Cisco 200-120

Cisco Certified Network Associate

Study Guide

Exam Code: 200-120

Certifications:

- CCNA Composite Exam

The 200-120 CCNA exam is the composite exam associated with the CCNA Routing and Switching certification. Candidates can prepare for this exam by taking the Interconnecting Cisco Networking Devices Part 1 (ICND1) v2.0 and the Interconnecting Cisco Networking Devices Part 2 (ICND2) v2.0 courses. This exam tests a candidate's knowledge and skills required to install, operate, and troubleshoot a small to medium-size enterprise branch network. The topics include all the areas covered under the 100-101 ICND1 and 200-101 ICND2 exams.

About This Study Guide
This Study Guide provides all the information required to pass the 200-120 CCNA exam. It however, does not represent a complete reference work but is organized around the specific skills that are tested in the exam. Thus, the information contained Study Guide is specific to the 200-120 exam and not to Cisco Certified Network Associate entirely. It includes the information required to answer questions related to CCNA that may be asked during the exam. Topics covered in this Study Guide includes operation of IP data networks, LAN switching technologies, IP addressing, IP routing technologies, IP services, network device security, troubleshooting, LAN switching technologies, IP Routing technologies, IP services, troubleshooting and WAN technologies.

Intended Audience
This Study Guide is targeted specifically at network administrator, network engineer, network manager and systems engineer. This information in this Study Guide is specific to the exam and is not a complete reference work.

Good luck!

TABLE OF CONTENTS

Operation of IP Data Networks

Recognize the purpose and functions of various network devices such as Routers, Switches, Bridges and Hubs

Repeaters

Repeaters are Layer 1 devices utilizing the Physical Layer and are considered as outdated technology today. They have been replaced by Hubs and Switches. But for the purposes of understanding; a repeater consists of a transmitter and a receiver. The function of a repeater is to receive the signal, amplify it and retransmit it enabling the signal to be transmitted over a longer distance.

Repeaters are essential to maintain the quality of signals as they degrade over a distance. Repeaters regenerate and retime the signal, helping it travel a longer distance. Repeaters may be single port or multi port. The figure given below illustrates a repeater.

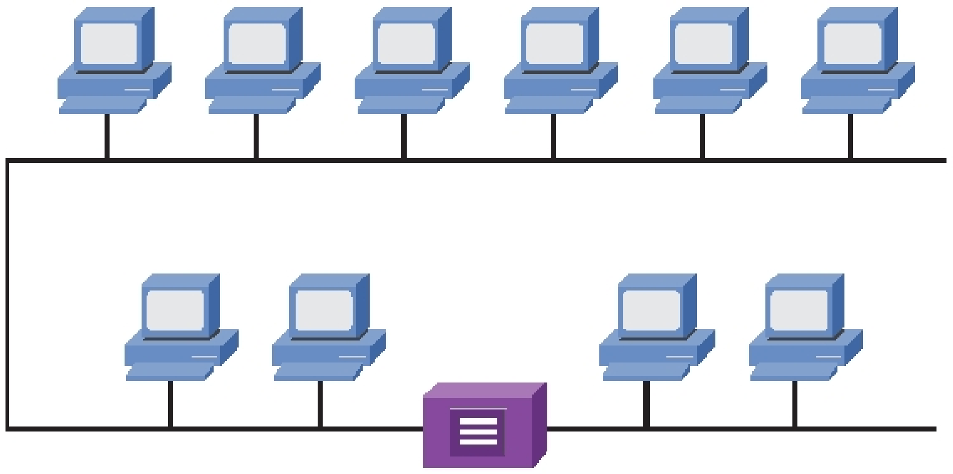


Figure 1: A Repeater

Hubs

A Multiple Port Repeater is termed as a Hub. It is also a Layer 1 device utilizing the Physical Layer. It can comprise of ports varying from 2 to 24 in number and may also be known as a workgroup hub. Its main job is cleaning up signals. By isolating the end points, Hubs increase the network reliability. A hub retransmits data on all the other ports. A twisted pair cable is used for achieving physical connectivity. The figure given below illustrates a HUB

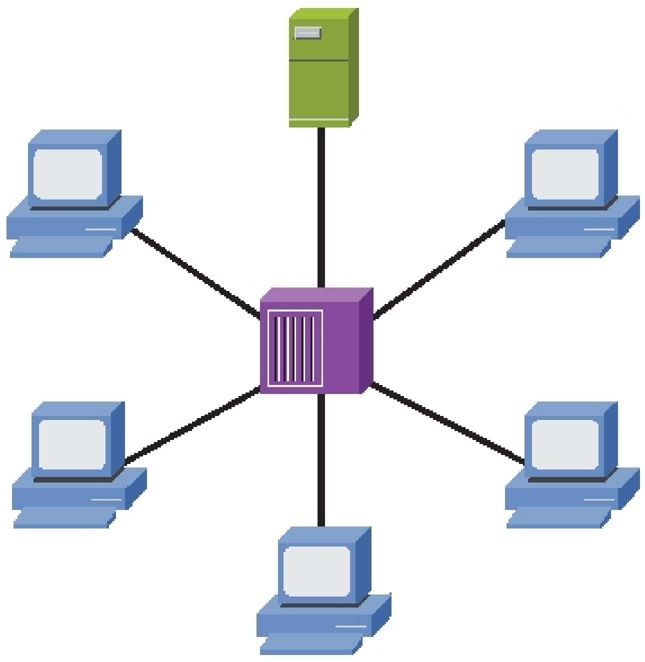


Figure 2: A Hub

Types of Hubs

Hubs can be of two types; Active and Passive hubs. The difference between the two is that Active Hubs regenerate the incoming signal, whereas the Passive Hubs do not do so. Active Hubs need individual power supply to help with the gain of signal before the data is forwarded to all ports. Gain is an electrical term, representing the ratio of signal output to signal input of a system.

The advantage of Hubs is that they are inexpensive. If more efficient use of bandwidth and its distribution among the ports is required, hubs may not be the best option. Traffic congestion because of collisions on the network is indispensable while using hubs. The best solution in this case is to use a switch.

Network Interfaces:

Network interfaces provide connectivity between an end-user computer to the public network. Depending on the interface that is being used, up to three light-emitting diodes (LEDs) may appear. These diodes help to determine the status of the connection.

- The Link Light LED: It lights up when the connectivity is there;
- The Activity Light LED: It flickers if some activity is taking place on the line;
- The Speed Light LED: This light indicates the connection speed. It may be there on the interface, it may not be there.

Blinking lights and colors other than green are indicative of error conditions.

Bridges:

Bridges were used as a solution for issues relating to network congestion. Hubs and Repeaters were longer proving sufficient to meet the challenges provided by systems growing complex. In comparison to Repeaters and Hubs and Bridges used the concept of segmentation. Repeaters and Hubs which do not use segmentation, share the same bandwidth and hence the traffic congestion on a network. When the other device on the network is not aware of the existence of a Bridge, it is called a Transparent Bridge. Figure 3 given below illustrates a Bridge.

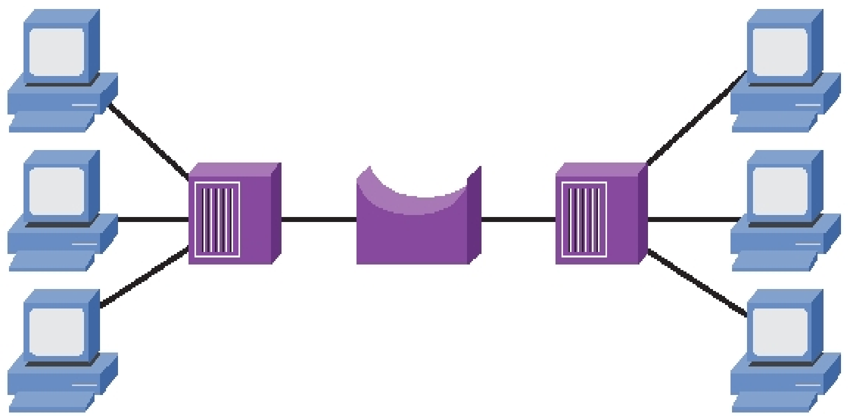


Figure 3: A Bridge

Switches:

Switches are very smart Bridges with the characteristics of being multi port and high speed. The differ bridges from the point that bridges process frames in software whereas switches process frames in hardware. Switches do so by using application integrated circuits (ASIC's). Figure 4 given below illustrates a Switch.

