 no switchport
 no ip address
 channel-group 12 mode on
!
interface TenGigabit1/1/2
 no switchport
 no ip address
 channel-group 12 mode on
!
interface Port-channel12
 no switchport
 ip address 10.1.12.1 255.255.255.0
```

Of particular importance, note that although the physical interfaces and PortChannel interface are all routed ports, the IP address should be placed on the PortChannel interface only. In fact, when the **no switchport** command is configured on an interface, IOS adds the **no ip address** command to the interface. Then configure the IP address on the PortChannel interface only.

Once configured, the PortChannel interface appears in several commands, as shown in Example 18-13. The commands that list IP addresses and routes refer to the PortChannel interface. Also, note that the **show interfaces status** command lists the fact that the physical ports and the port-channel 12 interface are all routed ports.

Example 18-13 *Verification Commands Listing Interface Port-Channel12 from Switch SW1*

```
SW1# show interfaces port-channel 12
Port-channel12 is up, line protocol is up (connected)
  Hardware is EtherChannel, address is 4488.165a.f26c (bia 4488.165a.f26c)
  Internet address is 10.1.12.1/24
! lines omitted for brevity

SW1# show interfaces status
! Only ports related to the example are shown.
Port        Name               Status       Vlan       Duplex  Speed Type
Te1/1/1                        connected    routed       full    10G SFP-10GBase-SR
Te1/1/2                        connected    routed       full    10G SFP-10GBase-SR
Po12                           connected    routed     a-full    a-10G N/A

SW1# show ip route
! legend omitted for brevity
       10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C         10.1.2.0/24 is directly connected, Vlan2
L         10.1.2.1/32 is directly connected, Vlan2
C         10.1.12.0/24 is directly connected, Port-channel12
L         10.1.12.1/32 is directly connected, Port-channel12
```

For a final bit of verification, you can examine the EtherChannel directly with the **show etherchannel summary** command as listed in Example 18-14. Note in particular that it lists a flag legend for characters that identify key operational states, such as whether a port is bundled (included) in the PortChannel (P) and whether it is acting as a routed (R) or switched (S) port.

Example 18-14 *Verifying the EtherChannel*

```
SW1# show etherchannel 12 summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
12     Po12(RU)         -          Te1/1/1(P)      Te1/1/2(P)
```

Troubleshooting Layer 3 EtherChannels

When you are troubleshooting a Layer 3 EtherChannel, there are two main areas to consider. First, you need to look at the configuration of the **channel-group** command, which enables an interface for an EtherChannel. Second, you should check a list of settings that must match on the interfaces for a Layer 3 EtherChannel to work correctly.

As for the **channel-group** interface subcommand, this command can enable EtherChannel statically or dynamically. If dynamic, this command's keywords imply either Port Aggregation Protocol (PAgP) or Link Aggregation Control Protocol (LACP) as the protocol to negotiate between the neighboring switches whether they put the link into the EtherChannel.

If all this sounds vaguely familiar, it is the exact same configuration covered way back in the Chapter 10 section "Configuring Dynamic EtherChannels." The configuration of the **channel-group** subcommand is exactly the same, with the same requirements, whether configuring Layer 2 or Layer 3 EtherChannels. So, it might be a good time to review those

EtherChannel configuration details from Chapter 10. Regardless of when you review and master those commands, note that the configuration of the EtherChannel (with the **channel-group** subcommand) is the same, whether Layer 2 or Layer 3.

Additionally, you must do more than just configure the **channel-group** command correctly for all the physical ports to be bundled into the EtherChannel. Layer 2 EtherChannels have a longer list of requirements, but Layer 3 EtherChannels also require a few consistency checks between the ports before they can be added to the EtherChannel. The following is the list of requirements for Layer 3 EtherChannels:

Key Topic

no switchport: The PortChannel interface must be configured with the **no switchport** command, and so must the physical interfaces. If a physical interface is not also configured with the **no switchport** command, it will not become operational in the EtherChannel.

Speed: The physical ports in the channel must use the same speed.

duplex: The physical ports in the channel must use the same duplex.

VLAN Routing on a Router's LAN Switch Ports

The earlier major sections of this chapter all show designs with hosts connected to a switch. The switch might do the routing, or a router connected to the switch might do the routing, but the hosts are connected via a cable to the switch.

This last major section discusses a design with only a router, specifically one whose hardware includes a set of LAN switch ports. In effect, the router hardware embeds a LAN switch into the hardware. As a result, the router needs to implement both routing and switching.

Designs that include small branch offices can make good use of a router with embedded switch ports. Those sites need a router that can route over the WAN and may need only a few LAN ports. Many vendors (Cisco included) offer router products that include a small set of LAN switch ports for such cases. Many consumer-grade routers, like the one you have at home for Internet access, also embed a small number of switch ports.

As an example of the hardware, Figure 18-9 shows photos of an ISR1K router and a Catalyst 8200L edge router. Some ISR1Ks, positioned by Cisco as enterprise-class routers for smaller sites, include a few routed ports plus a set of **switched ports**. The routed ports act like ports on routers, as discussed so far in this book, receiving and forwarding IP packets while requiring the configuration of an IP address and mask. The switch ports work like ports on a LAN switch, forwarding and receiving Ethernet frames, with the router's switching logic learning entries for a MAC address table.

The bottom of the figure shows a photo of three Catalyst 8200 edge platforms (routers). The models in the figure each support four fixed routed Ethernet ports (two RJ-45 and two modular SFP slots). The Network Interface Module (NIM) slot allows for expansion with modular NIMs, including NIMs with four and eight LAN switch ports.

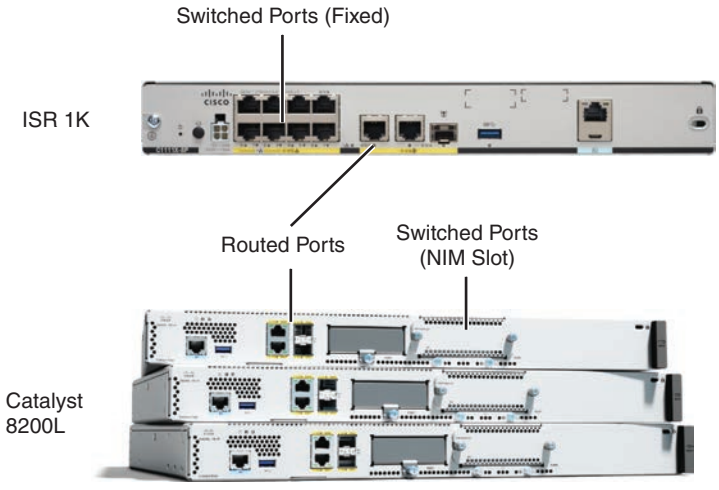

Figure 18-9 *Cisco ISR 1000 and Catalyst 8200L Routers*

Configuring Routing for Embedded Switch Ports

Routers that have switch ports implement both routing and switching. To make that work, configure LAN switching commands on the switch ports and routing commands for the routed interfaces. To stitch the logic together, use a concept familiar from multilayer switch configuration: SVI interfaces.

Figures 18-10 and Figure 18-11 give the context for an example. Figure 18-10 shows Router B1, at a branch office, with four switch ports connected to some PCs. The design calls for two VLANs with the same VLAN IDs and subnets, as in several earlier examples in this chapter. The router (the same model of ISR1K shown in Figure 18-9) uses routed port G0/0/1 to connect to the WAN.

Key Topic

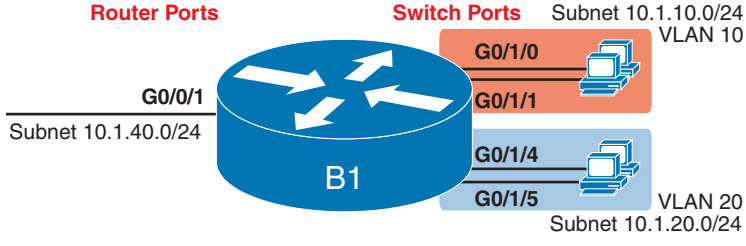

Figure 18-10 *Physical Representation of Router with Routed and Switched Ports*

Figure 18-11 shows the internal logic using SVIs that give the routing process inside Router B1 interfaces that connect directly to the subnets that reside on VLANs 10 and 20. The concepts should look familiar; the internal concepts work like a Layer 3 switch.

Key Topic

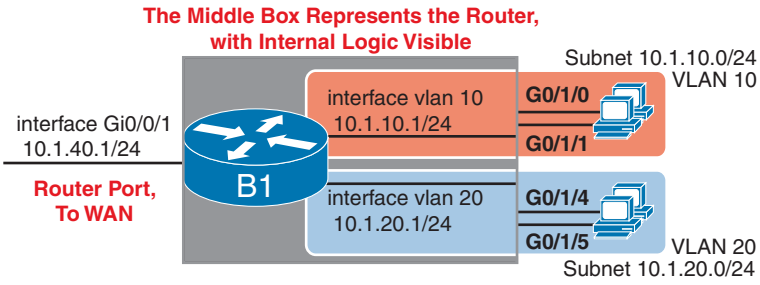

Figure 18-11 *Configuration View of a Router with Routed and Switched Ports*

The configuration steps include familiar tasks for the LAN switch, other familiar tasks for IP routing and the routed ports, and the additional task of configuring the SVIs. To review, the following configuration checklist lists a minimal set of tasks, beginning with a default configuration, to enable IP on the three interfaces so that the router has connected routes to all three subnets and can route packets between those connected subnets.

Config Checklist

Step 1. Configure VLANs and assign access VLANs to the switch ports.

- **a.** Use the **vlan** *vlan-number* command in global configuration mode to create each VLAN.
- **b.** Use the **switchport access vlan** *vlan-id* command in interface configuration mode on the switched interfaces to assign the correct access VLAN.
- **c.** (As needed) Use the **no shutdown** command in interface configuration mode to enable the interface.

Step 2. Configure an SVI interface for each VLAN used by the switch ports.

- **a.** Use the **interface vlan** *vlan_id* command in global configuration mode to create a VLAN interface. Use the same number as the VLAN ID.
- **b.** Use the **ip address** *address mask* command in VLAN interface configuration mode to assign an IP address and mask to the interface, enabling IPv4 processing on the interface.
- **c.** (As needed) Use the **no shutdown** command in interface configuration mode to enable the VLAN interface.

Step 3. Configure any routed interfaces with IP addresses.

- **a.** Use the **ip address** *address mask* command in interface configuration mode to assign an IP address and mask to the interface, enabling IPv4 processing.
- **b.** Configure any routing protocols as needed.

Example 18-15 shows a sample configuration to match Figure 18-11. The top of the configuration adds VLANs 10 and 20, assigning them as access VLANs to the ports per the figure. The **interface vlan** commands create the SVIs, with IP address configuration added to each. The final interface, G0/0/1, a routed port, shows traditional router interface configuration with an **ip address** command but without a **switchport access** subcommand.

Example 18-15 *VLAN Interface Configuration for Routing to/from Switch Port VLANs*

```
vlan 10
vlan 20
!
interface GigabitEthernet0/1/0
 switchport access vlan 10
!
interface GigabitEthernet0/1/1
 switchport access vlan 10
!
interface GigabitEthernet0/1/4
 switchport access vlan 20
!
interface GigabitEthernet0/1/5
 switchport access vlan 20
!
interface vlan 10
 ip address 10.1.10.1 255.255.255.0
!
interface vlan 20
 ip address 10.1.20.1 255.255.255.0
!
interface gigabitEthernet0/0/1
 description physical, routed interface

ip address 10.1.40.1 255.255.255.0
```

Verifying Routing for Embedded Switch Ports

With the VLAN configuration shown in the previous section, the router is ready to route packets between the subnets as shown in Figure 18-11. The output in Example 18-16 lists two routes as connected to different VLAN interfaces, with one route connected to the one routed interface (G0/0/1). Beyond those routes, the router also needs routes to remote subnets. The router could use static routes or a routing protocol.

Example 18-16 *Connected Routes on Router B1 from Example 18-15*

```
B1# show ip route connected
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
! Remaining legend omitted for brevity

      10.0.0.0/8 is variably subnetted, 13 subnets, 2 masks
C        10.1.10.0/24 is directly connected, Vlan10
L        10.1.10.1/32 is directly connected, Vlan10
C        10.1.20.0/24 is directly connected, Vlan20
L        10.1.20.1/32 is directly connected, Vlan20
C        10.1.30.0/24 is directly connected, GigabitEthernet0/0/1
L        10.1.30.1/32 is directly connected, GigabitEthernet0/0/1
```

The router adds the connected routes when the related interface reaches an up/up state. For SVIs, the state varies based on rules related to the state of the underlying VLAN and the ports in that VLAN. Those rules work just as described in this chapter's earlier section titled "Troubleshooting Routing with SVIs."

The router performs LAN switching with the same conventions you learned about LAN switches in Parts II and III of this book. For instance, the router supports the exact commands you expect to find on LAN switches to support the LAN switch ports. Example 18-17 shows the MAC addresses learned on the four switch ports shown in the most recent figures. It confirms that addresses learned on ports G0/1/0 and G0/1/1 exist in VLAN 10 while those learned on ports G0/1/4 and G0/1/5 exist in VLAN 20.

Example 18-17 *MAC Address Table on the Router*

```
B1# show mac address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  10    0200.aaaa.aaaa    DYNAMIC     Gi0/1/0
  10    0200.bbbb.bbbb    DYNAMIC     Gi0/1/1
  20    0200.cccc.cccc    DYNAMIC     Gi0/1/4
  20    0200.dddd.dddd    DYNAMIC     Gi0/1/5
Total Mac Addresses for this criterion: 4
```

NOTE Switches use the **show vlan** [**brief**] command to display VLAN configuration. Routers with embedded switch ports have a command that mimics the switch command: **show vlan-switch** [**brief**] (IOS) and **show vlan** [**brief**] (IOS XE). As a reminder, the similar **show vlans** command on routers does not list VLANs but rather ROAS configuration.

Identifying Switched Ports in Routers

When you look at a router or use the CLI of a router, how do you know which Ethernet ports happen to be routed ports and which are switch ports? Finding the answer can be difficult. This last short section tells you how.

First, you do not configure a router port's role as a routed or switch port. Instead, Cisco creates router interfaces that act as one or the other. You can research the router model and family at www.cisco.com and learn about the fixed ports that come with each model. That documentation also identifies modular options for cards like NIMs.

You can also discover the router's switched ports from the CLI, but the conventions are not obvious. Most **show** commands do not help, but a few do. Of note:

Key Topic

show running-config, show startup-config: These commands do not identify routed versus switched ports.

show interfaces, show interfaces description, show ip interface brief, show protocols: All these commands list all interfaces, routed and switched, with no notation of which is which.

show interfaces status: This command lists only switched ports.

Using **show interfaces status** to identify the switch ports makes the most sense.

Additionally, you can confirm which ports act as routed versus switched ports by trying to configure commands supported only on one or the other. For instance, routed ports accept the **ip address** *address mask* subcommand but reject the **switchport access vlan** *vlan-id* subcommand. Switched ports accept the opposite.

Example 18-18 shows a few cases of identifying routed and switched ports using Router B1 from the previous examples. It first shows only the switched ports G0/1/0–G0/1/7 in the **show interfaces status** command output. Note the output follows the format from LAN switches, identifying the access VLAN assigned to each interface. Following that, the output shows a switched port rejecting the **ip address** interface subcommand.

Example 18-18 *Displaying Switched Interfaces but Omitting Routed Interfaces*