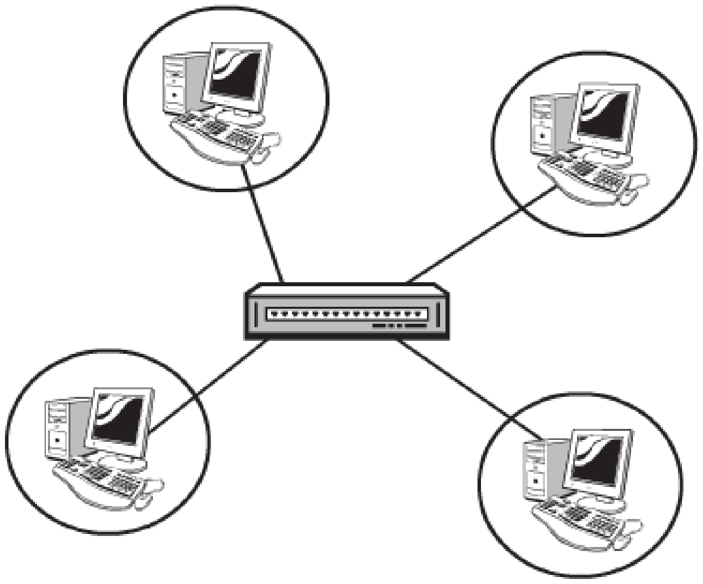


Figure 4: A Switch.

In addition to the above mentioned Switches have the following features:

- Speed Back Plane: this function increases the speed of the network; it allows monitoring of multiple conversations.
- Data Buffering: This function allows storage of frames and later forwarding the frame to the right port.
- High Port Density: Switches can support multiple ports at one time.
- High Port Speed: Switches can support high speeds varying from speeds from 10 Mbps to 10Gbps.
- Lower Latency: Latency is a term that is used to measure the time it takes an incoming frame to come back out of a switch. In the case of switches latency is low.
- VLAN's: This feature allows segmentation of networks into separate broadcast domains.

These features permit micro segmentation.

Micro segmentation:

Micro segmentation means that a dedicated switch ports are created for every end station; meaning that dedicated paths for sending and receiving transmission with each connected hosts are created. These reduce traffic congestion to a great extent for the reason that separate collision domain and individual bandwidth is available for every host. But faster computers, broadcasts and multicasts can still cause congestion.

Bridges and Switches perform the following tasks:

- Ascertainment of MAC Address: Examine the source MAC address of every inbound frame to ascertain its MAC address;
- Filtration/Forwarding: Depending on the destination of the MAC address, filtration or forwarding of frames as the case may demand;
- Elimination: Eliminating loops caused by superfluous connections.

Select the components required to meet a given network specification

Select the components required to meet a network specification
As mentioned in the previous objectives, we use routers, bridges, and switches in an internetwork. Figure 1.5 shows how a network would look with all these internetwork devices in place. Remember that the router will not only break up broadcast domains for every LAN interface, it will break up collision domains as well.

When you looked at Figure 1.5, did you notice that the router is found at center stage and that it connects each physical network together? We have to use this layout because of the older technologies involved—bridges and hubs.

On the top internetwork in Figure 1.5, you’ll notice that a bridge was used to connect the hubs to a router. The bridge breaks up collision domains, but all the hosts connected to both hubs are still crammed into the same broadcast domain. Also, the bridge only created two collision domains, so each device connected to a hub is in the same collision domain as every other device connected to that same hub. This is actually pretty lame, but it’s still better than having one collision domain for all hosts.

Notice something else: The three hubs at the bottom that are connected also connect to the router, creating one collision domain and one broadcast domain. This makes the bridged network look much better indeed!

The best network connected to the router is the LAN switch network on the left. Why? Because each port on that switch breaks up collision domains. But it's not all good—all devices are still in the same broadcast domain. Do you remember why this can be a really bad thing? Because all devices must listen to all broadcasts transmitted, that's why. And if your broadcast domains are too large, the users have less bandwidth and are required to process more broadcasts, and network response time will slow to a level that could cause office riots.

Once we have only switches in our network, things change a lot! Figure 1.6 shows the network that is typically found today.

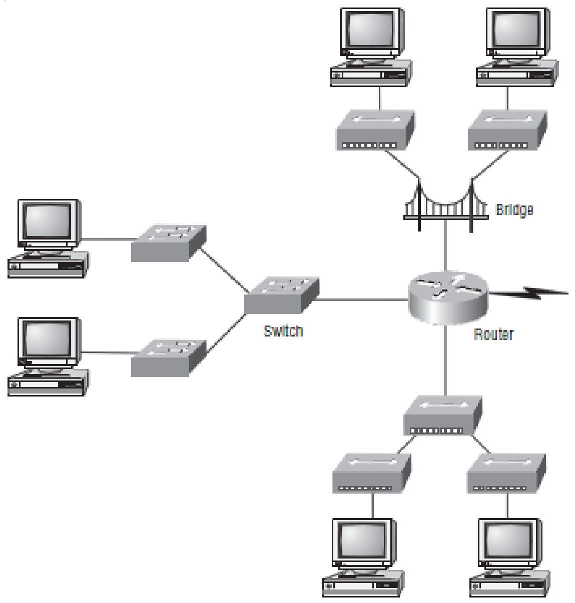


FIGURE 1. 5 Internetworking devices

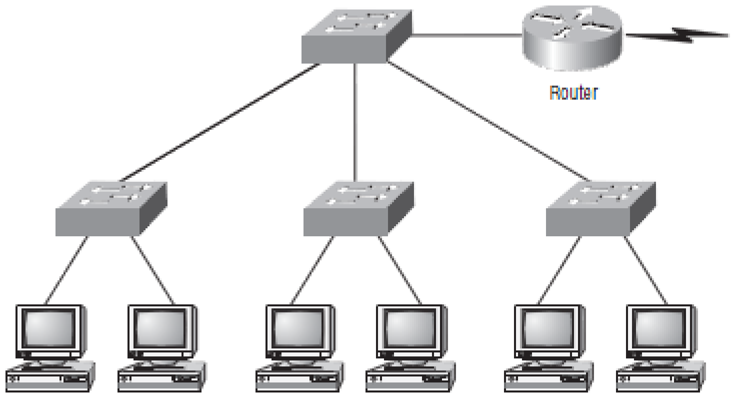


FIGURE 1. 6 Switched networks creating an internetwork

Here, I've placed the LAN switches at the center of the network world so that the routers are connecting only logical networks together. If I implemented this kind of setup, I've created virtual LANs (VLANs). But it is really important to understand that even though you have a switched network, you still need a router to provide your inter-VLAN communication, or internetworking.

Obviously, the best network is one that's correctly configured to meet the business requirements of the company it serves. LAN switches with routers, correctly placed in the network, are the best network design.

Let's go back to Figure 1.6. Looking at the figure, how many collision domains and broadcast domains are in this internetwork? Hopefully, you answered nine collision domains and three broadcast domains! The broadcast domains are definitely the easiest to see because only routers break up broadcast domains by default. And since there are three connections, that gives you three broadcast domains. But do you see the nine collision domains? Just in case that's a no, I'll explain. The all-hub network is one collision domain; the bridge network equals three collision domains. Add in the switch network of five collision domains—one for each switch port—and you've got a total of nine.

So now that you've gotten an introduction to internetworking and the various devices that live in an internetwork, it's time to head into internetworking models.

Identify common applications and their impact on the network

Describe the impact of applications (Voice over IP and Video over IP) on a network
The main purpose of the Host-to-Host layer is to shield the upper-layer applications from the complexities of the network. This layer says to the upper layer, "Just give me your data stream, with any instructions, and I'll begin the process of getting your information ready to send."

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

By understanding how TCP and UDP work, you can interpret the impact of applications on networks when using Voice and Video over IP.

Transmission Control Protocol (TCP)
Transmission Control Protocol (TCP) takes large blocks of information from an application and breaks them into segments. It numbers and sequences each segment so that the destination's TCP stack can put the segments back into the order the application intended. After these segments are sent, TCP (on the transmitting host) waits for an acknowledgment of the receiving end's TCP virtual circuit session, retransmitting those that aren't acknowledged.

Before a transmitting host starts to send segments down the model, the sender's TCP stack contacts the destination's TCP stack to establish a connection. What is created is known as a virtual circuit. This type of communication is called connection-oriented. During this initial handshake, the two TCP layers also agree on the amount of information that's going to be sent before the recipient's TCP sends back an acknowledgment. With everything agreed upon in advance, the path is paved for reliable communication to take place.

TCP is a full-duplex, connection-oriented, reliable, and accurate protocol, but establishing all these terms and conditions, in addition to error checking, is no small task. TCP is very complicated and, not surprisingly, costly in terms of network overhead. And since today's networks are much more reliable than those of yore, this added reliability is often unnecessary.

TCP Segment Format

Since the upper layers just send a data stream to the protocols in the Transport layers, I'll demonstrate how TCP segments a data stream and prepares it for the Internet layer. When the Internet layer receives the data stream, it routes the segments as packets through an internetwork. The segments are handed to the receiving host's Host-to-Host layer protocol, which rebuilds the data stream to hand to the upper-layer applications or protocols.

Figure 1.7 shows the TCP segment format. The figure shows the different fields within the TCP header.

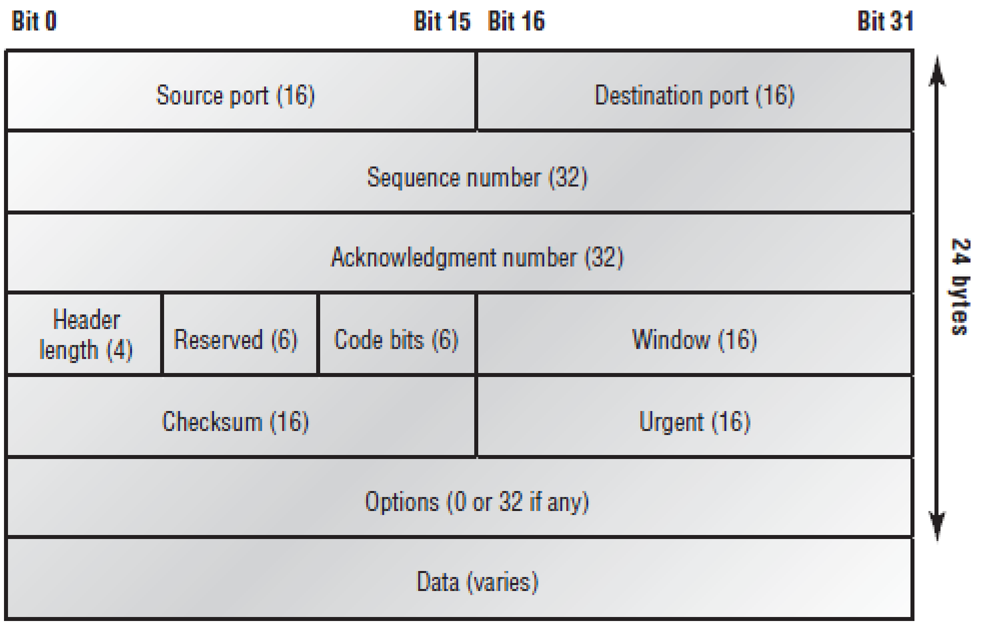


FIGURE 1.7 TCP segment format

The TCP header is 20 bytes long, or up to 24 bytes with options. You need to understand what each field in the TCP segment is:

- **Source port** the port number of the application on the host sending the data.
- **Destination port** The port number of the application requested on the destination host. Sequence number A number used by TCP that puts the data back in the correct order or retransmits missing or damaged data, a process called sequencing.
- **Acknowledgment number** The TCP octet that is expected next.
- **Header length** The number of 32-bit words in the TCP header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits in length. Reserved Always set to zero.
- **Code bits** Control functions used to set up and terminate a session.
- **Window** The window size the sender is willing to accept, in octets.
- **Checksum** The cyclic redundancy check (CRC), because TCP doesn't trust the lower layers and checks everything. The CRC checks the header and data fields.
- **Urgent** A valid field only if the Urgent pointer in the code bits is set. If so, this value indicates the offset from the current sequence number, in octets, where the first segment of non-urgent data begins.
- **Options** May be 0 or a multiple of 32 bits, if any. What this means is that no options have to be present (option size of 0). However, if any options are used that do not cause the option field to total a multiple of 32 bits, padding of 0s must be used to make sure the data begins on a 32-bit boundary.

Data Handed down to the TCP protocol at the Transport layer, which includes the upperlayer headers.

Let's take a look at a TCP segment copied from a network analyzer:

TCP - Transport Control Protocol

```
Source Port: 5973
Destination Port: 23
Sequence Number: 1456389907
Ack Number: 1242056456
Offset: 5
Reserved: %000000
Code: %011000
Ack is valid
Push Request
Window: 61320
Checksum: 0x61a6
Urgent Pointer: 0
No TCP Options
```

```
TCP Data Area:
vL.5.+.5.+.5.+.5 76 4c 19 35 11 2b 19 35 11 2b 19 35 11
2b 19 35 +. 11 2b 19
Frame Check Sequence: 0x0d00000f
```

Did you notice that everything I talked about earlier is in the segment? As you can see from the number of fields in the header, TCP creates a lot of overhead. Application developers may opt for efficiency over reliability to save overhead, so the User Datagram Protocol was also defined at the Transport layer as an alternative.

User Datagram Protocol (UDP)

If you were to compare the User Datagram Protocol (UDP) with TCP, the former is basically the scaled-down economy model that’s sometimes referred to as a thin protocol. Like a thin person on a park bench, a thin protocol doesn’t take up a lot of room—or in this case, much bandwidth on a network.

UDP doesn’t offer all the bells and whistles of TCP either, but it does do a fabulous job of transporting information that doesn’t require reliable delivery—and it does so using far fewer network resources. (UDP is covered thoroughly in Request for Comments 768.)

There are some situations in which it would definitely be wise for developers to opt for UDP rather than TCP. Remember the watchdog SNMP up there at the Process/Application layer? SNMP monitors the network, sending intermittent messages and a fairly steady flow of status updates and alerts, especially when running on a large network. The cost in overhead to establish, maintain, and close a TCP connection for each one of those little messages would reduce what would be an otherwise healthy, efficient network to a dammed-up bog in no time!

Another circumstance calling for UDP over TCP is when reliability is already handled at the Process/Application layer. Network File System (NFS) handles its own reliability issues, making the use of TCP both impractical and redundant. But ultimately, it’s up to the application developer to decide whether to use UDP or TCP, not the user who wants to transfer data faster.