```
B1# show interfaces status

Port         Name               Status       Vlan       Duplex  Speed Type
Gi0/1/0                         connected    routed     a-full a-1000 10/100/1000BaseTX
Gi0/1/1                         connected    10         a-full a-1000 10/100/1000BaseTX
Gi0/1/2                         notconnect   1            auto   auto 10/100/1000BaseTX
Gi0/1/3                         notconnect   1            auto   auto 10/100/1000BaseTX
Gi0/1/4                         connected    20         a-full a-1000 10/100/1000BaseTX
Gi0/1/5                         connected    20         a-full a-1000 10/100/1000BaseTX
Gi0/1/6                         notconnect   1            auto   auto 10/100/1000BaseTX
Gi0/1/7                         notconnect   1            auto   auto 10/100/1000BaseTX
B1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
B1(config)# interface g0/1/7
B1(config-if)# ip address 10.1.90.1 255.255.255.0
               ^
% Invalid input detected at '^' marker.
B1(config-if)#
```

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 18-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 18-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review config checklists		Book, website
Review command tables		Book
Do labs		Blog
Watch video		Website

Review All the Key Topics

Key Topic

Table 18-3 Key Topics for Chapter 18

Key Topic Element	Description	Page Number
Figure 18-2	Concept of VLAN subinterfaces on a router	459
List	Two alternative methods to configure the native VLAN in a ROAS configuration	460
List	Troubleshooting suggestions for ROAS configuration	463
Figure 18-3	Layer 3 switching with SVIs concept and configuration	464
Figure 18-4	Layer 3 switching with routed ports concept and configuration	465
List	Troubleshooting suggestions for correct operation of a Layer 3 switch that uses SVIs	467
Figure 18-5	A figure showing the concepts behind IP routing on a routed interface on a switch	470
List	**show** commands that list Layer 3 routed ports in their output	471
Figure 18-8	Layer 3 EtherChannel concept and configuration	474
List	List of configuration settings that must be consistent before IOS will bundle a link with an existing Layer 3 EtherChannel	477
Figure 18-10	A concept diagram showing a router with routed and switched ports	478
Figure 18-11	Conceptual view of the configuration for a router with integrated LAN switch ports	479
List	Router commands and whether they list routed ports, switched ports, or both	481

Key Terms You Should Know

Layer 3 EtherChannel (L3 EtherChannel), Layer 3 switch, multilayer switch, routed port, router-on-a-stick (ROAS), subinterfaces, switched port, switched virtual interface (SVI), VLAN interface

Command References

Tables 18-4 and 18-5 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 18-4 Chapter 18 Configuration Command Reference

Command	Description
interface *type number.subint*	Router global command to create a subinterface and to enter configuration mode for that subinterface
encapsulation dot1q *vlan-id* [**native**]	Router subinterface subcommand that tells the router to use 802.1Q trunking, for a particular VLAN, and with the **native** keyword, to not encapsulate in a trunking header
[**no**] **ip routing**	Global command that enables (**ip routing**) or disables (**no ip routing**) the routing of IPv4 packets on a router or Layer 3 switch
interface vlan *vlan-id*	Global command to create a VLAN interface and to enter VLAN interface configuration mode; valid on Layer 3 switches and on routers that have embedded LAN switch ports
[**no**] **switchport**	Layer 3 switch subcommand that makes the port act as a Layer 2 port (**switchport**) or Layer 3 routed port (**no switchport**)
interface port-channel *channel-number*	A switch command to enter PortChannel configuration mode and also to create the PortChannel if not already created
channel-group *channel-number* **mode** {**auto** | **desirable** | **active** | **passive** | **on**}	Interface subcommand that enables EtherChannel on the interface
[**no**] **autostate**	Interface subcommand on VLAN interfaces that enables (**autostate**) or disables (**no autostate**) the autostate feature.
vlan *vlan-id*	Global config command that both creates the VLAN and puts the CLI into VLAN configuration mode
name *vlan-name*	VLAN subcommand that names the VLAN
[**no**] **shutdown**	VLAN mode subcommand that enables (**no shutdown**) or disables (**shutdown**) the VLAN

Table 18-5 Chapter 18 EXEC Command Reference

Command	Description
show ip route	Lists the router's entire routing table
show ip route [**connected**]	Lists a subset of the IP routing table
show vlans	Lists VLAN configuration and statistics for VLAN trunks configured on routers

Command	Description
show interfaces [**interface** *type number*]	Lists detailed status and statistical information, including IP address and mask, about all interfaces (or the listed interface only)
show interfaces [**interface** *type number*] **status**	Among other facts, for switch ports, lists the access VLAN or the fact that the interface is a trunk; or, for routed ports, lists "routed"
show interfaces *interface-id* **switchport**	For switch ports, lists information about any interface regarding administrative settings and operational state; for routed ports, the output simply confirms the port is a routed (not switched) port
show interfaces vlan *number*	Lists the interface status, the switch's IPv4 address and mask, and much more
show etherchannel [*channel-group-number*] **summary**	Lists information about the state of EtherChannels on this switch, including whether the channel is a Layer 2 or Layer 3 EtherChannel
show mac address-table dynamic	Shows all dynamically learned MAC table entries on LAN switches and on routers that have embedded switch ports
show vlan-switch [**brief**]	(IOS only) On a router with embedded LAN switch ports, lists the VLAN configuration
show vlan [**brief**]	(IOS XE only) On a router with embedded LAN switch ports, lists the VLAN configuration

CHAPTER 19

IP Addressing on Hosts

This chapter covers the following exam topics:

1.0 Network Fundamentals

1.10 Verify IP parameters for Client OS (Windows, Mac OS, Linux)

4.0 IP Services

4.3 Explain the role of DHCP and DNS within the network

4.6 Configure and verify DHCP client and relay

In the world of TCP/IP, the word *host* refers to any device with an IP address: your phone, your tablet, a PC, a server, a router, a switch—any device that uses IP to provide a service or just needs an IP address to be managed. The term *host* includes some less-obvious devices as well: the electronic advertising video screen at the mall, your electrical power meter that uses the same technology as mobile phones to submit your electrical usage information for billing, your new car.

No matter the type of host, any host that uses IPv4 needs four IPv4 settings to work properly:

- IP address
- Subnet mask
- Default routers
- DNS server IP addresses

This chapter discusses these basic IP settings on hosts. The chapter begins by discussing how a host can dynamically learn these four settings using the Dynamic Host Configuration Protocol (DHCP). The second half of this chapter then shows how to find the settings on hosts and the key facts to look for when displaying the settings.

"Do I Know This Already?" Quiz

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

Table 19-1 "Do I Know This Already?" Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions
Dynamic Host Configuration Protocol	1–4
Identifying Host IPv4 Settings	5, 6

1. A PC connects to a LAN and uses DHCP to lease an IP address for the first time. Of the usual four DHCP messages that flow between the PC and the DHCP server, which ones does the client send? (Choose two answers.)
   a. Acknowledgment
   b. Discover
   c. Offer
   d. Request
2. Which of the following kinds of information are part of a DHCP server configuration? (Choose two answers.)
   a. Ranges of IP addresses in subnets that the server should lease
   b. Ranges of IP addresses to not lease per subnet
   c. DNS server hostnames
   d. The default router IP and MAC address in each subnet
3. Which answers list a criterion for choosing which router interfaces need to be configured as a DHCP relay agent? (Choose two answers.)
   a. If the subnet off the interface does not include a DHCP server
   b. If the subnet off the interface does include a DHCP server
   c. If the subnet off the interface contains DHCP clients
   d. If the router interface already has an **ip address dhcp** command
4. A router connects to an Internet service provider (ISP) using its G0/0/0 interface, with the **ip address dhcp** command configured. What does the router do with the DHCP-learned default gateway information?
   a. The router ignores the default gateway value learned from the DHCP server.
   b. The router uses the default gateway just like a host, ignoring its routing table.
   c. The router forwards received packets based on its routing table but uses its default gateway setting to forward packets it generates itself.
   d. The router adds a default route based on the default gateway to its IP routing table.
5. In the following excerpt from a command on a Mac, which of the following parts of the output represent information learned from a DHCP server? (Choose two answers.)

```
Macprompt$ ifconfig en0
En1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=10b<RXCSUM,TXCSUM,VLAN_HWTAGGING,AV>
        ether 00:6d:e7:b1:9a:11
        inet 172.16.4.2 netmask 0xffffff00 broadcast 172.16.4.255
```

a. 00:6d:e7:b1:9a:11
b. 172.16.4.2
c. 0xffffff00
d. 172.16.4.255

6. Which of the following commands on a Windows OS should list both the host's IP address and DNS servers' IP addresses as learned with DHCP?
   a. ifconfig
   b. ipconfig
   c. ifconfig /all
   d. ipconfig /all

Foundation Topics

Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) provides one of the most commonly used services in a TCP/IP network. The vast majority of hosts in a TCP/IP network are user devices, and the vast majority of user devices learn their IPv4 settings using DHCP.

Using DHCP has several advantages over the other option of manually configuring IPv4 settings. The configuration of host IP settings sits in a DHCP server, with each client learning these settings using DHCP messages. As a result, the host IP configuration is controlled by the IT staff, rather than on local configuration on each host, resulting in fewer user errors. DHCP allows the permanent assignment of host addresses, but more commonly, a temporary lease of IP addresses. With these leases, the **DHCP server** can reclaim IP addresses when a device is removed from the network, making better use of the available addresses.

DHCP also enables mobility. For example, every time a user moves to a new location with a tablet computer—to a coffee shop, a client location, or back at the office—the user's device can connect to another wireless LAN, use DHCP to lease a new IP address in that LAN, and begin working on the new network. Without DHCP, the user would have to ask for information about the local network and configure settings manually, with more than a few users making mistakes.

Although DHCP works automatically for user hosts, it does require some preparation from the network, with some configuration on routers. In some enterprise networks, that router configuration can be a single command on many of the router's LAN interfaces (**ip helper-address** *server-ip*), which identifies the DHCP server by its IP address. In other cases, the router acts as the DHCP server. Regardless, the routers have some role to play.

This first major section of the chapter takes a tour of DHCP, including concepts and the router configuration to enable the routers to work well with a separate DHCP server.

DHCP Concepts

Sit back for a moment and think about the role of DHCP for a host computer. The host acts as a **DHCP client**. As a DHCP client, the host begins with no IPv4 settings—no IPv4 address, no mask, no default router, and no **DNS server** IP addresses. But a DHCP client does have

knowledge of the DHCP protocol, so the client can use that protocol to (a) discover a DHCP server and (b) request to lease an IPv4 address.

DHCP uses the following four messages between the client and server. (Also, to help remember the messages, note that the first letters spell DORA):

Discover: Sent by the DHCP client to find a willing DHCP server

Offer: Sent by a DHCP server to offer to lease to that client a specific IP address (and inform the client of its other parameters)

Request: Sent by the DHCP client to ask the server to lease the IPv4 address listed in the Offer message

Acknowledgment: Sent by the DHCP server to assign the address and to list the mask, default router, and DNS server IP addresses

DHCP clients, however, have a somewhat unique problem: they do not have an IP address yet, but they need to send these DHCP messages inside IP packets. To make that work, DHCP messages make use of two special IPv4 addresses that allow a host that has no IP address to still be able to send and receive messages on the local subnet:

Key Topic

0.0.0.0: An address reserved for use as a source IPv4 address for hosts that do not yet have an IP address.

255.255.255.255: The local broadcast IP address. Packets sent to this destination address are broadcast on the local data link, but routers do not forward them.

To see how these addresses work, Figure 19-1 shows an example of the IP addresses used between a host (A) and a DHCP server on the same LAN. Host A, a client, sends a Discover message, with source IP address of 0.0.0.0 because Host A does not have an IP address to use yet. Host A sends the packet to destination 255.255.255.255, which is sent in a LAN broadcast frame, reaching all hosts in the subnet. The client hopes that there is a DHCP server on the local subnet. Why? Packets sent to 255.255.255.255 only go to hosts in the local subnet; Router R1 will not forward this packet.

NOTE Figure 19-1 shows one example of the addresses that can be used in a DHCP request. This example shows details assuming the DHCP client chooses to use a DHCP option called the *broadcast flag*; all examples in this book assume the broadcast flag is used.

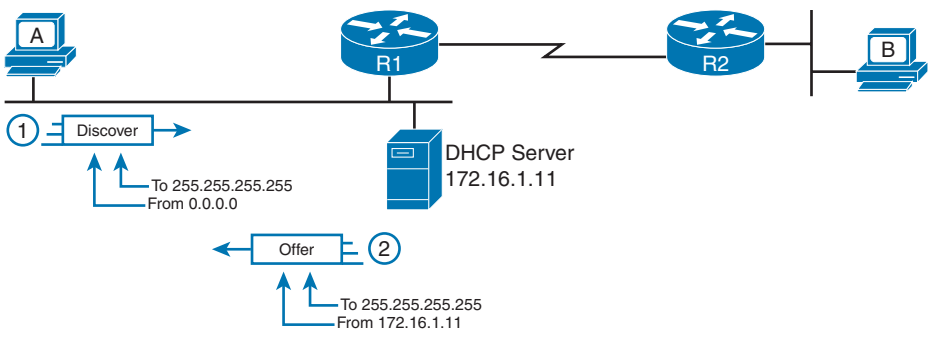

Figure 19-1 *DHCP Discover and Offer*

Now look at the Offer message sent back by the DHCP server. The server sets the destination IP address to 255.255.255.255 again. Why? Host A still does not have an IP address, so the server cannot send a packet directly to host A. So, the server sends the packet to "all local hosts in the subnet" address (255.255.255.255). (The packet is also encapsulated in an Ethernet broadcast frame.)

Note that all hosts in the subnet receive the Offer message. However, the original Discover message lists a number called the client ID, which includes the host's MAC address, that identifies the original host (Host A in this case). As a result, Host A knows that the Offer message is meant for host A. The rest of the hosts will receive the Offer message, but notice that the message lists another device's DHCP client ID, so the rest of the hosts ignore the Offer message.

APIPA IP Addresses (169.254.x.x)

If the DHCP process fails for a DHCP client, hosts have a default means to self-assign an IP address using a feature called Automatic Private IP Addressing (**APIPA**). When the DHCP process fails, the DHCP client self-assigns an APIPA IP address from within a subset of the 169.254.0.0 Class B network, along with default mask 255.255.0.0.

Using an APIPA address does not help devices work like a normal host. For instance, hosts do not know of a default router or learn a list of DNS servers. The host can send packets only to other APIPA hosts on the same LAN.

The APIPA process works like this:

1. The client chooses any IP address from network 169.254.0.0 (actual address range: 169.54.1.0 – 169.254.254.255).
2. The DHCP client discovers if any other host on the same link already uses the APIPA address it chose by using ARP to perform Duplicate Address Detection (DAD). If another host already uses the same address, the client stops using the address and chooses another.
3. The client can send/receive packets on the local network only.

Supporting DHCP for Remote Subnets with DHCP Relay

Network engineers have a major design choice to make with DHCP: Do they put a DHCP server in every LAN subnet or locate a DHCP server in a central site? The question is legitimate. Cisco routers can act as the DHCP server, so a distributed design could use the router at each site as the DHCP server. With a DHCP server in every subnet, as shown in Figure 19-1, the protocol flows stay local to each LAN.

However, a centralized DHCP server approach has advantages as well. In fact, some Cisco design documents suggest a centralized design as a best practice, in part because it allows for centralized control and configuration of all the IPv4 addresses assigned throughout the enterprise.

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

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

With a centralized DHCP server, those DHCP messages that flowed only on the local subnet in Figure 19-1 somehow need to flow over the IP network to the centralized DHCP server and back. To make that work, the routers connected to the remote LAN subnets need an interface subcommand: the **ip helper-address** *server-ip* command.

The **ip helper-address** *server-ip* subcommand tells the router to do the following for the messages coming in an interface, from a DHCP client:

Key Topic

1. Watch for incoming DHCP messages, with destination IP address 255.255.255.255.
2. Change that packet's source IP address to the router's incoming interface IP address.
3. Change that packet's destination IP address to the address of the DHCP server (as configured in the **ip helper-address** command).
4. Route the packet to the DHCP server.

This command gets around the "do not route local subnet broadcast packets sent to address 255.255.255.255" rule by changing the destination IP address. Once the destination has been set to match the DHCP server's IP address, the network can route the packet to the server.

NOTE This feature, by which a router relays DHCP messages by changing the IP addresses in the packet header, is called *DHCP relay*.

Figure 19-2 shows an example of the process. Host A sits on the left, as a DHCP client. The DHCP server (172.16.2.11) sits on the right. R1 has an **ip helper-address 172.16.2.11** command configured, under its G0/0 interface. At Step 1, Router R1 notices the incoming DHCP packet destined for 255.255.255.255. Step 2 shows the results of changing both the source and destination IP address, with R1 routing the packet.

Key Topic

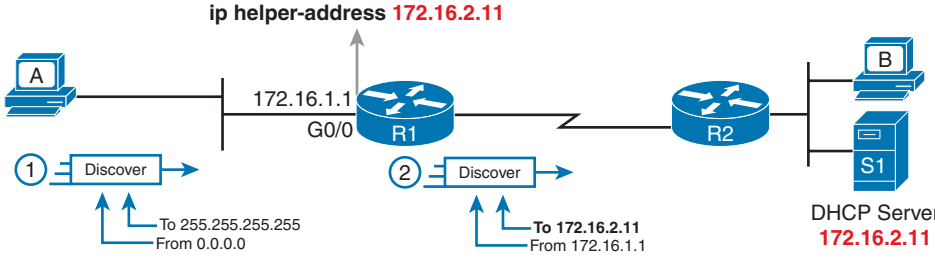

Figure 19-2 *IP Helper Address Effect*

The router uses a similar process for the return DHCP messages from the server. First, for the return packet from the DHCP server, the server simply reverses the source and destination IP address of the packet received from the router (relay agent). For example, in Figure 19-2, the Discover message lists source IP address 172.16.1.1, so the server sends the Offer message back to destination IP address 172.16.1.1.

When a router receives a DHCP message addressed to one of the router's own IP addresses, the router realizes the packet might be part of the DHCP relay feature. When that happens, the **DHCP relay agent** (Router R1) needs to change the destination IP address so that the

real DHCP client (host A), which does not have an IP address yet, can receive and process the packet.

Figure 19-3 shows one example of how these addresses work, when R1 receives the DHCP Offer message sent to R1's own 172.16.1.1 address. R1 changes the packet's destination to 255.255.255.255 and forwards it out G0/0, because the packet was destined to G0/0's 172.16.1.1 IP address. As a result, all hosts in that LAN (including the DHCP client A) will receive the message.

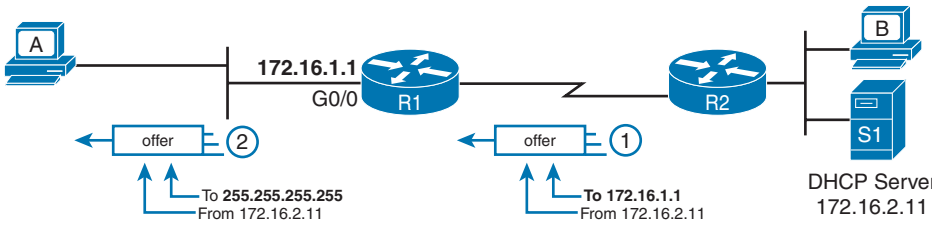

Figure 19-3 *IP Helper Address for the Offer Message Returned from the DHCP Server*

Many enterprise networks use a centralized DHCP server, so the normal router configuration includes an **ip helper-address** command on every LAN interface/subinterface. With that standard configuration, user hosts off any router LAN interface can always reach the DHCP server and lease an IP address.

Information Stored at the DHCP Server

A DHCP server might sound like some large piece of hardware, sitting in a big locked room with lots of air conditioning to keep the hardware cool. However, like most servers, the server is actually software, running on some server OS. The DHCP server could be a piece of software downloaded for free and installed on an old PC. However, because the server needs to be available all the time, to support new DHCP clients, most companies install the software on a very stable and highly available data center, with high availability features. The DHCP service is still created by software, however.

To be ready to answer DHCP clients and to supply them with an IPv4 address and other information, the DHCP server (software) needs configuration. DHCP servers organize these IPv4 settings per subnet, listing an address pool and a default router setting—settings that differ from subnet to subnet. The following list shows the types of settings the DHCP server needs to know to support DHCP clients:

Subnet ID and mask: The DHCP server can use this information to know all addresses in the subnet. (The DHCP server knows to not lease the subnet ID or subnet broadcast address.)

Reserved (excluded) addresses: The server needs to know which addresses in the subnet to *not* lease. This list allows the engineer to reserve addresses to be used as static IP addresses. For example, most router and switch IP addresses, server addresses, and addresses of most anything other than user devices use a statically assigned IP address. Most of the time, engineers use the same convention for all subnets, either reserving the lowest IP addresses in all subnets or reserving the highest IP addresses in all subnets.

Default router(s): This is the IP address of the router on that subnet.

DNS IP address(es): This is a list of DNS server IP addresses.

Figure 19-4 shows the concept behind the preconfiguration on a DHCP server for two LAN-based subnets: 172.16.1.0/24 and 172.16.2.0/24. The DHCP server sits on the right. For each subnet, the server defines all the items in the list. In this case, the configuration reserves the lowest IP addresses in the subnet to be used as static addresses.

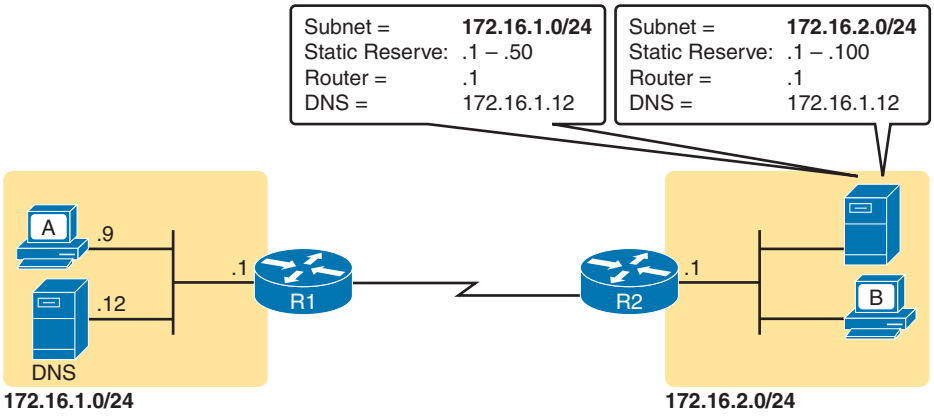

Figure 19-4 *Preconfiguration on a DHCP Server*

The configuration can list other parameters as well. For example, it can set the time limit for leasing an IP address. The server leases an address for a time (usually a number of days), and then the client can ask to renew the lease. If the client does not renew, the server can reclaim the IP address and put it back in the pool of available IP addresses. The server configuration sets the maximum time for the lease.

DHCP uses three allocation modes, based on small differences in the configuration at the DHCP server. *Dynamic allocation* refers to the DHCP mechanisms and configuration described throughout this chapter. Another method, *automatic allocation*, sets the DHCP lease time to infinite. As a result, once the server chooses an address from the pool and assigns the IP address to a client, the IP address remains with that same client indefinitely. A third mode, *static allocation*, preconfigures the specific IP address for a client based on the client's MAC address. That specific client is the only client that then uses the IP address. (Note that this chapter shows examples and configuration for dynamic allocation only.)

Additionally, the DHCP server can be configured to supply some other useful configuration settings. For instance, a server can supply the IP address of a Trivial File Transfer Protocol (TFTP) server. TFTP servers provide a basic means of storing files that can then be transferred to a client host. For instance, Cisco IP phones rely on TFTP to retrieve several configuration files when the phone initializes. DHCP plays a key role by supplying the IP address of the TFTP server that the phones should use.

Configuring DHCP Features on Routers and Switches

Cisco routers and switches support a variety of features. Routers can be configured to act as a DHCP server with just a few straightforward commands—a feature useful in the lab and

in some limited cases. More commonly, the enterprise uses a centralized DHCP server (that does not run on a router) but with the DHCP relay feature on most every router interface. Finally, Cisco routers and switches can also act as DHCP clients, learning their IP addresses from a DHCP server.

This section discusses the DHCP configuration topics mentioned for the CCNA 200-301 V1.1 exam blueprint. Those include the router DHCP relay feature and the configuration to enable DHCP client services on both switches and routers.

NOTE The CCNA 200-301 V1.1 exam blueprint does not mention the DHCP server function, but many people like to use the IOS DHCP server in the lab for testing with DHCP. If interested in how to configure a DHCP server on a router, refer to Appendix K, "Topics from Previous Editions."

Configuring DHCP Relay

Configuring DHCP relay requires a simple decision and a single straightforward configuration command. First, you must identify the interfaces that need the feature. The DHCP relay feature must be configured for any router interface that connects to a subnet where

- DHCP clients exist in the subnet
- DHCP servers do not exist in the subnet

Once such interfaces have been identified, the configuration requires the **ip helper-address** interface subcommand on each of those interfaces. For instance, in the topology of the previous three figures, R1's G0/0 interface needs to be configured with the **ip helper-address 172.16.2.11** interface subcommand. Once enabled on an interface, the IOS DHCP relay agent makes changes in the incoming DHCP messages' addresses as described earlier in the chapter. Without the DHCP relay agent, the DHCP request never arrives at the server.

To verify the relay agent, you can use the **show running-config** command and look for the single configuration command or use the **show ip interface g0/0** command as shown in Example 19-1. The highlighted line confirms the configured setting. Note that if there were no **ip helper-address** commands configured on the interface, the highlighted line would instead read "Helper address is not set."

Example 19-1 *Listing the Current Helper Address Setting with* **show ip interface**