UDP does not sequence the segments and does not care in which order the segments arrive at the destination. But after that, UDP sends the segments off and forgets about them. It doesn’t follow through, check up on them, or even allow for an acknowledgment of safe arrival—complete abandonment. Because of this, it’s referred to as an unreliable protocol.

This does not mean that UDP is ineffective, only that it doesn’t handle issues of reliability. Further, UDP doesn’t create a virtual circuit, nor does it contact the destination before delivering information to it. Because of this, it’s also considered a connectionless protocol.

Since UDP assumes that the application will use its own reliability method, it doesn't use any. This gives an application developer a choice when running the Internet Protocol stack: TCP for reliability or UDP for faster transfers.

So if you're using Voice over IP (VoIP), for example, you really don't want to use UDP, because if the segments arrive out of order (very common in IP networks), they'll just be passed up to the next OSI (DoD) layer in whatever order they're received, resulting in some seriously garbled data. On the other hand, TCP sequences the segments so they get put back together in exactly the right order—something that UDP just can't do.

UDP Segment Format
Figure 1.8 clearly illustrates UDP's markedly low overhead as compared to TCP's hungry usage. Look at the figure carefully—can you see that UDP doesn't use windowing or provide for acknowledgments in the UDP header?

It's important for you to understand what each field in the UDP segment is: Source port Port number of the application on the host sending the data Destination port Port number of the application requested on the destination host Length Length of UDP header and UDP data Checksum Checksum of both the UDP header and UDP data fields Data Upper-layer data

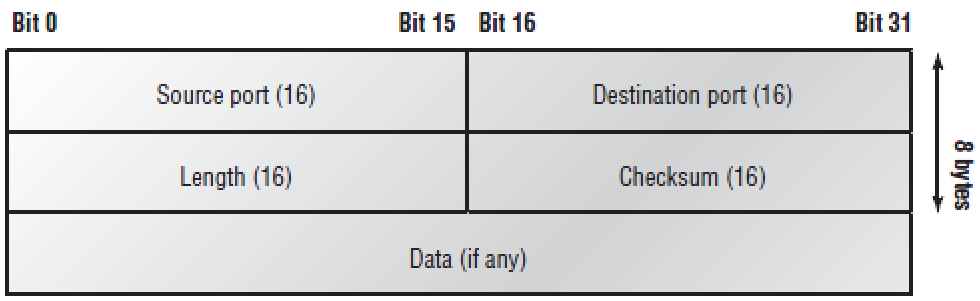


FIGURE 1.8 UDP segment

UDP, like TCP, doesn't trust the lower layers and runs its own CRC. Remember that the Frame Check Sequence (FCS) is the field that houses the CRC, which is why you can see the FCS information. The following shows a UDP segment caught on a network analyzer:

UDP - User Datagram Protocol

```
Source Port: 1085
Destination Port: 5136
Length: 41
Checksum: 0x7a3c
UDP Data Area:
..Z......00 01 5a 96 00 01 00 00 00 00 00 11 0000 00
...C..2._C._C 2e 03 00 43 02 1e 32 0a 00 0a 00 80 43 00 80
```

```
Frame Check Sequence: 0x00000000
```

Notice that low overhead! Try to find the sequence number, ack number, and window size in the UDP segment. You can't because they just aren't there!

Key Concepts of Host-to-Host Protocols

Since you've seen both a connection-oriented (TCP) and connectionless (UDP) protocol in action, it would be good to summarize the two here. Table 1. 2 highlight some of the key concepts that you should keep in mind regarding these two protocols. You should memorize this table.

TABLE 1.2 Key Features of TCP and UDP

TCP	UDP
Sequenced	Unsequenced
Reliable	Unreliable
Connection-oriented	Connectionless
Virtual circuit	Low overhead
Acknowledgments	No acknowledgment
Windowing flow control	No windowing or flow control

A telephone analogy could really help you understand how TCP works. Most of us know that before you speak to someone on a phone, you must first establish a connection with that other person—wherever they are. This is like a virtual circuit with the TCP protocol. If you were giving someone important information during your conversation, you might say, "You know?" or ask, "Did you get that?" Saying something like this is a lot like a TCP acknowledgment— it's designed to get you verification. From time to time (especially on cell phones), people also ask, "Are you still there?" They end their conversations with a "Goodbye" of some kind, putting closure on the phone call. TCP also performs these types of functions.

Alternately, using UDP is like sending a postcard. To do that, you don't need to contact the other party first. You simply write your message, address the postcard, and mail it. This is analogous to UDP's connectionless orientation. Since the message on the postcard is probably not a matter of life or death, you don't need an acknowledgment of its receipt. Similarly, UDP does not involve acknowledgments.

Describe the purpose and basic operation of the protocols in the OSI and TCP/IP models

Overview of the TCP/IP Networking Model
The TCP/IP model both defines and references a large collection of protocols that allow computers to communicate. To define a protocol, TCP/IP uses documents called Requests for Comments (RFC). (You can find these RFCs using any online search engine.) The TCP/IP model also avoids repeating work already done by some other standards body or vendor consortium by simply referring to standards or protocols created by those groups.

For example, the Institute of Electrical and Electronic Engineers (IEEE) defines Ethernet LANs; the TCP/IP model does not define Ethernet in RFCs, but refers to IEEE Ethernet as an option.
An easy comparison can be made between telephones and computers that use TCP/IP. You go to the store and buy a phone from one of a dozen different vendors. When you get home and plug in the phone to the same cable in which your old phone was connected, the new phone works. The phone vendors know the standards for phones in their country and build their phones to match those standards.

Similarly, when you buy a new computer today, it implements the TCP/IP model to the point that you can usually take the computer out of the box, plug in all the right cables, turn it on, and it connects to the network. You can use a web browser to connect to your favorite website. How? Well, the OS on the computer implements parts of the TCP/IP model. The Ethernet card, or wireless LAN card, built into the computer implements some LAN standards referenced by the TCP/IP model. In short, the vendors that created the hardware and software implemented TCP/IP.

To help people understand a networking model, each model breaks the functions into a small number of categories called layers. Each layer includes protocols and standards that relate to that category of functions. TCP/IP actually has two alternative models, as shown in Figure 1.9.

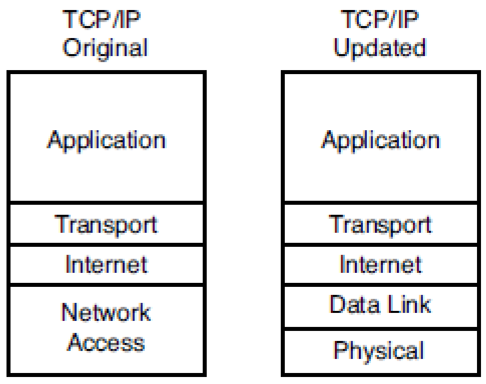
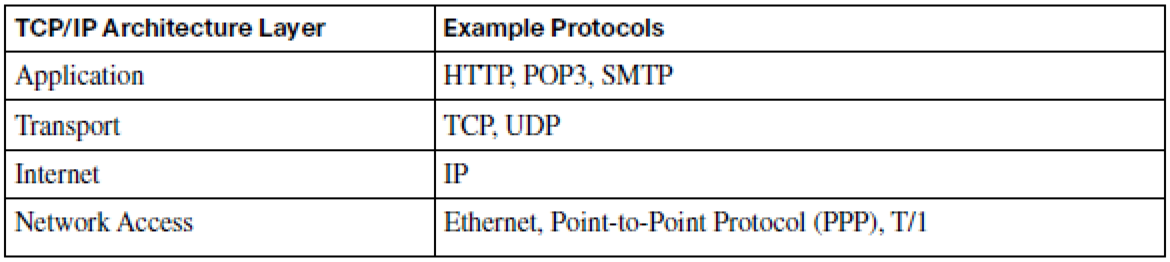


Figure 1.9 The Two TCP/IP Networking Models

The model on the left, the original TCP/IP model, breaks TCP/IP into four layers. The top layers focus more on the applications that need to send and receive data, whereas the lower layers focus more on the need to somehow transmit the bits from one device to another. The model on the right is a newer version of the model, formed by expanding the network access layer on the left into two separate layers: data link and physical. Note that the model on the right is used more often today.