```
R1# show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
 Internet address is 172.16.1.1/24
 Broadcast address is 255.255.255.255
 Address determined by non-volatile memory
 MTU is 1500 bytes
 Helper address is 172.16.2.11
! Lines omitted for brevity
```

Configuring a Switch as DHCP Client

A switch can act as a DHCP client to lease its management IP address. In most cases, you will want to instead use a static IP address so that the staff can more easily identify the switch's address for remote management. However, as an example of how a DHCP client can work, this next topic shows how to configure and verify DHCP client operations on a switch.

NOTE Chapter 6, "Configuring Basic Switch Management," also shows this same example of how to configure a switch to be a DHCP client. This chapter repeats the example here so you can see all the related DHCP configuration details in a single place in this volume.

To configure a switch to use DHCP to lease an address, configure a switch's IP address as normal, but with the **ip address dhcp** interface subcommand. Example 19-2 shows a sample.

Example 19-2 *Switch Dynamic IP Address Configuration with DHCP*

```
Emma# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Emma(config)# interface vlan 1
Emma(config-if)# ip address dhcp
Emma(config-if)# no shutdown
Emma(config-if)# ^Z
Emma#
00:38:20: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
00:38:21: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to up
```

To verify that DHCP worked, start with the traditional way to check IP addresses on switch VLAN interfaces: the **show interfaces vlan** *x* command as demonstrated in Example 19-3. First, check the interface state, because the switch does not attempt DHCP until the VLAN interface reaches an up/up state. Notably, if you forget to issue the **no shutdown** command, the VLAN 1 interface will remain in a shutdown state and be listed as "administratively down" in the **show** command output. Then, if DHCP has not yet worked, you will see the highlighted line shown in the upper part of the example. Once the switch leases an IP address, you will see the different text shown in the bottom half of the example.

Example 19-3 *Verifying DHCP-Learned IP Address on a Switch*

```
! First, output when the switch has not yet leased an IP address
Emma# show interfaces vlan 1
Vlan1 is up, line protocol is up, Autostate Enabled
 Hardware is Ethernet SVI, address is 4488.165a.f247 (bia 4488.165a.f247)
 Internet address will be negotiated using DHCP
! lines omitted for brevity
! Next, the output after DHCP works
Emma# show interfaces vlan 1
Vlan1 is up, line protocol is up, Autostate Enabled
 Hardware is Ethernet SVI, address is 4488.165a.f247 (bia 4488.165a.f247)
 Internet address is 192.168.1.101/24
! lines omitted for brevity
```

Interestingly, output line three, which lists the address, appears the same whether you configure the address explicitly or whether the switch leases the address. For instance, the latter half of Example 19-3 lists 192.168.1.101 as the address, but with no information to identify whether the IP address is a static or DHCP-learned IP address.

To see more details specific to DHCP, instead use the **show dhcp lease** command to see the (temporarily) leased IP address and other parameters. (Note that the switch does not store the DHCP-learned IP configuration in the running-config file.) Example 19-4 shows sample output. Note also that the switch learns its default-gateway setting using DHCP as well.

Key Topic

Example 19-4 *Verifying DHCP-Learned Information on a Switch*

```
Emma# show dhcp lease
Temp IP addr: 192.168.1.101 for peer on Interface: Vlan1
Temp sub net mask: 255.255.255.0
 DHCP Lease server: 192.168.1.1, state: 3 Bound
 DHCP transaction id: 1966
 Lease: 86400 secs, Renewal: 43200 secs, Rebind: 75600 secs
Temp default-gateway addr: 192.168.1.1
 Next timer fires after: 11:59:45
 Retry count: 0 Client-ID: cisco-0019.e86a.6fc0-Vl1
 Hostname: Emma

Emma# show ip default-gateway
192.168.1.1
```

Configuring a Router as DHCP Client

Just as with switches, you can configure router interfaces to lease an IP address using DHCP rather than using a static IP address, although those cases will be rare. In most every case it makes more sense to statically configure router interface IP addresses with the address listed in the **ip address** *address mask* interface subcommand. However, configuring a router to lease an address using DHCP makes sense in some cases with a router connected to the Internet; in fact, most every home-based router does just that.

A router with a link to the Internet can learn its IP address and mask with DHCP and also learn the neighboring ISP router's address as the **default gateway**. Figure 19-5 shows an example, with three routers on the left at one enterprise site. Router R1 uses DHCP to learn its IP address (192.0.2.2) from the ISP router over a connection to the Internet.

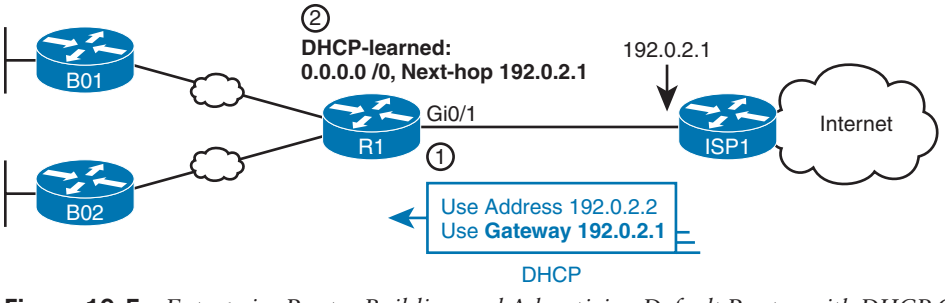

Figure 19-5 *Enterprise Router Building and Advertising Default Routes with DHCP Client*

The DHCP process supplies a default gateway IP address to Router R1, but routers do not normally use a default gateway setting; only hosts use a default gateway setting. However, the router takes advantage of that information by turning that default gateway IP address into the basis for a default route. For instance, in Figure 19-5, Router R1 dynamically adds a default route to its routing table with the default gateway IP address from the DHCP message—which is the ISP router's IP address—as the next-hop address. At that point, R1 has a good route to use to forward packets into the Internet.

Additionally, Router R1 can distribute that default route to the rest of the routers using an interior routing protocol like OSPF. See the section "OSPF Default Routes" in Chapter 23, "Implementing Optional OSPF Features," for more information.

Example 19-5 shows the configuration on Router R1 to match Figure 19-5. Note that it begins with R1 configuring its G0/1 interface to use DHCP to learn the IP address to use on the interface, using the **ip address dhcp** command.

Example 19-5 *Learning an Address and Default Static Route with DHCP*

```
R1# configure terminal
R1(config)# interface gigabitethernet0/1
R1(config-if)# ip address dhcp
R1(config-if)# end
R1#
R1# show ip route static
! Legend omitted
Gateway of last resort is 192.0.2.1 to network 0.0.0.0