Many of you will have already heard of several TCP/IP protocols, like the examples listed in

Table 1.3 TCP/IP Architectural Model and Example Protocols



TCP/IP Architecture Layer	Example Protocols
Application	HTTP, POP3, SMTP
Transport	TCP, UDP
Internet	IP
Network Access	Ethernet, Point-to-Point Protocol (PPP), T/1

TCP/IP Application Layer

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

Table 1.3 TCP/IP Architectural Model and Example Protocols Arguably, the most popular TCP/IP application today is the web browser. Many major software vendors either have already changed or are changing their application software to support access from a web browser.

OSI Layers and Their Functions

Cisco requires that CCNAs demonstrate a basic understanding of the functions defined by each OSI layer, as well as remembering the names of the layers. You understand which layers of the OSI model most closely match the functions defined by that device or protocol.

Today, because most people happen to be much more familiar with TCP/IP functions than with OSI functions, one of the best ways to learn about the function of different OSI layers is to think about the functions in the TCP/IP model, and correlate those with the OSI model.

If you use the five-layer TCP/IP model, the bottom four layers of OSI and TCP/IP map closely together. The only difference in these bottom four layers is the name of OSI Layer 3 (network) compared to TCP/IP (Internet). The upper three layers of the OSI reference model (application, presentation, and session—Layers 7, 6, and 5) define functions that all map to the TCP/IP application layer. Table 1.4 defines the functions of the seven layers.

Table 1.4 OSI Reference Model Layer Definitions

Layer	Functional Description
7	Layer 7 provides an interface between the communications software and any applications that need to communicate outside the computer on which the application resides. It also defines processes for user authentication.
6	This layer's main purpose is to define and negotiates data formats, such as ASCII text, EBCDIC text, binary, BCD, and JPEG. Encryption is also defined by OSI as a presentation layer service.
5	The session layer defines how to start, control, and end conversations (called sessions). This includes the control and management of multiple bidirectional messages so that the
4	Layer 4 protocols provide a large number of services, "Fundamentals of TCP/IP Transport, Applications, and Security." Although OSI Layers 5 through 7 focus on issues related to the
3	The network layer defines three main features: logical addressing, routing (forwarding), and path determination. Routing defines how devices (typically routers) forward packets to their final destination.
2	The data link layer defines the rules that determine when a device can send data over a particular medium. Data link protocols also define the format of a header and trailer that allows devices attached
1	This layer typically refers to standards from other organizations. These standards deal with the physical characteristics of the transmission medium, including connectors, pins, use of pins,

Table 1.5 lists most of the devices and protocols covered in the CCNA exams and their comparable OSI layers. Note that many network devices must actually understand the protocols at multiple OSI layers, so the layer listed in Table 1.5 actually refers to the highest layer that the device normally thinks about when performing its core work. For example, routers need to think about Layer 3 concepts, but they must also support features at both Layers 1 and 2.

Besides remembering the basics of the features of each OSI layer (as in Table 1.4), and some example protocols and devices at each layer (as in Table 1.5), you should also Layer Functional Description 4 Layer 4 protocols provide a large number of services, "Fundamentals of TCP/IP Transport, Applications, and Security." Although OSI Layers 5 through 7 focus on issues related to the application, Layer 4 focuses on issues related to data delivery to another computer (for instance, error recovery and flow control).

3 The network layer defines three main features: logical addressing, routing (forwarding), and path determination. Routing defines how devices (typically routers) forward packets to their final destination. Logical addressing defines how each device can have an address that can be used by the routing process. Path determination refers to the work done by routing protocols to learn all possible routes, and choose the best route.

2 The data link layer defines the rules that determine when a device can send data over a particular medium. Data link protocols also define the format of a header and trailer that allows devices attached to the medium to successfully send and receive data.

1 This layer typically refers to standards from other organizations. These standards deal with the physical characteristics of the transmission medium, including connectors, pins, use of pins, electrical currents, encoding, light modulation, and the rules for how to activate and deactivate the use of the physical medium.

Table 1.5 OSI Reference Model—Example Devices and Protocols

Layer Name	Protocols and	Devices
Application, presentation, session (Layers 5–7)	Telnet, HTTP, FTP, SMTP, POP3, VoIP, SNMP	Firewall, intrusion detection systems, hosts
Transport (Layer 4)	TCP, UDP	Hosts, firewalls
Network (Layer 3)	IP	Router
Data link (Layer 2)	Ethernet (IEEE 802.3), HDLC, Frame Relay, PPP	LAN switch, wireless access point, cable modem,
Physical (Layer 1)	RJ-45, EIA/TIA-232, V.35, Ethernet (IEEE 802.3)	LAN hub, LAN repeater, cables

Memorize the names of the layers. You can simply memorize them, but some people like to use a mnemonic phrase to make memorization easier. In the following three phrases, the first letter of each word is the same as the first letter of an OSI layer name, in the order specified in parentheses:

- All People Seem To Need Data Processing (Layers 7 to 1)
- Please Do Not Take Sausage Pizzas Away (Layers 1 to 7)

- Pew! Dead Ninja Turtles Smell Particularly Awful (Layers 1 to 7)

Predict the data flow between two hosts across a network

Determine the path between two hosts across a network
Once you create an internetwork by connecting your WANs and LANs to a router, you'll need to configure logical network addresses, such as IP addresses, to all hosts on the internetwork so that they can communicate across that internetwork.

The term routing is used for taking a packet from one device and sending it through the network to another device on a different network. Routers don't really care about hosts—they only care about networks and the best path to each network. The logical network address of the destination host is used to get packets to a network through a routed network, and then the hardware address of the host is used to deliver the packet from a router to the correct destination host.

If your network has no routers, then it should be apparent that you are not routing. Routers route traffic to all the networks in your internetwork. To be able to route packets, a router must know, at a minimum, the following:

- Destination address
- Neighbor routers from which it can learn about remote networks
- Possible routes to all remote networks
- The best route to each remote network
- How to maintain and verify routing information

The router learns about remote networks from neighbor routers or from an administrator. The router then builds a routing table (a map of the internetwork) that describes how to find the remote networks. If a network is directly connected, then the router already knows how to get to it.

If a network isn't directly connected to the router, the router must use one of two ways to learn how to get to the remote network: static routing, meaning that someone must hand-type all network locations into the routing table, or something called dynamic routing. In dynamic routing, a protocol on one router communicates with the same protocol running on neighbor routers. The routers then update each other about all the networks they know about and place this information into the routing table. If a change occurs in the network, the dynamic routing protocols automatically inform all routers about the event. If static routing is used, the administrator is responsible for updating all changes by hand into all routers. Typically, in a large network, a combination of both dynamic and static routing is used.

Figure 1.10 shows a simple two-router network. Lab_A has one serial interface and three LAN interfaces.
Looking at Figure 1.10, can you see which interface Lab_A will use to forward an IP datagram to a host with an IP address of 10.10.10.10?