S* 0.0.0.0/0 [254/0] via 192.0.2.1
```

The end of the example shows the resulting default route. Oddly, IOS displays this route as a static route (destination 0.0.0.0/0), although the route is learned dynamically based on the DHCP-learned default gateway. To recognize this route as a DHCP-learned default route, look to the administrative distance value of 254. IOS uses a default administrative distance of 1 for static routes configured with the **ip route** configuration command but a default of 254 for default routes added because of DHCP.

Identifying Host IPv4 Settings

Whether learned using DHCP or not, every host that uses IP version 4 needs to have some settings to work correctly. This second major section of the chapter examines those settings and shows examples of those settings on Windows, Linux, and macOS.

Host Settings for IPv4

To work correctly, an IPv4 host needs to know these values:

Key Topic

- DNS server IP addresses
- Default gateway (router) IP address
- Device's own IP address
- Device's own subnet mask

To review the basics, the host must know the IP address of one or more DNS servers to send the servers' name resolution requests. For enterprises, the servers may reside in the enterprise, as shown in Figure 19-6. The host on the left (sometimes called an endpoint) typically knows the addresses of at least two DNS servers for redundancy. If the first DNS fails to respond, the endpoint can then attempt name resolution with the next DNS server.

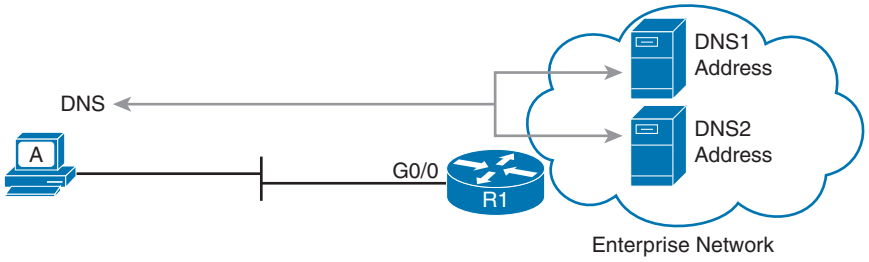

Figure 19-6 *Host A Needs to Know the IP Address of the DNS Servers*

Each endpoint needs to know the IP address of a router that resides in the same subnet. The endpoint uses that router as its default router or default gateway, as shown in Figure 19-7. From a host logic perspective, the host can then forward packets destined for addresses outside the subnet to the default router, with that router then forwarding the packet based on its routing table.

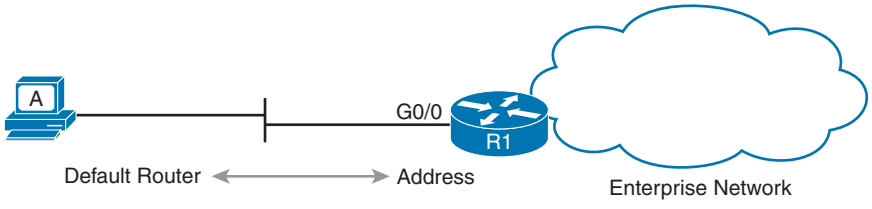

Figure 19-7 *Host Default Router Setting Should Equal Router Interface Address*

Of course, each device needs its own IP address and subnet mask. Equally as important, note that the host and the default router must agree on the addresses inside the subnet. The host will use the address and mask to do the math to determine which addresses are in the same subnet and which are in other subnets. For routing to work correctly, the default router's interface address and mask should result in the same definition of the subnet with the same addresses, as shown in Figure 19-8.

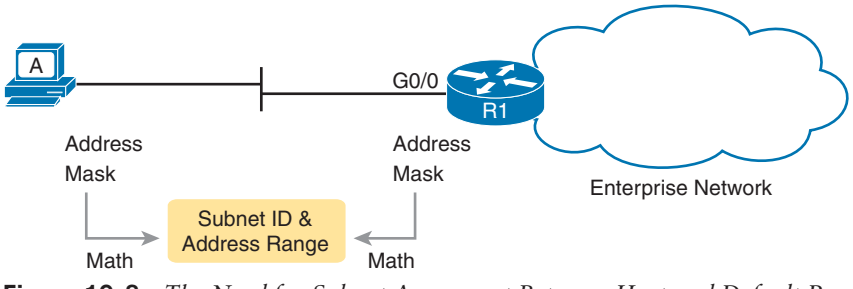

Figure 19-8 *The Need for Subnet Agreement Between Host and Default Router*

The rest of this section shows examples of these settings in the graphical user interface (GUI) and command-line interface (CLI) of three different host operating systems.

The examples for the remainder of the chapter use hosts in subnet 10.1.1.0/24, as shown on the left side of Figure 19-9.

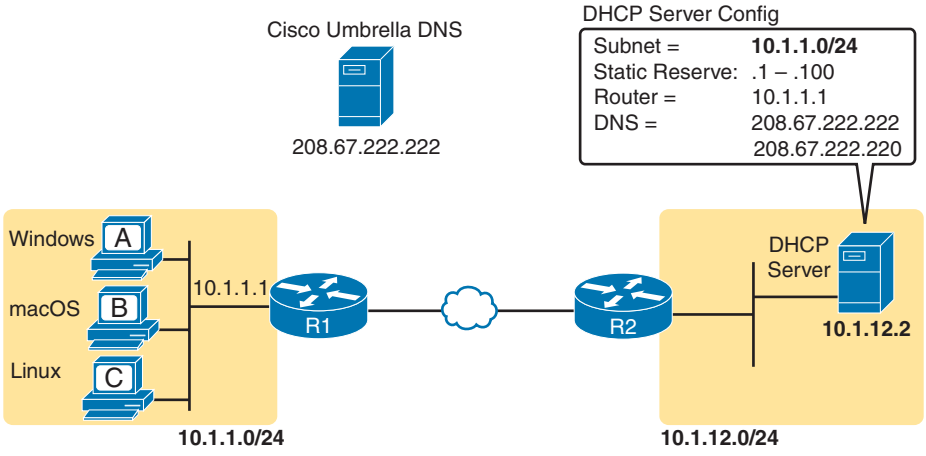

Figure 19-9 *Subnet 10.1.1.0/24 Used for the Upcoming Host Examples*

Host IP Settings on Windows

Almost every OS in the world—certainly the more common OSs people work with every day—have a fairly easy-to-reach settings window that lists most if not all the IPv4 settings in one place.

For example, Figure 19-10 shows the Network configuration screen from a Windows host from the network area of the Windows Control Panel. This image shows the four significant settings: address, mask, router, and DNS. Note that this host, Windows Host A in Figure 19-9, uses DHCP to learn its settings from the DHCP server in that figure.

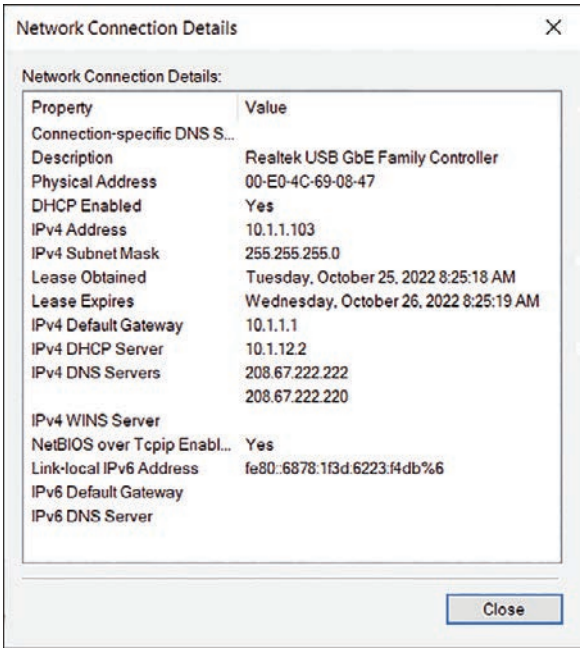

Figure 19-10 *Windows DHCP Client IP Address, Mask, and Default Router Settings*

However, beyond the GUI, most OSs have a variety of networking commands available from a command line. For many decades and Windows versions, the **ipconfig** and **ipconfig /all** commands supply the most direct help, as shown in Example 19-6. As you can see, both list the address, mask, and default gateway. The **ipconfig /all** command lists much more detail, including the **DNS server list**, confirmation that the host used DHCP, along with the IP address of the DHCP server.

Key Topic

Example 19-6 **ipconfig** *and* **ipconfig /all** *(Windows)*

```
Windows_A> ipconfig
! Lines omitted for brevity
Ethernet adapter Ethernet 5:

   Connection-specific DNS Suffix . :
   Link-local IPv6 Address . . . . . : fe80::6878:1f3d:6223:f4db%6
   IPv4 Address. . . . . . . . . . . : 10.1.1.103
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.1.1.1

Windows_A> ipconfig /all
! Lines omitted for brevity
Ethernet adapter Ethernet 5:

   Connection-specific DNS Suffix .  :
   Description . . . . . . . . . . . : Realtek USB GbE Family Controller
   Physical Address. . . . . . . . . : 00-E0-4C-69-08-47
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::6878:1f3d:6223:f4db%6(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.1.1.103(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Tuesday, October 25, 2022 8:25:19 AM
   Lease Expires . . . . . . . . . . : Wednesday, October 26, 2022 8:25:19 AM
   Default Gateway . . . . . . . . . : 10.1.1.1
   DHCP Server . . . . . . . . . . . : 10.1.12.2
   DHCPv6 IAID . . . . . . . . . . . : 100720716
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-19-A0-DC-F0-24-FD-52-CB-50-C9
   DNS Servers . . . . . . . . . . . : 208.67.222.222
                                       208.67.222.220
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

While Example 19-6 shows the traditional **ipconfig** Windows command, Example 19-7 shows a similar command as part of the Windows network shell. The network shell commands replace some of the older Windows command-line commands like **ipconfig**.

Example 19-7 *Windows Network Shell* **show config** *Command*

```
Windows_A> netsh interface ip show config

Configuration for interface "Ethernet 5"
   DHCP enabled:                         Yes
   IP Address:                           10.1.1.103
   Subnet Prefix:                        10.1.1.0/24 (mask 255.255.255.0)
   Default Gateway:                      10.1.1.1
   Gateway Metric:                       0
   InterfaceMetric:                      25
   DNS servers configured through DHCP:  208.67.222.222
                                         208.67.222.220
   Register with which suffix:           Primary only
   WINS servers configured through DHCP: None

! Next, use the network shell to allow repeated commands.
Windows_A> netsh
netsh> interface ip
netsh interface ip> show config

Configuration for interface "Ethernet 5"
   DHCP enabled:                         Yes
   IP Address:                           10.1.1.103
   Subnet Prefix:                        10.1.1.0/24 (mask 255.255.255.0)
! Remaining lines omitted; same output as at the top of the example.

netsh interface ip> show dnsservers

Configuration for interface "Ethernet 5"
   DNS servers configured through DHCP:  208.67.222.222
                                         208.67.222.220
   Register with which suffix:           Primary only
```

Interestingly, you can issue network shell commands as a single command (as seen at the top of Example 19-7) or by navigating within the network shell interface. The example shows the use of the **netsh** command and then **interface ip** so that you reach a mode where you can use short versions of many IP-related commands, like the **show config** and **show dnsservers** commands. (Note that all output has been edited to show only the entries related to the Ethernet interface used in the example.)

You should also look at the host's IP routing table, focusing on two routes that direct most routing in the host. When using DHCP, the host creates these routes based on information learned from the DHCP server:

- A default route, with the default gateway as the next-hop address, with the destination subnet and mask listed as 0.0.0.0 and 0.0.0.0.
- A route to the subnet connected to the working interface, subnet 10.1.1.0/24 in this case, calculated from the interface IP address and mask.

Example 19-8 displays Windows host A's routing table with two commands: the older **netstat -rn** command and the newer **netsh interface ip show route** command. The example has been edited to list only routes related to subnet 10.1.1.0/24 and the Ethernet LAN interface used in the example. Note that a gateway of "on-link" means that the PC thinks the destination is on the local subnet (link).

Key Topic

Example 19-8 *Windows Host IPv4 Routes*

```
Windows_A> netstat -rn
IPv4 Route Table
===========================================================================
Active Routes:
Network Destination         Netmask    Gateway    Interface  Metric
          0.0.0.0           0.0.0.0    10.1.1.1   10.1.1.103     25
         10.1.1.0     255.255.255.0    On-link    10.1.1.103    281
       10.1.1.103   255.255.255.255    On-link    10.1.1.103    281
       10.1.1.255   255.255.255.255    On-link    10.1.1.103    281
! Lines omitted for brevity
Windows_A> netsh interface ip show route
Publish Type      Met Prefix                    Idx Gateway/Interface Name
------- --------- --- ------------------------ --- -------------------------
No      Manual    0   0.0.0.0/0                6 10.1.1.1
No      System    256 10.1.1.0/24              6 Ethernet 5
No      System    256 10.1.1.103/32            6 Ethernet 5
No      System    256 10.1.1.255/32            6 Ethernet 5
!       Lines     omitted for brevity
```

Host IP Settings on macOS

Although the details vary as compared to Windows, macOS has a graphical interface to see network settings and a variety of network commands. This section shows examples of each, beginning with Figure 19-11. It shows the network settings in macOS for an Ethernet interface, with the address, mask, default router, and DNS server addresses. Also note the setting states that the interface is using DHCP.

Both macOS and Linux support the **ifconfig** command to list information similar to the Windows **ipconfig /all** command. However, the **ifconfig** command lacks an option like **/all**, listing no information about the default gateway or DNS servers. Example 19-9 gives an example of the **ifconfig** command from Mac Host B in Figure 19-11, along with another command that lists the default router.

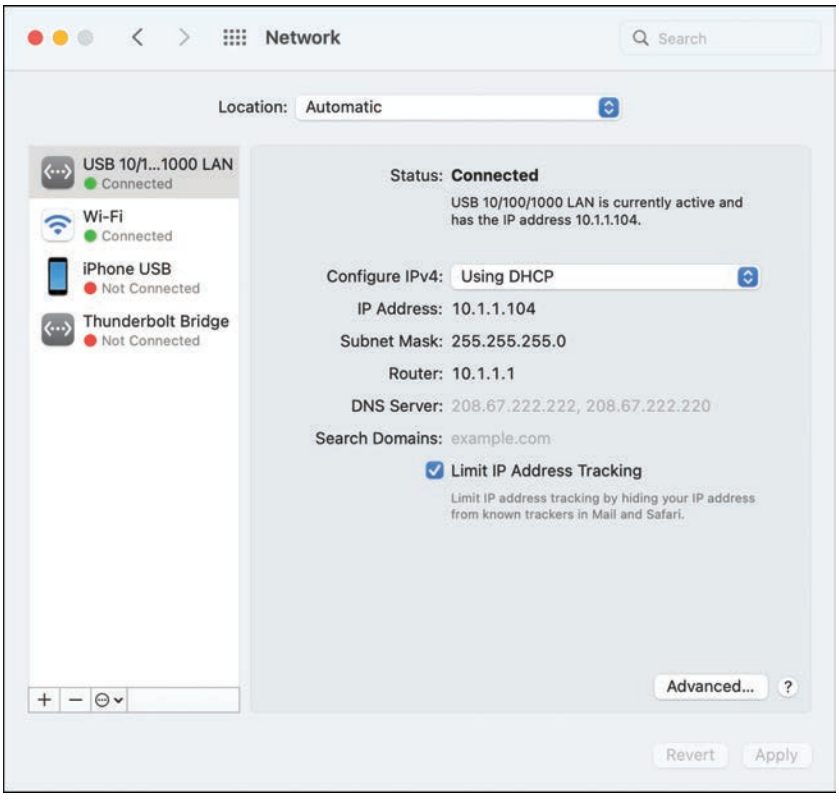

Figure 19-11 *macOS DHCP Client IP Address, Mask, and Default Router Settings*

Key Topic

Example 19-9 ifconfig *and* networksetup -getinfo *(macOS)*

```
Mac_B$ ifconfig en8
 en8: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
       options=6467<RXCSUM,TXCSUM,VLAN_MTU,TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,
       ZEROINVERT_CSUM>
          ether 00:e0:4c:68:1f:26
          inet6 fe80::184c:56f9:fd3b:d6e7%en8 prefixlen 64 secured scopeid 0x14
          inet 10.1.1.104 netmask 0xffffff00 broadcast 10.1.1.255
          nd6 options=201<PERFORMNUD,DAD>
          media: autoselect (1000baseT <full-duplex>)
          status: active

Mac_B$ networksetup -getinfo "USB 10/100/1000 LAN"
DHCP Configuration
IP address: 10.1.1.104
Subnet mask: 255.255.255.0
Router: 10.1.1.1
Client ID:
```

```
IPv6: Automatic
IPv6 IP address: none
IPv6 Router: none
Ethernet Address: 00:e0:4c:68:1f:26
```

Focusing on the **ifconfig** command, it lists the subnet mask in hex, listing 0xffffff00 in the example. To convert the mask to dotted decimal, convert each pair of hex digits to binary and then from binary to the decimal equivalent, giving you four octets for the dotted-decimal mask. In this case, the 0xffffff00 begins with three pairs of ff, which convert to binary 11111111. Each binary 11111111 converts to decimal 255: the last hex pair, 00, converts to binary 00000000, which converts to decimal 0. The final decimal mask is the familiar value 255.255.255.0.

The **networksetup** command in Example 19-9 lists more detail than the older **ifconfig** command. It lists the usual IP settings, other than the DNS server list. Note that the **networksetup** macOS command has a large variety of options.