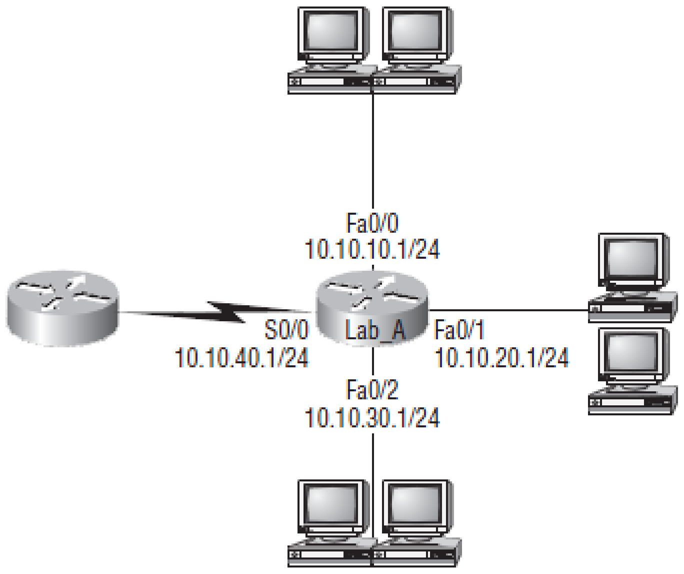


FIGURE 1.10 A simple routing example

By using the command show ip route, we can see the routing table (map of the internetwork) that Lab_A uses to make forwarding decisions:

```
Lab_A#sh ip route
[output cut]
Gateway of last resort is not set
C 10.10.10.0/24 is directly connected, FastEthernet0/0
C 10.10.20.0/24 is directly connected, FastEthernet0/1
C 10.10.30.0/24 is directly connected, FastEthernet0/2
C 10.10.40.0/24 is directly connected, Serial 0/0
```

The C in the routing table output means that the networks listed are “directly connected,” and until we add a routing protocol—something like RIP, EIGRP, or the like—to the routers in our internetwork (or use static routes), we’ll have only directly connected networks in our routing table.

So let’s get back to the original question: By looking at the figure and the output of the routing table, can you tell what IP will do with a received packet that has a destination IP address of 10.10.10.10? The router will packet-switch the packet to interface FastEthernet 0/0, and this interface will frame the packet and then send it out on the network segment.

Because we can, let's do another example: Based on the output of the next routing table, which interface will a packet with a destination address of 10.10.10.14 be forwarded from?

```
Lab_A#sh ip route
[output cut]
Gateway of last resort is not set
C 10.10.10.16/28 is directly connected, FastEthernet0/0
C 10.10.10.8/29 is directly connected, FastEthernet0/1
C 10.10.10.4/30 is directly connected, FastEthernet0/2
C 10.10.10.0/30 is directly connected, Serial 0/0
```

First, you can see that the network is sub-netted and each interface has a different mask. And I have to tell you—you just can't answer this question if you can't subnet! 10.10.10.14 would be a host in the 10.10.10.8/29 subnet connected to the FastEthernet0/1 interface.

Figure 1.11 shows a LAN connected to Router A, which is, in turn, connected via a WAN link to RouterB. RouterB has a LAN connected with an HTTP server attached.

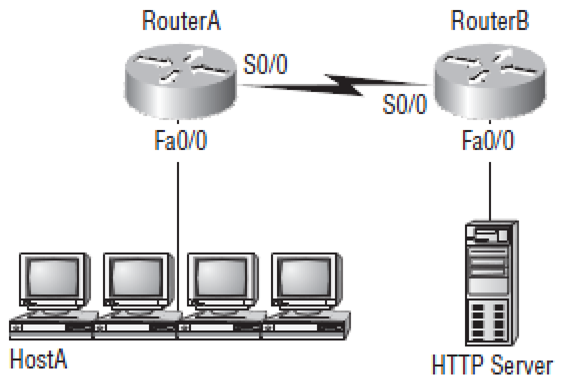


FIGURE 1.11 IP routing example 1

The critical information you need to glean from this figure is exactly how IP routing will occur in this example. Okay—we'll cheat a bit. I'll give you the answer, but then you should go back over the figure and see if you can answer example 2 without looking at my answers.

1. The destination address of a frame, from HostA, will be the MAC address of the F0/0 interface of the RouterA router.
2. The destination address of a packet will be the IP address of the network interface card (NIC) of the HTTP server.
3. The destination port number in the segment header will have a value of 80.

That example was a pretty simple one, and it was also very to the point. One thing to remember is that if multiple hosts are communicating to the server using HTTP, they must all use a different source port number. That is how the server keeps the data separated at the Transport layer.

Let's mix it up a little and add another internetworking device into the network and then see if you can find the answers. Figure 1.12 shows a network with only one router but two switches.

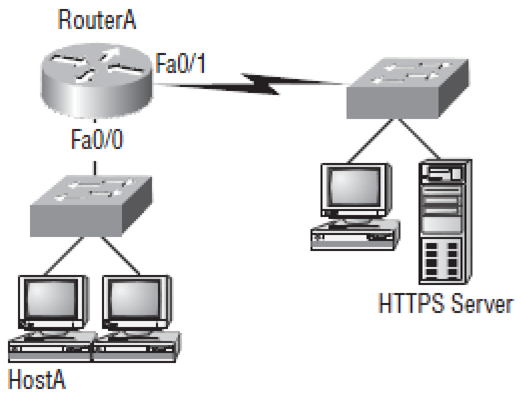


FIGURE 1.12 IP routing example 2.

What you want to understand about the IP routing process here is what happens when HostA sends data to the HTTPS server:

1. The destination address of a frame, from HostA, will be the MAC address of the F0/0 interface of the RouterA router.
2. The destination address of a packet will be the IP address of the network interface card (NIC) of the HTTPS server.
3. The destination port number in the segment header will have a value of 443.

Notice that the switches weren't used as either a default gateway or another destination.
That's because switches have nothing to do with routing. I wonder how many of you chose the switch as the default gateway (destination) MAC address for HostA? If you did, don't feel bad—just take another look with that fact in mind. It's very important to remember that the destination MAC address will always be the router's interface—if your packets are destined for outside the LAN, as they were in these last two examples.

Before we move into some of the more advanced aspects of IP routing, let's discuss ICMP in more detail, as well as how ICMP is used in an internetwork. Take a look at the network shown in Figure 1.13. Ask yourself what will happen if the LAN interface of Lab_C goes down.

Lab_C will use ICMP to inform Host A that Host B can’t be reached, and it will do this by sending an ICMP destination unreachable message. Lots of people think that the Lab_A router would be sending this message, but they would be wrong because the router that sends the message is the one with that interface that’s down is located.

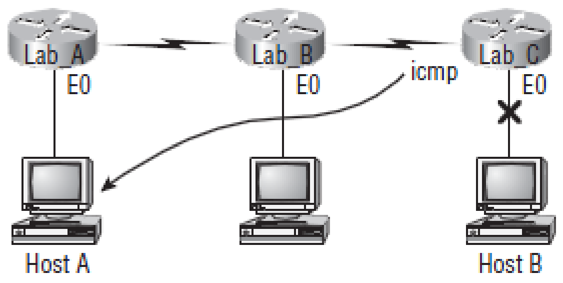


FIGURE 1.13 ICMP error example

Let’s look at another problem: Look at the output of a corporate router’s routing table:

```
Corp#sh ip route
[output cut]
R 192.168.215.0 [120/2] via 192.168.20.2, 00:00:23, Serial0/0
R 192.168.115.0 [120/1] via 192.168.20.2, 00:00:23, Serial0/0
R 192.168.30.0 [120/1] via 192.168.20.2, 00:00:23, Serial0/0
C 192.168.20.0 is directly connected, Serial0/0
C 192.168.214.0 is directly connected, FastEthernet0/0
```

What do we see here? If I were to tell you that the corporate router received an IP packet with a source IP address of 192.168.214.20 and a destination address of 192.168.22.3, what do you think the Corp router will do with this packet?