Like Windows, macOS adds a default route to its host routing table based on the default gateway and a route to the local subnet calculated based on the IP address and mask. And like Windows, macOS uses the **netstat -rn** command to list those routes—but with several differences in the output. Of note, in the macOS sample shown in Example 19-10, the output represents the default route using the word *default* rather than the destination subnet and mask of 0.0.0.0 and 0.0.0.0. The example highlights the default route and the route to the connected subnet. As usual, the example aids readability by limiting the output to those routes related to network 10.0.0.0.

Example 19-10 **netstat -rn** *Command (macOS)*

```
Mac_B> netstat -rn
Routing tables

Internet:
Destination      Gateway            Flags        Netif Expire
default          10.1.1.1           UGScg        en8
10.1.1/24        link#20            UCS          en8         !
10.1.1.1/32      link#20            UCS          en8         !
10.1.1.1         2:0:11:11:11:11    UHLWIir      en8       385
10.1.1.104/32    link#20            UCS          en8         !
! lines omitted for brevity
```

Host IP Settings on Linux

As with the other desktop OSs, Linux shows networking settings from the GUI as well as with commands. However, be aware that the Linux world includes many different Linux distributions. Additionally, the Linux architecture separates the OS from the desktop (the graphical interface). So, other Linux users may use different GUI administration tools and commands to see network settings.

Figure 19-12 shows an example from the Ubuntu Linux distribution using the Mate desktop (www.ubuntu-mate.org). The host, Host C in Figure 19-9, uses DHCP and learns an address

in subnet 10.1.1.0/24. Figure 19-12 shows IP details about an Ethernet adapter and includes the IPv4 address, mask, default router, and DNS IP addresses.

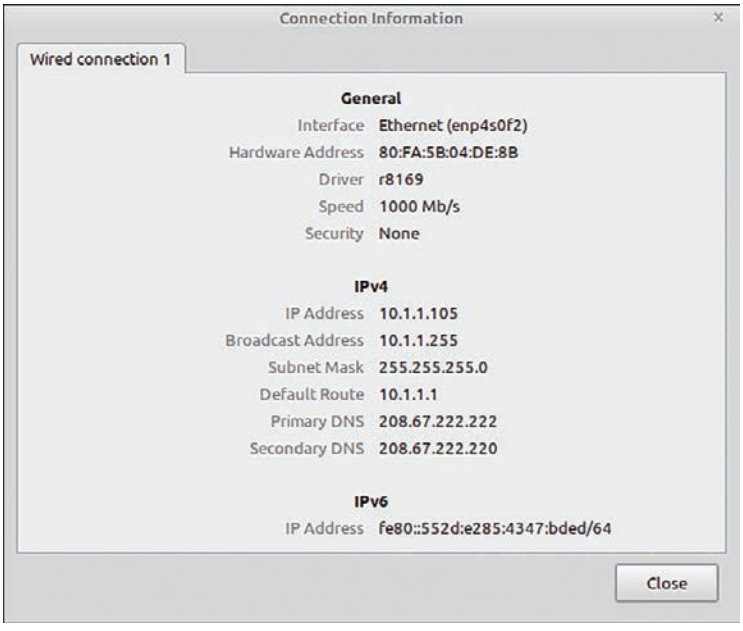

Figure 19-12 *Linux DHCP Client IP Address, Mask, and Default Router Settings*

Linux hosts often support a large set of commands from the command line. However, an older set of commands, referenced together as *net-tools*, has been deprecated in Linux to the point that some Linux distributions do not include net-tools. (You can easily add net-tools to most Linux distributions.) The net-tools library has popular commands like **ifconfig** and **netstat -rn**. To replace those tools, Linux uses the *iproute* library, which includes a set of replacement commands and functions, many performed with the **ip** command and some parameters.

Example 19-11 shows a sample of the Linux **ifconfig** command and the replacement command **ip address** for the same interface detailed in Figure 19-12. Both commands list the Ethernet MAC and IPv4 addresses, along with the subnet mask.

Example 19-11 **ifconfig** *and* **ip address** *Commands (Linux)*

```
Linux_C$ ifconfig
enp4s0f2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.1.1.105 netmask 255.255.255.0 broadcast 10.1.1.255
        inet6 fe80::552d:e285:4347:bded prefixlen 64 scopeid 0x20<link>
        ether 80:fa:5b:04:de:8b txqueuelen 1000 (Ethernet)
        RX packets 4958 bytes 384498 (384.4 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 16270 bytes 1320108 (1.3 MB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
Linux_C$ ip address
2: enp4s0f2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 80:fa:5b:04:de:8b brd ff:ff:ff:ff:ff:ff
    inet 10.1.1.105/24 brd 10.1.1.255 scope global dynamic noprefixroute enp4s0f2
      valid_lft 81056sec preferred_lft 81056sec
    inet6 fe80::552d:e285:4347:bded/64 scope link noprefixroute
      valid_lft forever preferred_lft forever
```

Example 19-12 shows another pair of older and newer Linux commands: the **netstat -rn** and **ip route** commands. The **netstat -rn** command lists the default route with a style that shows the destination and mask both as 0.0.0.0. As usual, the default route uses the learned default gateway IP address (10.1.1.1) as the next-hop IP address. The **netstat -rn** output also lists a route to the local subnet (10.1.1.0/24). The **ip route** command lists those same routes but with much different formatting and phrasing.

Example 19-12 **netstat -rn** *and* **ip route** *Commands (Linux)*

```
Linux_C$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         10.1.1.1        0.0.0.0         UG        0 0          0 enp4s0f2
10.1.1.0        0.0.0.0         255.255.255.0   U         0 0          0 enp4s0f2
! Lines omitted for brevity

Linux_C$ ip route
default via 10.1.1.1 dev enp4s0f2 proto dhcp metric 20100
10.1.1.0/24 dev enp4s0f2 proto kernel  scope link  src 10.1.1.105 metric 100
! Lines omitted for brevity
```

Troubleshooting Host IP Settings

The examples for Windows, macOS, and Linux shown over the last few pages show the successful results for DHCP clients A, B, and C per Figure 19-9. To round out your perspective, the final few examples in this section show a few different scenarios with the resulting output from the popular Windows **ipconfig /all** command. The scenarios, all from Windows Host A in the figure, are as follows:

1. Host A with static IP configuration, using correct settings.
2. Host A as a DHCP client, but with a network issue that prevents the DHCP client and server from communicating.
3. Host A as a DHCP client with a working network so that DHCP works, but with the DHCP server supplying some incorrect information (wrong default gateway address).

A Working Windows Host with Static IP Configuration

Example 19-13 shows the first scenario. Someone has disabled DHCP on Windows Host A and configured all IP settings manually. They chose well, using correct values per

Figure 19-9. In particular, the figure shows addresses .1 – .100 reserved for static addresses, so the person who manually configured Host A assigned 10.1.1.99.

Key Topic

Example 19-13 *Working Windows Host with Static IP Setting (No DHCP)*

```
Windows_A> ipconfig /all
! Showing only the Ethernet adapter connected to subnet 10.1.1.0/24
Ethernet adapter Ethernet 5:

   Connection-specific DNS Suffix .  :
   Description . . . . . . . . . . . : Realtek USB GbE Family Controller
   Physical Address. . . . . . . . . : 00-E0-4C-69-08-47
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::6878:1f3d:6223:f4db%6(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.1.1.99(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.1.1.1
   DHCPv6 IAID . . . . . . . . . . . : 100720716
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-19-A0-DC-F0-24-FD-52-CB-50-C9
   DNS Servers . . . . . . . . . . . : 208.67.222.222
                                       208.67.222.220
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

The **ipconfig /all** command output in Example 19-13 implies that the host uses static settings by noting that DHCP is disabled (per the first highlighted line). Take a moment to compare that line to the same line back in Example 19-7, in which Host A acts as a DHCP client.

The output here in Example 19-13 also signals that the host uses a static setting rather than DHCP, but it tells us that by the absence of several lines of output usually shown for DHCP clients, omitting the lines for

- DHCP Enabled
- Lease Obtained
- Lease Expires

Knowing those facts, you should be able to distinguish between cases in which the Windows host does and does not use DHCP.

19

A Failed Windows DHCP Client Due to IP Connectivity Issues

The following example, again taken from Windows Host A, shows a case in which the host similarly acts as a DHCP client to follow the design from Figure 19-9. However, any problem that prevents the DHCP DORA messages from flowing between the client and DHCP server results in a failure of the DHCP process. The client does not lease an address or learn the other IP details from the DHCP server, but it self-assigns an APIPA address and uses the APIPA default mask of 255.255.0.0. (For a review of APIPA, see the section titled "APIPA IP Addresses (169.254.x.x)" earlier in this chapter.)

Example 19-14 shows the resulting **ipconfig /all** command. The output confirms the client enables DHCP. However, the following facts identify that the process failed:

- It omits the output line that would list the DHCP server's IP address.
- It omits the two output lines that list DHCP lease details.
- It shows an APIPA IP address that begins 169.254.
- It lists no IPv4 default gateway address.
- It lists no IPv4 DNS servers.

Example 19-14 *Windows DHCP Client Fails Due to IP Connectivity Failure*

```
Windows_A> ipconfig /all
! Showing only the Ethernet adapter connected to subnet 10.1.1.0/24
Ethernet adapter Ethernet 5:

   Connection-specific DNS Suffix .  :
   Description . . . . . . . . . . . : Realtek USB GbE Family Controller
   Physical Address. . . . . . . . . : 00-E0-4C-69-08-47
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::6878:1f3d:6223:f4db%6(Preferred)
   Autoconfiguration IPv4 Address. . : 169.254.244.219(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 100720716
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-19-A0-DC-F0-24-FD-52-CB-50-C9
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
                                       fec0:0:0:ffff::3%1
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

A Working Windows DHCP Client with Incorrect Settings

For one final scenario, imagine a case in which the DHCP process works but the server configuration is incorrect. As a result, the DHCP client can learn incorrect information and not be able to communicate in some cases. For instance, the DHCP process could complete with the client learning these inaccurate or incomplete settings:

- Wrong subnet mask
- Wrong or missing default gateway
- Wrong or missing DNS server list

Example 19-15 shows a case with the incorrect default gateway (10.1.1.254) configured in the DHCP server's settings for subnet 10.1.1.0/24. The correct default gateway per the design is Router R1 (10.1.1.1). The **ipconfig /all** output in Example 19-15 shows the usual indications

that DHCP worked, showing it as enabled, with a lease start and end time. It also shows the host's IP address, mask, default gateway, and DNS servers. But compared to the intended design, the output shows that the host learned a default gateway setting for a nonexistent router.

Example 19-15 *Windows DHCP Client Learns Incorrect Default Gateway*

```
Windows_A> ipconfig /all

Ethernet adapter Ethernet 5:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek USB GbE Family Controller
   Physical Address. . . . . . . . . : 00-E0-4C-69-08-47
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::6878:1f3d:6223:f4db%6(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.1.1.103(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Tuesday, October 25, 2022 7:54:17 AM
   Lease Expires . . . . . . . . . . : Wednesday, October 26, 2022 7:54:16 AM
   Default Gateway . . . . . . . . . : 10.1.1.254
   DHCP Server . . . . . . . . . . . : 10.1.12.2
   DHCPv6 IAID . . . . . . . . . . . : 100720716
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-19-A0-DC-F0-24-FD-52-CB-50-C9
   DNS Servers . . . . . . . . . . . : 208.67.222.222
                                       208.67.222.220
   NetBIOS over Tcpip. . . . . . . . : Enabled

Windows_A> netstat -rn

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination          Netmask      Gateway     Interface  Metric
          0.0.0.0            0.0.0.0   10.1.1.254     10.1.1.103     25
         10.1.1.0      255.255.255.0      On-link     10.1.1.103    281
! Lines omitted for brevity
```

The underlying solution to this problem is to fix the configuration mistake in the DHCP server. For the exam, the more likely scenario would be for you to think about the design and notice a difference between what the DHCP client learned and the correct settings. Also, note that the next chapter, "Troubleshooting IPv4 Routing," gives some insights into how to explore different symptoms for problems in an IP network.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 19-2 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 19-2 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website
Answer DIKTA questions		Book, PTP
Review command tables		Book
Watch video		Website

Review All the Key Topics

Key Topic

Table 19-3 Key Topics for Chapter 19

Key Topic Element	Description	Page Number
List	Definitions of special IPv4 addresses 0.0.0.0 and 255.255.255.255	489
List	Four logic steps created by the **ip helper-address** command	491
Figure 19-2	What the **ip helper-address** command changes in a DHCP Discover message	491
List	The two facts that must be true about a subnet for a router to need to be a DHCP relay agent for that subnet	494
Example 19-4	Switch commands that confirm the details of DHCP client operations based on the **ip address dhcp** interface subcommand	496
List	The IPv4 settings expected on an end-user host	497
Example 19-6	Output from a Windows **ipconfig /all** command when the host successfully uses DHCP	500
Example 19-8	IP routes on a Windows host	502
Example 19-9	Output from a macOS **ifconfig** command plus two **networksetup** commands	503
Example 19-13	Output from a Windows **ipconfig /all** command when the host uses static IP configuration	507

Key Terms You Should Know

APIPA, default gateway, DHCP client, DHCP relay agent, DHCP server, DNS server, DNS server list

Command References

Tables 19-4, 19-5, and 19-6 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 19-4 Chapter 19 Configuration Command Reference

Command	Description
ip helper-address *IP-address*	An interface subcommand that tells the router to notice local subnet broadcasts (to 255.255.255.255) that use UDP, and change the source and destination IP address, enabling DHCP servers to sit on a remote subnet
ip address dhcp	An interface subcommand that tells the router or switch to use DHCP to attempt to lease a DHCP address from a DHCP server

Table 19-5 Chapter 19 EXEC Command Reference

Command	Description
show arp, **show ip arp**	Command that lists the router's IPv4 ARP table
show dhcp lease	Switch command that lists information about addresses leased because of the configuration of the **ip address dhcp** command
show ip default-gateway	Switch command that lists the switch's default gateway setting, no matter whether learned by DHCP or statically configured

Table 19-6 Chapter 19 Generic Host Networking Command Reference

Command	Description
ipconfig /all	(Windows) Lists IP address, mask, gateway, and DNS servers
ifconfig	(Mac, Linux) Lists IP address and mask for an interface
networksetup -getinfo *interface*	(Mac) Lists IP settings including default router
netstat -rn	(Windows, Mac, Linux) Lists the host's routing table, including a default route that uses the DHCP-learned default gateway
arp -a	(Windows, Mac, Linux) Lists the host's ARP table
ip address	(Linux) Lists IP address and mask information for interfaces; the Linux replacement for **ifconfig**
ip route	(Linux) Lists routes, including the default route and a route to the local subnet; the Linux replacement for **netstat -rn**
netsh interface ip show addresses	(Windows) Windows network shell command to list interface IP address configuration settings; a replacement for the **ipconfig /all** command
netsh interface ip show route	(Windows) Windows network shell command to list IPv4 routes; a replacement for the **netstat -rn** command

CHAPTER 20

Troubleshooting IPv4 Routing

This chapter covers the following exam topics:

- **2.0 Network Access**
  - **2.8 Describe network device management access (Telnet, SSH, HTTP, HTTPS, console, TACACS+/RADIUS, and cloud managed)**
- **3.0 IP Connectivity**
  - **3.3 Configure and verify IPv4 and IPv6 static routing**
    - **3.3.a Default route**
    - **3.3.b Network route**
    - **3.3.c Host route**
    - **3.3.d Floating static**

This chapter turns our attention to routing from end-to-end across an entire enterprise network. How do you troubleshoot an IPv4 network? How do you verify correct operation, identify root causes, and fix those for various IP routing features? How do you do that in the presence of an IP addressing and subnetting plan, requiring you to apply all that subnetting math from Part IV of this book and the basic address/mask and static route configuration from the other chapters here in Part V? This chapter answers some of those questions.

In particular, this chapter focuses on two tools and how to use them: *ping* and *traceroute*. Both tools test the IPv4 data plane; that is, the ability of each networking device to route or forward IPv4 packets. This chapter devotes a major section each to **ping** and **traceroute**. The chapter then ends with a short discussion of two other router tools that can also be useful for troubleshooting: Telnet and Secure Shell (SSH).

"Do I Know This Already?" Quiz

I put DIKTA quizzes in most of the chapters as a tool to help you decide how to approach reading a chapter. However, this chapter does not have a DIKTA quiz because I think you should read it regardless of your prior knowledge. As with all chapters in this book, this chapter introduces new concepts, but it also acts as a tool to review and deepen your understanding of IP routing. Hope you enjoy the perspectives on using **ping** and **traceroute** in this chapter.