If you said, “The packet came in on the FastEthernet 0/0 interface, but since the routing table doesn’t show a route to network 192.168.22.0 (or a default route), the router will discard the packet and send an ICMP destination unreachable message back out interface FastEthernet 0/0,” you’re a genius! The reason it does this is because that’s the source LAN where the packet originated from.

Identify the appropriate media, cables, ports, and connectors to connect Cisco network devices to other network devices and hosts in a LAN

Select the appropriate media, cables, ports, and connectors to connect switches to other network devices and hosts

Ethernet cabling is an important discussion, especially if you are planning on taking the Cisco exams. Three types of Ethernet cables are available:

- Straight-through cable
- Crossover cable
- Rolled cable

Straight-Through Cable
The straight-through cable is used to connect

- Host to switch or hub
- Router to switch or hub

Four wires are used in straight-through cable to connect Ethernet devices. It is relatively simple to create this type; Figure 1.12 shows the four wires used in a straight-through Ethernet cable.

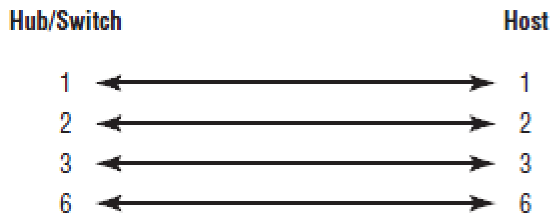


FIGURE 1.12 Straight-through Ethernet cable

Notice that only pins 1, 2, 3, and 6 are used. Just connect 1 to 1, 2 to 2, 3 to 3, and 6 to 6, and you'll be up and networking in no time. However, remember that this would be an Ethernet-only cable and wouldn't work with voice, Token Ring, ISDN, and so on.

Crossover Cable
The crossover cable can be used to connect

- Switch to switch
- Hub to hub
- Host to host
- Hub to switch
- Router direct to host

The same four wires are used in this cable as in the straight-through cable; we just connect different pins together. Figure 1.13 shows how the four wires are used in a crossover Ethernet cable.

Notice that instead of connecting 1 to 1, 2 to 2, and so on, here we connect pins 1 to 3 and 2 to 6 on each side of the cable.

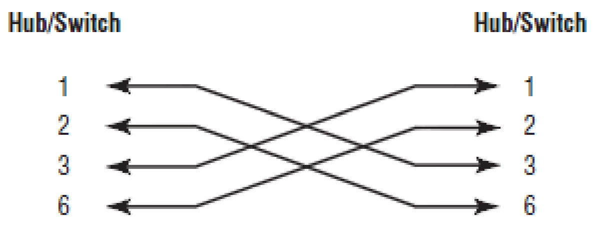


FIGURE 1.13 Crossover Ethernet cable

Rolled Cable
Although rolled cable isn't used to connect any Ethernet connections, you can use a rolled Ethernet cable to connect a host to a router console serial communication (com) port.

If you have a Cisco router or switch, you would use this cable to connect your PC running HyperTerminal to the Cisco hardware. Eight wires are used in this cable to connect serial devices, although not all eight are used to send information, just as in Ethernet networking. Figure 1.14 shows the eight wires used in a rolled cable.

These are probably the easiest cables to make because you just cut the end off on one side of a straight-through cable, turn it over, and put it back on (with a new connector, of course).

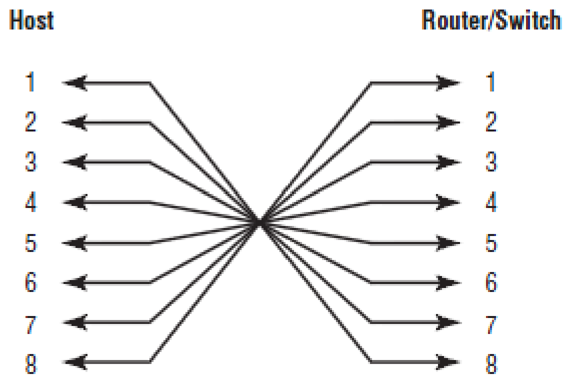
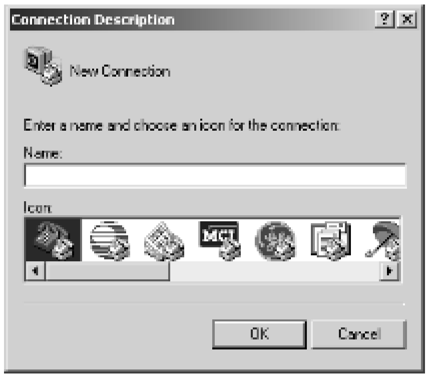


FIGURE 1.14 Rolled Ethernet cable

Once you have the correct cable connected from your PC to the Cisco router or switch, you can start HyperTerminal to create a console connection and configure the device. Set the configuration as follows:

1. Open HyperTerminal and enter a name for the connection. It is irrelevant what you name it, but I always just use Cisco. Then click OK.



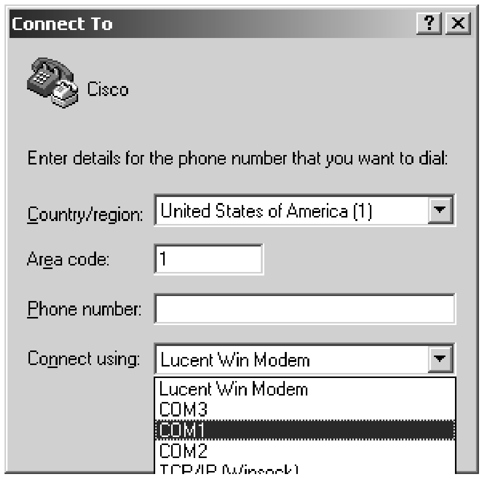
New Connection

Enter a name and choose an icon for the connection:

Name:

Icon:

2. Choose the communications port—either COM1 or COM2, whichever is open on your PC.



Cisco

Enter details for the phone number that you want to dial:

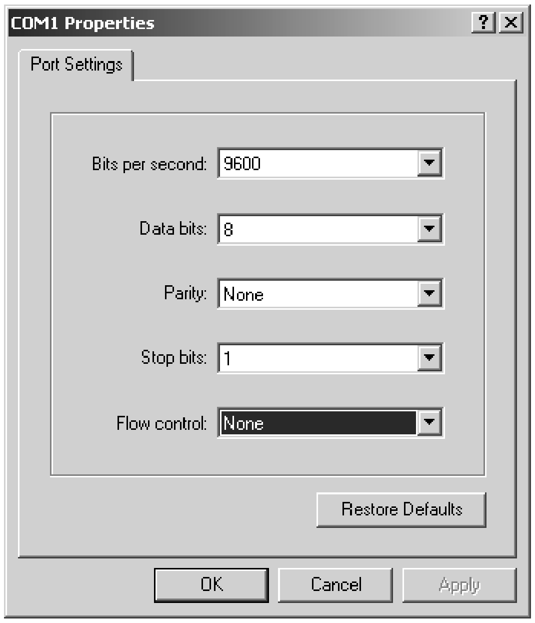
Country/region: United States of America (1)

Area code: 1

Phone number:

Connect using: Lucent Win Modem

3. Now set the port settings. The default values (2400bps and no flow control hardware) will not work; you must set the port settings as shown in Figure 1.14.



Bits per second: 9600

Data bits: 8

Parity: None

Stop bits: 1

Flow control: None

FIGURE 1.14 Port settings for a rolled cable connection

Notice that the bit rate is now set to 9600 and the flow control is set to None. At this point, you can click OK and press the Enter key and you should be connected to your Cisco device console port.

We've taken a look at the various RJ45 unshielded twisted pair (UTP) cables. Keeping this in mind, what cable is used between the switches in Figure 1.15?