Foundation Topics

Problem Isolation Using the ping Command

Someone sends you an email or text, or a phone message, asking you to look into a user's network problem. You Secure Shell (SSH) to a router and issue a **ping** command that works. What does that result rule out as a possible reason for the problem? What does it rule in as still being a possible root cause?

Then you issue another **ping** to another address, and this time the ping fails. Again, what does the failure of that **ping** command tell you? What parts of IPv4 routing may still be a problem, and what parts do you now know are not a problem?

The **ping** command gives us one of the most common network troubleshooting tools. When the **ping** command succeeds, it confirms many individual parts of how IP routing works, ruling out some possible causes of the current problem. When a **ping** command fails, it often helps narrow down where in the internetwork the root cause of the problem may be happening, further isolating the problem.

This section begins with a brief explanation of how ping works. It then moves on to some suggestions and analysis of how to use the **ping** command to isolate problems by removing some items from consideration.

Ping Command Basics

The **ping** command tests connectivity by sending packets to an IP address, expecting the device at that address to send packets back. The command sends packets that mean "if you receive this packet, and it is addressed to you, send a reply back." Each time the **ping** command sends one of these packets and receives the message sent back by the other host, the **ping** command knows a packet made it from the source host to the destination and back.

More formally, the **ping** command uses the Internet Control Message Protocol (ICMP), specifically the **ICMP echo request** and **ICMP echo reply** messages. ICMP defines many other messages as well, but these two messages were made specifically for connectivity testing by commands like **ping**. As a protocol, ICMP does not rely on TCP or UDP, and it does not use any application layer protocol. It functions as part of Layer 3, as a control protocol to assist IP by helping manage the IP network functions.

Figure 20-1 shows the ICMP messages, with IP headers, in an example. In this case, the user at host A opens a command prompt and issues the **ping 172.16.2.101** command, testing connectivity to host B. The command sends one echo request and waits (Step 1); host B receives the messages and sends back an echo reply (Step 2).

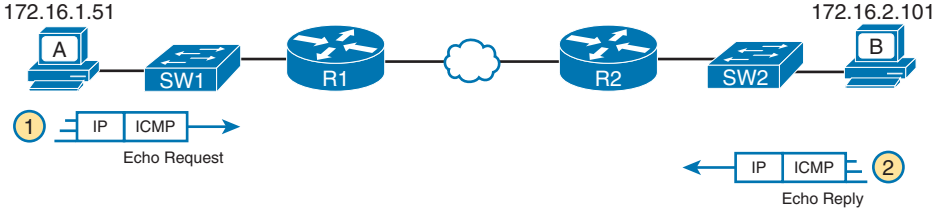

Figure 20-1 *Concept Behind* **ping 172.16.2.101** *on Host A*

The **ping** command is supported on many different devices and many common operating systems. The command has many options: the name or IP address of the destination, how many times the command should send an echo request, how long the command should wait (timeout) for an echo reply, how big to make the packets, and many other options. Example 20-1 shows a sample from host A, with the same command that matches the concept in Figure 20-1: a **ping 172.16.2.101** command on host A.

Example 20-1 *Sample Output from Host A's* **ping 172.16.2.101** *Command*

```
Mac_A$ ping 172.16.2.101
PING 172.16.2.101 (172.16.2.101): 56 data bytes
64 bytes from 172.16.2.101: icmp_seq=0 ttl=64 time=1.112 ms
64 bytes from 172.16.2.101: icmp_seq=1 ttl=64 time=0.673 ms
64 bytes from 172.16.2.101: icmp_seq=2 ttl=64 time=0.631 ms
64 bytes from 172.16.2.101: icmp_seq=3 ttl=64 time=0.674 ms
64 bytes from 172.16.2.101: icmp_seq=4 ttl=64 time=0.642 ms
64 bytes from 172.16.2.101: icmp_seq=5 ttl=64 time=0.656 ms
^C
--- 172.16.2.101 ping statistics ---
6 packets transmitted, 6 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.631/0.731/1.112/0.171 ms
```

Strategies and Results When Testing with the ping Command

Often, the person handling initial calls from users about problems (often called a customer support rep, or CSR) cannot issue **ping** commands from the user's device. In some cases, talking users through typing the right commands and making the right clicks on their machines can be a problem. Or the user just might not be available. As an alternative, using different **ping** commands from different routers can help isolate the problem.

The problem with using **ping** commands from routers, instead of from the host that has the problem, is that no single router **ping** command can exactly replicate a **ping** command done from the user's device. However, each different **ping** command can help isolate a problem further. The rest of this section of **ping** commands discusses troubleshooting IPv4 routing by using various **ping** commands from the command-line interface (CLI) of a router.

Testing Longer Routes from Near the Source of the Problem

Most problems begin with some idea like "host X cannot communicate with host Y." A great first troubleshooting step is to issue a **ping** command from X for host Y's IP address. However, assuming the engineer does not have access to host X, the engineer can instead issue the **ping** from the router nearest X, typically the router acting as host X's default gateway.

For instance, in Figure 20-1, imagine that the user of host A had called IT support with a problem related to sending packets to host B. A **ping 172.16.2.101** command on host A would be a great first troubleshooting step, but the CSR cannot access host A or get in touch with the user of host A. So, the CSR telnets to Router R1 and pings host B from there, as shown in Example 20-2.

Example 20-2 *Router R2 Pings Host B (Two Commands)*

```
R1# ping 172.16.2.101
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms
R1# ping 172.16.2.101
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

First, take a moment to review the output of the first IOS **ping** command. By default, the Cisco IOS **ping** command sends five echo messages, with a timeout of 2 seconds. If the command does not receive an echo reply within 2 seconds, the command considers that message to be a failure, and the command lists a period. If a successful reply is received within 2 seconds, the command displays an exclamation point. So, in this first command, the first echo reply timed out, whereas the other four received a matching echo reply within 2 seconds.

As a quick aside, the example shows a common and normal behavior with **ping** commands: the first **ping** command shows one failure to start, but then the rest of the messages work. This usually happens because some device in the end-to-end route is missing an ARP table entry.

Now think about troubleshooting and what a working **ping** command tells us about the current behavior of this internetwork. First, focus on the big picture for a moment:

- R1 can send ICMP echo request messages to host B (172.16.2.101).
- R1 sends these messages from its outgoing interface's IP address (by default), 172.16.4.1 in this case.
- Host B can send ICMP echo reply messages to R1's 172.16.4.1 IP address (hosts send echo reply messages to the IP address from which the echo request was received).

Figure 20-2 shows the packet flow.

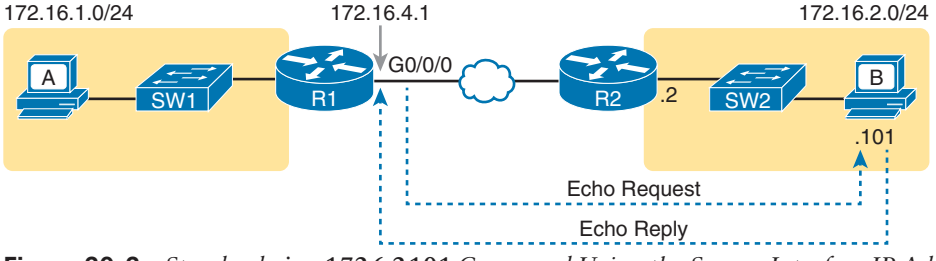

Figure 20-2 *Standard* **ping 172.6.2.101** *Command Using the Source Interface IP Address*

Next, think about IPv4 routing. In the forward direction, R1 must have a route that matches host B's address (172.16.2.101); this route will be either a static route or one learned with a

routing protocol. R2 also needs a route for host B's address, in this case a connected route to B's subnet (172.16.2.0/24), as shown in the top arrow lines in Figure 20-3.

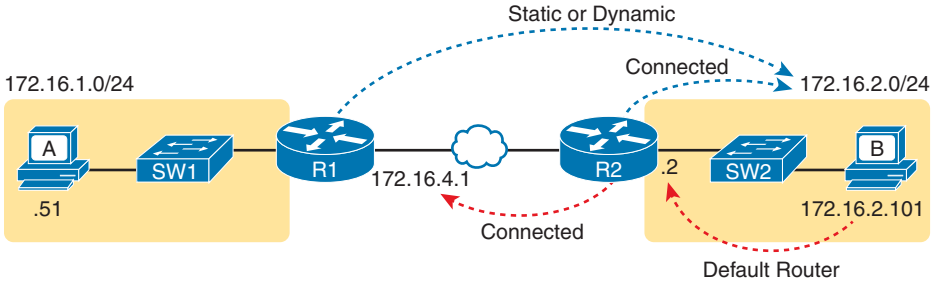

Figure 20-3 *Layer 3 Routes Needed for R1's* **ping 172.16.2.101** *to Work*

The arrow lines on the bottom of Figure 20-3 show the routes needed to forward the ICMP echo reply message back to Router R1's 172.16.4.1 interface. First, host B must have a valid default router setting because 172.16.4.1 sits in a different subnet than host B. R2 must also have a route that matches destination 172.16.4.1 (in this case, likely to be a connected route).

The working **ping** commands in Example 20-2 also require the data-link and physical layer details to be working. The WAN link must be working: The router interfaces must be up/up, which typically indicates that the link can pass data. On the LAN, R2's LAN interface must be in an up/up state. In addition, everything discussed about Ethernet LANs must be working because the **ping** confirmed that the packets went all the way from R1 to host B and back. In particular

- The switch interfaces in use are in a connected (up/up) state.
- Port security (discussed in the *CCNA 200-301 Official Cert Guide, Volume 2*, Second Edition) does not filter frames sent by R2 or host B.
- STP has placed the right ports into a forwarding state.

The **ping 172.16.2.101** command in Example 20-2 also confirms that IP access control lists (ACLs) did not filter the ICMP messages. One ACL contains a set of matching rules and actions: some matched packets are filtered (discarded), while others can continue on their path as normal. ACLs can examine packets as they enter or exit a router interface, so Figure 20-4 shows the various locations on routers R1 and R2 where an ACL could have filtered (discarded) the ICMP messages. (Note that an outbound ACL on Router R1 would not filter packets created on R1, so there is no rightward-facing arrow over R1.)

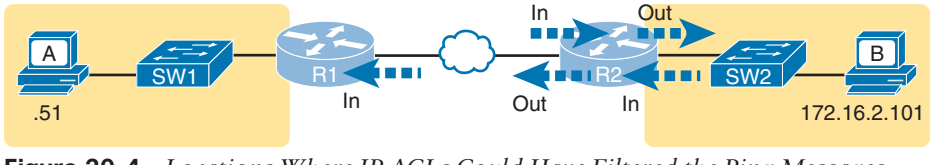

Figure 20-4 *Locations Where IP ACLs Could Have Filtered the Ping Messages*

Finally, the working **ping 172.16.2.101** command on R1 can also be used to reasonably predict that ARP worked and that switch SW2 learned MAC addresses for its MAC address

table. R2 and host B need to know each other's MAC addresses so that they can encapsulate the IP packet inside an Ethernet frame, which means both must have a matching ARP table entry. The switch learns the MAC address used by R2 and by host B when it sends the ARP messages or when it sends the frames that hold the IP packets. Figure 20-5 shows the type of information expected in those tables.

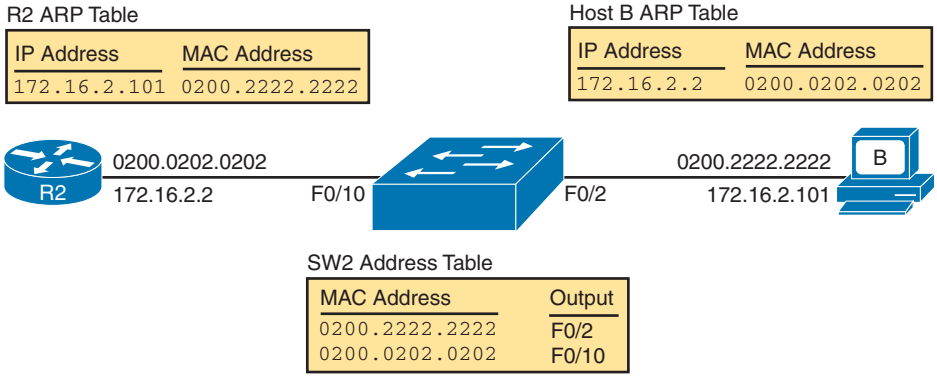

Figure 20-5 *Router and Host ARP Tables, with the Switch MAC Address Table*

As you can see from the last few pages, a strategy of using a **ping** command from near the source of the problem can rule out a lot of possible root causes of any problems between two hosts—assuming the **ping** command succeeds. However, this **ping** command does not act exactly like the same **ping** command on the actual host. To overcome some of what is missing in the **ping** command from a nearby router, the next several examples show some strategies for testing other parts of the path between the two hosts that might have a current problem.

Using Extended Ping to Test the Reverse Route

Pinging from the default router, as discussed in the past few pages, misses an opportunity to test IP routes more fully. Such tests check the **forward route**, that is, the route toward the destination. However, a ping from the default route does not test the **reverse route** back toward the original host.

For instance, referring to the internetwork in Figure 20-2 again, note that the reverse routes do not point to an address in host A's subnet. When R1 processes the **ping 172.16.2.101** command, R1 has to pick a source IP address to use for the echo request, and routers choose the *IP address of the outgoing interface*. The echo request from R1 to host B flows with source IP address 172.16.4.1 (R1's G0/0/0 IP address). The echo reply flows back to that same address (172.16.4.1).

A standard ping often does not test the reverse route that you need to test. In this case, the standard **ping 172.16.2.101** command on R1 does not test whether the routers can route back to subnet 172.16.1.0/24, instead testing their routes for subnet 172.16.4.0. A better ping test would test the route back to host A's subnet; an *extended ping* from R1 can cause that test to happen. An extended ping allows R1's **ping** command to use R1's LAN IP address from within subnet 172.16.1.0/24. Then, the echo reply messages would flow to host A's subnet, as shown in Figure 20-6.

Key Topic

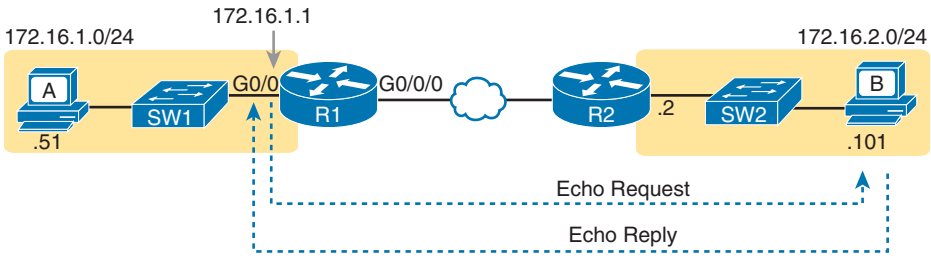

Figure 20-6 *Extended Ping Command Tests the Route to 172.16.1.51 (Host A)*

The extended **ping** command does allow the user to type all the parameters on a potentially long command, but it also allows users to simply issue the **ping** command, press Enter, with IOS then asking the user to answer questions to complete the command, as shown in Example 20-3. The example shows the **ping** command on R1 that matches the logic in Figure 20-6. This same command could have been issued from the command line as **ping 172.16.2.101 source 172.16.1.1**.

Example 20-3 *Testing the Reverse Route Using the Extended Ping*

```
R1# ping
Protocol [ip]:
Target IP address: 172.16.2.101
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 172.16.1.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
Packet sent with a source address of 172.16.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

This particular extended **ping** command tests the same routes for the echo request going to the right, but it forces a better test of routes pointing back to the left for the ICMP echo reply. For that direction, R2 needs a route that matches address 172.16.1.1, which is likely to be a route for subnet 172.16.1.0/24—the same subnet in which host A resides.

From a troubleshooting perspective, using both standard and extended **ping** commands can be useful. However, neither can exactly mimic a **ping** command created on the host itself because the routers cannot send packets with the host's IP address. For instance, the

extended **ping** in Example 20-3 uses source IP address 172.16.1.1, which is not host A's IP address. As a result, neither the standard nor extended **ping** commands in these two examples so far in this chapter can test for some kinds of problems, such as the following:

- IP ACLs that discard packets based on host A's IP address but allow packets that match the router's IP address
- LAN switch port security that filters A's frames (based on A's MAC address)
- IP routes on routers that happen to match host A's 172.16.1.51 address, with different routes that match R1's 172.16.1.1 address
- Problems with host A's default router setting

NOTE IP ACLs and LAN switch port security are covered in the *CCNA 200-301 Official Cert Guide, Volume 2*, Second Edition. For now, know that IP ACLs can filter packets on routers, focusing on the Layer 3 and 4 headers. Port security can be enabled on Layer 2 switches to filter based on source MAC addresses.

Testing LAN Neighbors with Standard Ping

Testing using a **ping** of another device on the LAN can quickly confirm whether the LAN can pass packets and frames. Specifically, a working **ping** rules out many possible root causes of a problem. For instance, Figure 20-7 shows the ICMP messages that occur if R1 issues the command **ping 172.16.1.51**, pinging host A, which sits on the same VLAN as R1.

Key Topic

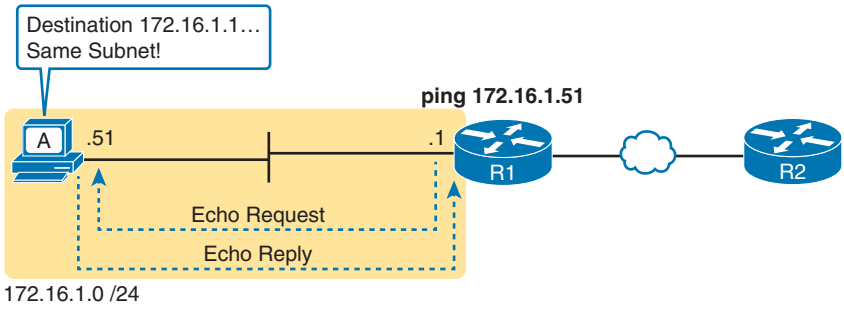

Figure 20-7 *Standard* **ping** *Command Confirms That the LAN Works*

If the ping works, it confirms the following, which rules out some potential issues:

- The host with address 172.16.1.51 replied.
- The LAN can pass unicast frames from R1 to host 172.16.1.51 and vice versa.
- You can reasonably assume that the switches learned the MAC addresses of the router and the host, adding those to the MAC address tables.
- Host A and Router R1 completed the ARP process and list each other in their respective Address Resolution Protocol (ARP) tables.

The failure of a ping, even with two devices on the same subnet, can point to a variety of problems, like those mentioned in this list. For instance, if the **ping 172.16.1.51** on R1 fails (refer to Figure 20-7), that result points to this list of potential root causes:

- **IP addressing problem:** Host A or the router could be configured with the wrong IP address.
- **IP mask problem:** Using an incorrect subnet mask on either the host or the router would change their calculation view of the range of addresses in the attached subnet, which would affect their forwarding logic. For example, the host, with address 172.16.1.51 but incorrect mask 255.255.255.240, would think that the router's address of 172.16.1.1 is in a different subnet.
- **DHCP problems:** If you are using Dynamic Host Configuration Protocol (DHCP), many problems could exist. Chapter 19, "IP Addressing on Hosts," discusses those possibilities in some depth.
- **VLAN trunking problems:** The router could be configured for 802.1Q trunking, when the switch is not (or vice versa).
- **LAN problems:** A wide variety of issues could exist with the Layer 2 switches, preventing any frames from flowing between host A and the router.

So, whether the ping works or fails, simply pinging a LAN host from a router can help further isolate the problem.

Testing LAN Neighbors with Extended Ping

A standard ping of a LAN host from a router does not test that host's default router setting. However, an extended ping can test the host's default router setting. Both tests can be useful, especially for problem isolation, because

- If a standard ping of a local LAN host works…
- But an extended ping of the same LAN host fails…
- The problem likely relates somehow to the host's default router setting.

First, to understand why the standard and extended ping results have different effects, consider first the standard **ping 172.16.1.51** command on R1, as shown previously in Figure 20-7. As a standard **ping** command, R1 used its LAN interface IP address (172.16.1.1) as the source of the ICMP Echo. So, when the host (A) sent back its ICMP echo reply, host A considered the destination of 172.16.1.1 as being on the same subnet. Host A's ICMP echo reply message, sent back to 172.16.1.1, would work even if host A did not have a default router setting at all!

In comparison, Figure 20-8 shows the difference when using an extended ping on Router R1. An extended ping from local Router R1, using R1's WAN IP address of 172.16.4.1 as the source of the ICMP echo request, means that host A's ICMP echo reply will flow to an address in another subnet, which makes host A use its default router setting.

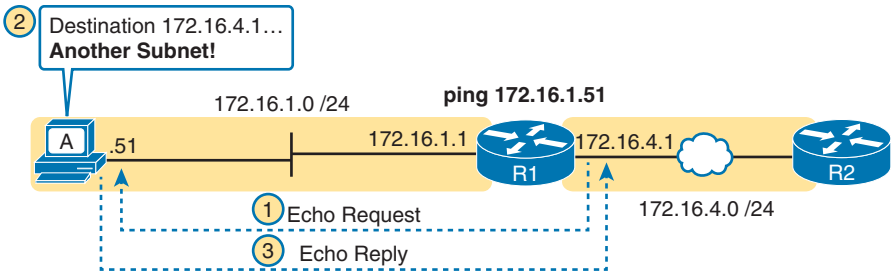

Figure 20-8 *Extended* **ping** *Command Does Test Host A's Default Router Setting*

The comparison between the previous two figures shows one of the most classic mistakes when troubleshooting networks. Sometimes, the temptation is to connect to a router and ping the host on the attached LAN, and it works. So, the engineer moves on, thinking that the network layer issues between the router and host work fine, when the problem still exists with the host's default router setting.

Testing WAN Neighbors with Standard Ping

As with a standard ping test across a LAN, a standard ping test between routers over a serial or Ethernet WAN link tests whether the link can pass IPv4 packets. With a properly designed IPv4 addressing plan, two routers on the same serial or Ethernet WAN link should have IP addresses in the same subnet. A ping from one router to the IP address of the other router confirms that an IP packet can be sent over the link and back, as shown in the **ping 172.16.4.2** command on R1 in Figure 20-9.

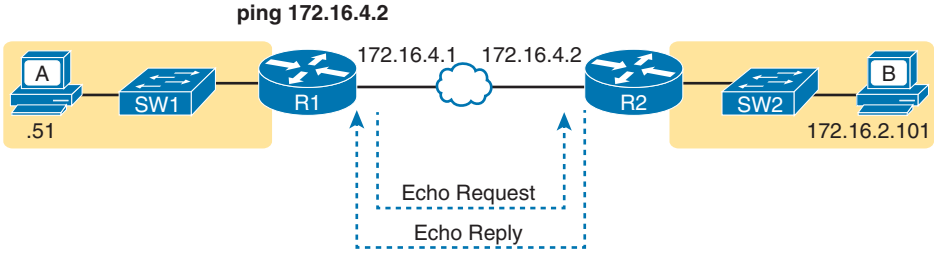

Figure 20-9 *Pinging Across a WAN Link*

A successful ping of the IP address on the other end of an Ethernet WAN link that sits between two routers confirms several specific facts, such as the following:

- Both routers' WAN interfaces are in an up/up state.
- The Layer 1 and 2 features of the link work.
- The routers believe that the neighboring router's IP address is in the same subnet.
- Inbound ACLs on both routers do not filter the incoming packets, respectively.
- The remote router is configured with the expected IP address (172.16.4.2 in this case).

Pinging the other neighboring router does not test many other features. However, although the test is limited in scope, it does let you rule out WAN links as having a Layer 1 or 2 problem, and it rules out some basic Layer 3 addressing problems.

Using Ping with Names and with IP Addresses

All the ping examples so far in this chapter show a ping of an IP address. However, the **ping** command can use **hostnames**, and pinging a hostname allows the network engineer to further test whether the Domain Name System (**DNS**) process works.

First, most every TCP/IP application uses hostnames rather than IP addresses to identify the other device. No one opens a web browser and types in 72.163.4.185. Instead, they type in a web address, like https://www.cisco.com, which includes the hostname www.cisco.com. Then, before a host can send data to a specific IP address, the host must first ask a DNS server to resolve that hostname into the matching IP address.

For example, in the small internetwork used for several examples in this chapter, a **ping B** command on host A tests A's DNS settings, as shown in Figure 20-10. When host A sees the use of a hostname (B), it first looks in its local DNS name cache to find out whether it has already resolved the name B. If not, host A first asks the DNS to supply (resolve) the name into its matching IP address (Step 1 in the figure). Only then does host A send a packet to 172.16.2.101, host B's IP address (Step 2).

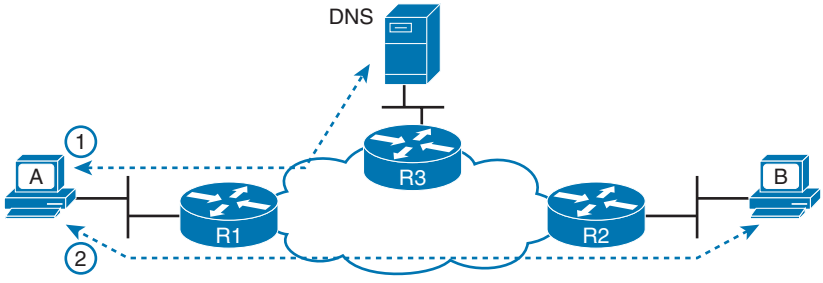

Figure 20-10 *DNS Name Resolution by Host A*

When troubleshooting, testing from the host by pinging using a hostname can be very helpful. The command, of course, tests the host's own DNS client settings. For instance, a classic comparison is to first ping the destination host using the hostname, which requires a DNS request. Then, repeat the same test, but use the destination host's IP address instead of its name, which does not require the DNS request. If the ping of the hostname fails but the ping of the IP address works, the problem usually has something to do with DNS.

Routers and switches can also use name resolution for commands that refer to hosts, such as the **ping** and **traceroute** commands. The networking device can use DNS, locally defined hostnames, or both. Example 20-4 shows an example DNS configuration on Router R1 from the most recent examples. In particular:

Key Topic

- The **ip domain lookup** command tells the router to attempt to use a DNS server.
- The **ip name-server** {*address* [*address address*]} command defines the list of DNS server IP addresses.
- The **ip domain name** *domain-name* command defines the domain used by the device.

In the example, note that once configured to use DNS, the **ping hostB** command works. The command output shows the IP address the DNS resolution process found for name hostB, 172.16.2.101.

Example 20-4 *Configuring to Use DNS (Router R1), DNS Has hostB Name*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip domain lookup
R1(config)# ip domain name example.com
R1(config)# ip name-server 8.8.8.8 8.8.8.4
R1(config)# ^Z
R1#
R1# ping hostB
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

NOTE Older IOS versions used a syntax of **ip domain-name** *domain-name* rather than the newer **ip domain name** *domain-name* (with a space instead of a dash).

NOTE When practicing, you might want to disable DNS resolution, particularly in lab devices, using the **no ip domain lookup** command. Cisco routers and switches enable DNS resolution by default with a setting of **ip domain lookup**, but with no name servers identified with the **ip name-server** command. The result of these two default settings causes the router or switch to perform name resolution on a name by broadcasting for a DNS server on each connected subnet. Additionally, if you mistype the first word of a command, IOS thinks you mean that word to be a hostname, and it attempts to perform name resolution on the mistyped command. The result: for any typo of the first word in a command, the default name resolution settings cause a few minutes wait until you get control of the CLI again.

In a lab environment, when not expecting to use DNS, disable DNS resolution with the **no ip domain lookup** command.

You can also configure the router or switch to use locally configured hostnames (or to use both locally configured names and DNS). Use a configuration like that in Example 20-5, adding the **ip host** *name address* global configuration command for each hostname. The router or switch will look for local hostnames whether you use DNS or have the **ip domain lookup** command configured.

Example 20-5 *Configuring to Use Local Hostnames, R1 Config Has hostB Name*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip host hostB 172.16.2.101
R1(config)# ^Z
R1#
```

```
R1# show hosts
Default domain is example.com
Name servers are 8.8.8.8, 8.8.8.4
NAME   TTL   CLASS    TYPE      DATA/ADDRESS
------------------------------------------
 101.2.16.172.in-addr.arpa        10        IN        PTR      hostB
 hostB                            10        IN        A        172.16.2.101
```

Problem Isolation Using the traceroute Command

Like **ping**, the **traceroute** command helps network engineers isolate problems. Here is a comparison of the two:

Key Topic

- Both send messages in the network to test connectivity.
- Both rely on other devices to send back a reply.
- Both have wide support on many different operating systems.
- Both can use a hostname or an IP address to identify the destination.
- On routers, both have a standard and extended version, allowing better testing of the reverse route.

The biggest differences relate to the more detailed results in the output of the **traceroute** command and the extra time and effort it takes **traceroute** to build that output. This next major section examines how traceroute works; plus it provides some suggestions on how to use this more detailed information to more quickly isolate IP routing problems.

traceroute Basics

Imagine some network engineer or CSR starts to troubleshoot some problem. The engineer pings from the user's host, pings from a nearby router, and after a few commands, convinces herself that the host can indeed send and receive IP packets. The problem might not be solved yet, but the problem does not appear to be a network problem.

Now imagine the next problem comes along, and this time the **ping** command fails. It appears that some problem does exist in the IP network. Where is the problem? Where should the engineer look more closely? Although the **ping** command can prove helpful in isolating the source of the problem, the **traceroute** command may be a better option. The **traceroute** command systematically helps pinpoint routing problems by showing how far a packet goes through an IP network before being discarded.

The **traceroute** command identifies the routers in the forward route from source host to destination host. Specifically, it lists the next-hop IP address of each router that would be in each of the individual routes. For instance, a **traceroute 172.16.2.101** command on host A in Figure 20-11 would identify an IP address on Router R1, another on Router R2, and then host B, as shown in the figure. Example 20-6, which follows, lists the output of the command, taken from host A.

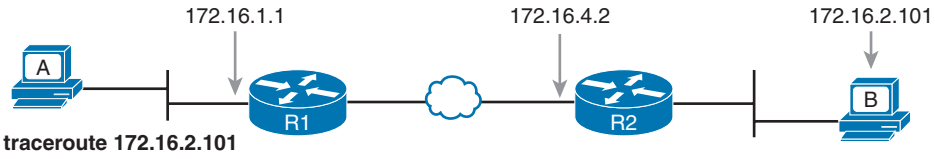

Figure 20-11 *IP Addresses Identified by a Successful* **traceroute 172.16.2.101** *Command*

Example 20-6 *Output from* **traceroute 172.16.2.101** *on Host A*

```
Mac_A$ traceroute 172.16.2.101
traceroute to 172.16.2.101, 64 hops max, 52 byte packets
  1 172.16.1.1 (172.16.1.1) 0.870 ms 0.520 ms 0.496 ms
  2 172.16.4.2 (172.16.4.2) 8.263 ms 7.518 ms 9.319 ms
  3 172.16.2.101 (172.16.2.101) 16.770 ms 9.819 ms 9.830 ms
```

How the traceroute Command Works

The **traceroute** command gathers information by generating packets that trigger error messages from routers; these messages identify the routers, letting the **traceroute** command list the routers' IP addresses in the output of the command. That error message is the ICMP Time-to-Live Exceeded (TTL Exceeded) message, originally meant to notify hosts when a packet had been looping around a network.

Ignoring traceroute for a moment and instead focusing on IP routing, IPv4 routers defeat routing loops in part by discarding looping IP packets. To do so, the IPv4 header holds a field called Time To Live (TTL). The original host that creates the packet sets an initial TTL value. Then each router that forwards the packet decrements the TTL value by 1. When a router decrements the TTL to 0, the router perceives the packet is looping, and the router discards the packet. The router also notifies the host that sent the discarded packet by sending an ICMP TTL Exceeded message.

Now back to traceroute. Traceroute sends messages with low TTL values to make the routers send back a TTL Exceeded message. Specifically, a **traceroute** command begins by sending several packets (usually three), each with the header TTL field equal to 1. When that packet arrives at the next router—host A's default Router R1 in the example of Figure 20-12—the router decrements TTL to 0 and discards the packet. The router then sends host A the TTL Exceeded message, which identifies the router's IP address to the **traceroute** command.

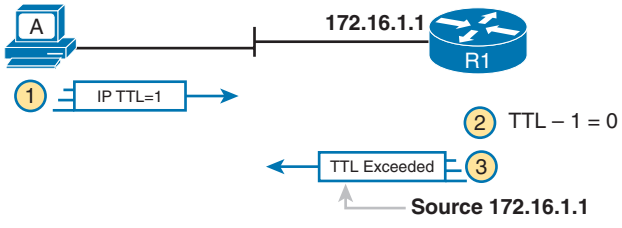

Figure 20-12 *How* **traceroute** *Identifies the First Router in the Route*

The **traceroute** command sends several TTL=1 packets, checking them to see whether the TTL Exceeded messages flow from the same router, based on the source IP address of the TTL Exceeded message. Assuming the messages come from the same router, the **traceroute** command lists that IP address as the next line of output on the command.

To find all the routers in the path, and finally confirm that packets flow all the way to the destination host, the **traceroute** command sends a small set of packets with TTL=1, then a small set with TTL=2, then 3, 4, and so on, until the destination host replies. Figure 20-13 shows the packet from the second set with TTL=2. In this case, one router (R1) actually forwards the packet, while another router (R2) happens to decrement the TTL to 0, causing a TTL Exceeded message to be sent back to host A.

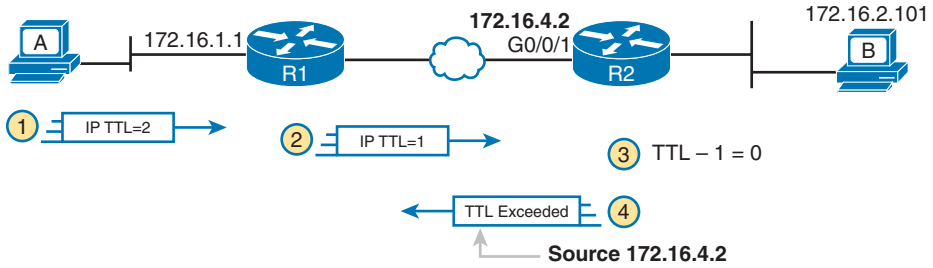

Figure 20-13 *TTL=2 Message Sent by* **traceroute**

The figure shows these four steps:

1. The **traceroute** command sends a packet from the second set with TTL=2.
2. Router R1 processes the packet and decrements TTL to 1. R1 forwards the packet.
3. Router R2 processes the packet and decrements TTL to 0. R2 discards the packet.
4. R2 notifies the sending host of the discarded packet by sending a TTL Exceeded ICMP message. The source IP address of that message is 172.16.4.2.

Finally, the choice of source IP address to use on the time-exceeded message returned by routers has a big impact on the output of the **traceroute** command. Most routers use simpler logic that also makes command output like **traceroute** more consistent and meaningful. That logic: choose the TTL Exceeded message's source IP address based on the interface in which the discarded original message arrived. In the example in Figure 20-13, the original message at Step 2 arrived on R2's G0/0/1 interface, so at Step 3, R2 uses G0/0/1's IP address as the source IP address of the TTL Exceeded message, and as the interface out which to send the message.

Standard and Extended traceroute

The standard and extended options for the **traceroute** command give you many of the same options as the **ping** command. For instance, Example 20-7 lists the output of a standard **traceroute** command on Router R1. Like the standard **ping** command, a standard **traceroute** command chooses an IP address based on the outgoing interface for the packet sent by the command. So, in this example, the packets sent by R1 come from source IP address 172.16.4.1, R1's G0/0/0 IP address.

Example 20-7 *Standard* **traceroute** *Command on R1*

```
R1# traceroute 172.16.2.101
Type escape sequence to abort.
Tracing the route to 172.16.2.101
VRF info: (vrf in name/id, vrf out name/id)
   1 172.16.4.2 0 msec 0 msec 0 msec
   2 172.16.2.101 0 msec 0 msec *
```

The extended **traceroute** command, as shown in Example 20-8, follows the same basic command structure as the extended **ping** command. The user can type all the parameters on one command line, but it is much easier to just type **traceroute**, press Enter, and let IOS prompt for all the parameters, including the source IP address of the packets (172.16.1.1 in this example).

Example 20-8 *Extended* **traceroute** *Command on R1*

```
R1# traceroute
Protocol [ip]:
Target IP address: 172.16.2.101
Source address: 172.16.1.1
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Type escape sequence to abort.
Tracing the route to 172.16.2.101
VRF info: (vrf in name/id, vrf out name/id)
   1 172.16.4.2 0 msec 0 msec 0 msec
   2 172.16.2.101 0 msec 0 msec *
```

Both the **ping** and **traceroute** commands exist on most operating systems, including Cisco IOS. However, some operating systems use a slightly different syntax for **traceroute**. For example, most Windows operating systems support **tracert** and **pathping**, and not **traceroute**. Linux and OS X support the **traceroute** command.

NOTE Host OS **traceroute** commands usually create ICMP echo requests. The Cisco IOS **traceroute** command instead creates IP packets with a UDP header. This bit of information may seem trivial at this point. However, note that an ACL may actually filter the traffic from a host's **traceroute** messages but not the router **traceroute** command, or vice versa.

Telnet and SSH

The **ping** and **traceroute** commands do give networkers two great tools to begin isolating the cause of an IP routing problem. However, these two commands tell us nothing about the

operation state inside the various network devices. Once you begin to get an idea of the kinds of problems and the possible locations of the problems using **ping** and **traceroute**, the next step is to look at the status of various router and switch features. One way to do that is to use Telnet or Secure Shell (SSH) to log in to the devices.

Common Reasons to Use the IOS Telnet and SSH Client

Normally, a network engineer would log in to the remote device using a Telnet or SSH client on a PC, tablet, or any other user device. In fact, often, the same software package does both Telnet and SSH. However, in some cases, you might want to take advantage of the Telnet and SSH client built in to IOS on the routers and switches to Telnet/SSH from one Cisco device to the next.

To understand why, consider the example shown in Figure 20-14. The figure shows arrowed lines to three separate IP addresses on three separate Cisco routers. PC1 has attempted to Telnet to each address from a different tab in PC1's Telnet/SSH client. However, R2 happens to have an error in its routing protocol configuration, so R1, R2, and R3 fail to learn any routes from each other. As a result, PC1's Telnet attempt to both 10.1.2.2 (R2) and 10.1.3.3 (R3) fails.

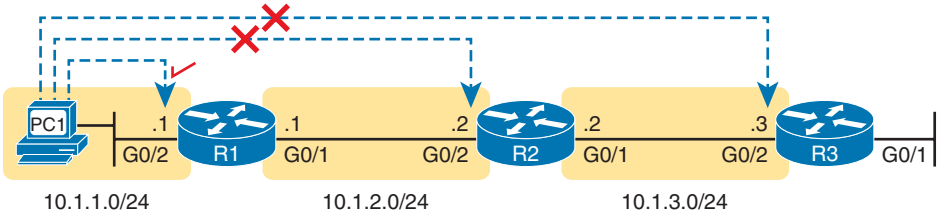

Figure 20-14 *Telnet Works from PC1 to R1 but Not to R2 or R3*

In some cases, like this one, a Telnet or SSH login from the network engineer's device can fail, while you could still find a way to log in using the **telnet** and **ssh** commands to use the Telnet and SSH clients on the routers or switches. With this particular scenario, all the individual data links work; the problem is with the routing protocol exchanging routes. PC1 can ping R1's 10.1.1.1 IP address, R1 can ping R2's 10.1.2.2 address, and R2 can ping R3's 10.1.3.3 address. Because each link works, and each router can send and receive packets with its neighbor on the shared data link, you could Telnet/SSH to each successive device.

Figure 20-15 shows the idea. PC1 begins with a Telnet/SSH connection into Router R1, as shown on the left. Then the user issues the **telnet 10.1.2.2** command from R1 to Telnet to R2. Once logged in to R2, the user can issue commands on R2. Then from R2, the user could issue the **telnet 10.1.3.3** command to Telnet to R3, from which the user could issue commands on R3.

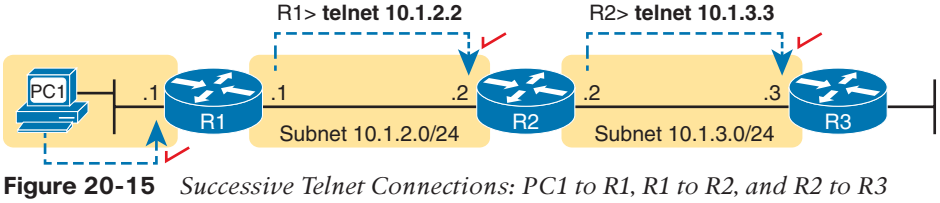

Figure 20-15 *Successive Telnet Connections: PC1 to R1, R1 to R2, and R2 to R3*

The Telnet connections shown in Figure 20-15 work because each Telnet in this case uses source and destination addresses in the same subnet. For example, R1's **telnet 10.1.2.2** command uses 10.1.2.2 as the destination, of course. R1 uses the outgoing interface IP address used to send packets to 10.1.2.2, 10.1.2.1 in this case. Because each of these **telnet** commands connects to an IP address in a connected subnet, the routing protocol could be completely misconfigured, and you could still Telnet/SSH to each successive device to troubleshoot and fix the problem.

Network engineers also use the IOS Telnet and SSH client just for preference. For instance, if you need to log in to several Cisco devices, you could open several windows and tabs on your PC, and log in from your PC (assuming the network was not having problems). Or you could log in from your PC to some nearby Cisco router or switch, and from there Telnet or SSH to other Cisco devices.

IOS Telnet and SSH Examples

Using the IOS Telnet client via the **telnet** *host* command is pretty simple. Just use the IP address or hostname to identify the host to which you want to connect, and press Enter. Example 20-9 shows an example based on Figure 20-15, with R1 using Telnet to connect to 10.1.2.2 (R2).

Example 20-9 *Telnet from R1 to R2 to View Interface Status on R2*

```
R1# telnet 10.1.2.2
Trying 10.1.2.2 ... Open


User Access Verification


Username: wendell
Password:
R2>
R2> show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  administratively down down
GigabitEthernet0/1         10.1.3.2        YES manual up                    up
GigabitEthernet0/2         10.1.2.2        YES manual up                    up
GigabitEthernet0/3         unassigned      YES unset  administratively down down
```

Take the time to pay close attention to the command prompts. The example begins with the user logged in to Router R1, with the R1# command prompt. After the user issues the **telnet 10.1.2.2** command, R2 asks the user for both a username and password because Router R2 uses local username authentication, which requires those credentials. The **show ip interfaces brief** command at the end of the output shows Router R2's interfaces and IP addresses again per Example 20-9 and Figure 20-15.

The **ssh -l** *username host* command in Example 20-10 follows the same basic ideas as the **telnet** *host* command, but with an SSH client. The **-l** flag means that the next parameter is the login username. In this case, the user begins logged in to Router R1 and then uses the **ssh -l wendell 10.1.2.2** command to SSH to Router R2. R2 expects a username/password of wendell/odom, with wendell supplied in the command and odom supplied when R2 prompts the user.

Example 20-10 *SSH Client from R1 to R2 to View Interface Status on R2*

```
R1# ssh -l wendell 10.1.2.2

Password:

R2>
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  administratively down down
GigabitEthernet0/1         10.1.3.2        YES manual up                    up
GigabitEthernet0/2         10.1.2.2        YES manual up                    up
GigabitEthernet0/3         unassigned      YES unset  administratively down down
```

When you have finished using the other router, you can log out from your Telnet or SSH connection using the **exit** or **quit** command.

Finally, note that IOS supports a mechanism to use hotkeys to move between multiple Telnet or SSH sessions from the CLI. Basically, starting at one router, you could telnet or SSH to a router, do some commands, and instead of using the **exit** command to end your connection, you could keep the connection open while still moving back to the command prompt of the original router. For instance, if starting at Router R1, you could telnet to R2, R3, and R4, suspending but not exiting those Telnet connections. Then you could easily move between the sessions to issue new commands with a few keystrokes.

Chapter Review

One key to doing well on the exams is to perform repetitive spaced review sessions. Review this chapter's material using either the tools in the book or interactive tools for the same material found on the book's companion website. Refer to the "Your Study Plan" element for more details. Table 20-1 outlines the key review elements and where you can find them. To better track your study progress, record when you completed these activities in the second column.

Table 20-1 Chapter Review Tracking

Review Element	Review Date(s)	Resource Used
Review key topics		Book, website
Review key terms		Book, website

Review All the Key Topics

Key Topic

Table 20-2 Key Topics for Chapter 20

Key Topic Element	Description	Page Number
Figure 20-5	ARP tables on Layer 3 hosts, with MAC address tables on Layer 2 switch	517
Figure 20-6	How extended ping in IOS performs a better test of the reverse route	518

Key Topic Element	Description	Page Number
Figure 20-7	Why a standard ping over a LAN does not exercise a host's default router logic	519
List	Network layer problems that could cause a ping to fail between a router and host on the same LAN subnet	520
List	Testing a host's default router setting using extended ping	520
List	DNS configuration commands	522
List	Comparisons between the **ping** and **traceroute** commands	524

Key Terms You Should Know

DNS, extended ping, forward route, hostname, ICMP echo reply, ICMP echo request, ping, reverse route, traceroute

Command References

Tables 20-3 and 20-4 list configuration and verification commands used in this chapter. As an easy review exercise, cover the left column in a table, read the right column, and try to recall the command without looking. Then repeat the exercise, covering the right column, and try to recall what the command does.

Table 20-3 Chapter 20 Configuration Command Reference

Command	Description
[no] ip domain lookup	Router/switch global command to enable or disable (**no**) the device from using DNS resolution when commands use a hostname
ip domain name *name*	Router/switch global command to define the DNS domain used by this device
ip name-server *address* [*address*...]	Global command to define one or more DNS server IP addresses
ip host *name address*	Global command that defines a hostname available on the local device

Table 20-4 Chapter 20 EXEC Command Reference

Command	Description
ping {*hostname* | *address*}	EXEC command that sends ICMP Echo Request packets to the address, expecting the distant host to send ICMP Echo Reply messages in return
traceroute {*hostname* | *address*}	EXEC command that messages to a distance host, expecting to force intermediate routes to send ICMP destination unreachable messages, to identify the routers in the path to the distant host
show host	Command to list the device's known hostnames and corresponding IP addresses

Command	Description
show ip interface brief	A router command that lists its interfaces and IP addresses, plus a few more facts, with one line per interface
telnet {*hostname* | *address*}	Command to initiate a Telnet connection to a remote host
ssh -l *username* {*hostname* | *address*}	Command to initiate an SSH connection, for the listed username, to a remote host

This page intentionally left blank

Part V Review

Keep track of your part review progress with the checklist in Table P5-1. Details on each task follow the table.

Table P5-1 Part V Part Review Checklist

Activity	1st Date Completed	2nd Date Completed
Repeat All DIKTA Questions		
Answer Part Review Questions		
Review Key Topics		
Do Labs		
Watch Video		
Use Per-Chapter Interactive Review		

Repeat All DIKTA Questions

For this task, answer the "Do I Know This Already?" questions again for the chapters in this part of the book, using the PTP software.

Answer Part Review Questions

For this task, use PTP to answer the Part Review questions for this part of the book.

Review Key Topics

Review all key topics in all chapters in this part, either by browsing the chapters or by using the Key Topics application on the companion website.

Do Labs

Depending on your chosen lab tool, here are some suggestions for what to do in lab:

Pearson Network Simulator: If you use the full Pearson CCNA simulator, focus more on the configuration scenario and troubleshooting scenario labs associated with the topics in this part of the book. These types of labs include a larger set of topics and work well as Part Review activities. (See the Introduction for some details about how to find which labs are about topics in this part of the book.)

Blog Config Labs: The author's blog includes a series of configuration-focused labs that you can do on paper or with Cisco Packet Tracer in about 15 minutes. To find them, open https://www.certskills.com and look under the Labs menu item.

Other: If using other lab tools, here are a few suggestions: Make sure to experiment heavily with IPv4 addressing, static routing, and Layer 3 switching. In each case, test all your routes using **ping** and **traceroute**.

Watch Video

The companion website includes a variety of common mistake and Q&A videos organized by part and chapter. Use these videos to challenge your thinking, dig deeper, review topics, and better prepare for the exam. Make sure to bookmark a link to the companion website and use the videos for review whenever you have a few extra minutes.