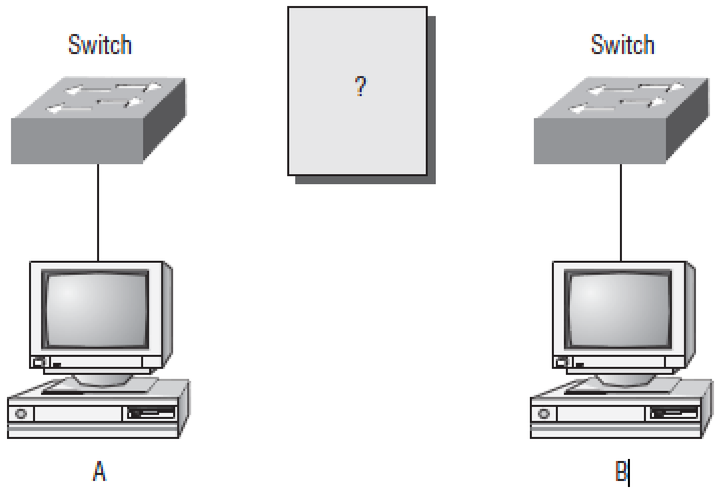


FIGURE 1.15 RJ45 cable questions #1

In order for host A to ping host B, you need a crossover cable to connect the two switches. But what types of cables are used in the network shown in Figure 1.16?

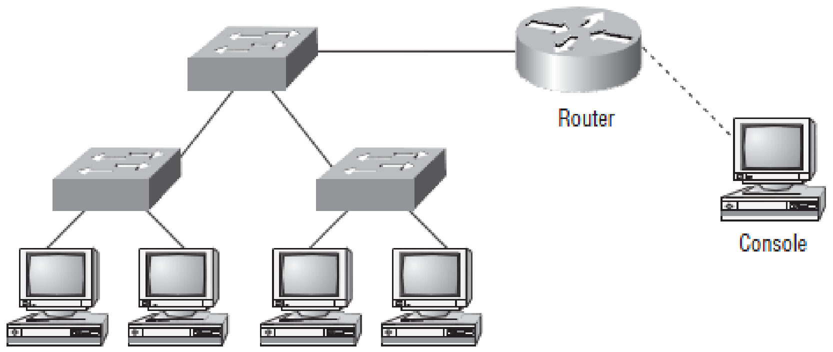


FIGURE 1.16 RJ45 cable questions #2

The trouble is, we have a console connection that uses a rolled cable. Plus, the connection from the router to the switch is a straight-through cable, as is true for the hosts to the switches. Keep in mind that if we had a serial connection (which we don't); it would be a V.35 that we'd use to connect us to a WAN.

LAN Switching Technologies

Determine the technology and media access control method for Ethernet networks

Explain the technology and media access control method for Ethernet networks

Ethernet is a contention media access method that allows all hosts on a network to share the same bandwidth of a link. Ethernet is popular because it’s readily scalable, meaning that it’s comparatively easy to integrate new technologies, such as Fast Ethernet and Gigabit Ethernet, into an existing network infrastructure. It’s also relatively simple to implement in the first place, and with it, troubleshooting is reasonably straightforward.

Ethernet networking uses Carrier Sense Multiple Access with Collision Detection (CSMA/CD), a protocol that helps devices share the bandwidth evenly without having two devices transmit at the same time on the network medium. CSMA/CD was created to overcome the problem of those collisions that occur when packets are transmitted simultaneously from different nodes. And trust me—good collision management is crucial, because when a node transmits in a CSMA/CD network, all the other nodes on the network receive and examine that transmission. Only bridges and routers can effectively prevent a transmission from propagating throughout the entire network!

So, how does the CSMA/CD protocol work? Let’s start by taking a look at Figure 2.1. When a host wants to transmit over the network, it first checks for the presence of a digital signal on the wire. If all is clear (no other host is transmitting), the host will then proceed with its transmission. But it doesn’t stop there. The transmitting host constantly monitors the wire to make sure no other hosts begin transmitting. If the host detects another signal on the wire, it sends out an extended jam signal that causes all nodes on the segment to stop sending data (think busy signal). The nodes respond to that jam signal by waiting a while before attempting to transmit again. Back off algorithms determine when the colliding stations can retransmit. If collisions keep occurring after 15 tries, the nodes attempting to transmit will then timeout. Pretty clean!

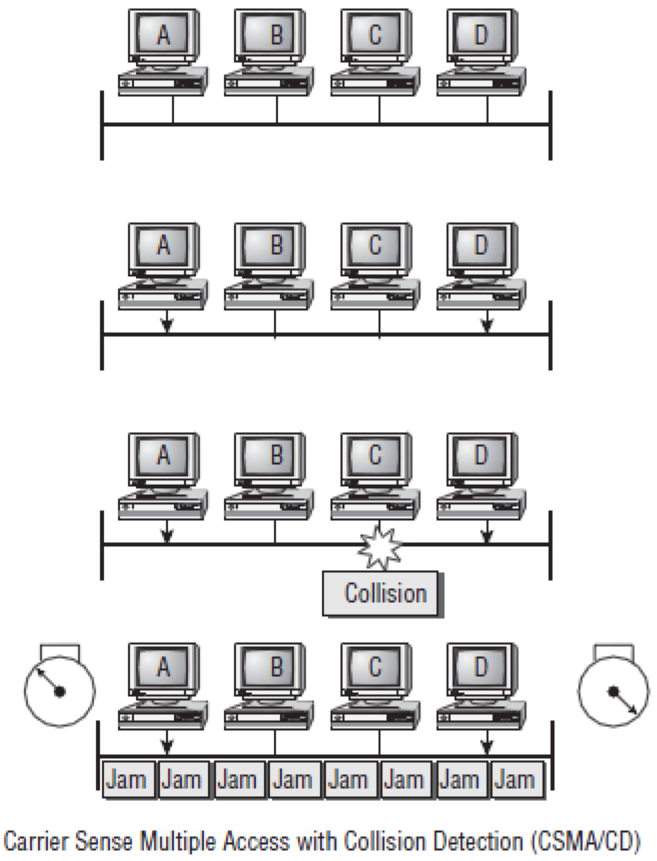

A
B
C
D
Collision
Jam
Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

FIGURE 2.1 CSMA/CD

When a collision occurs on an Ethernet LAN, the following happens:

- A jam signal informs all devices that a collision occurred.
- The collision invokes a random backoff algorithm.
- Each device on the Ethernet segment stops transmitting for a short time until the timers expire.
- All hosts have equal priority to transmit after the timers have expired.

The following are the effects of having a CSMA/CD network sustaining heavy collisions:

- Delay
- Low throughput
- Congestion

I am going to cover Ethernet in detail at both the Data Link layer (layer 2) and the Physical layer (layer 1).

Half- and Full-Duplex Ethernet

Half-duplex Ethernet is defined in the original 802.3 Ethernet; Cisco says it uses only one wire pair with a digital signal running in both directions on the wire. Certainly, the IEEE specifications discuss the process of half-duplex somewhat differently, but what Cisco is talking about is a general sense of what is happening here with Ethernet.

It also uses the CSMA/CD protocol to help prevent collisions and to permit retransmitting if a collision does occur. If a hub is attached to a switch, it must operate in half-duplex mode because the end stations must be able to detect collisions. Half-duplex Ethernet—typically 10BaseT—is only about 30 to 40 percent efficient as Cisco sees it because a large 10BaseT network will usually only give you 3 to 4Mbps, at most.

But full-duplex Ethernet uses two pairs of wires instead of one wire pair like half-duplex. And full-duplex uses a point-to-point connection between the transmitter of the transmitting device and the receiver of the receiving device. This means that with full-duplex data transfer, you get a faster data transfer compared to half-duplex. And because the transmitted data is sent on a different set of wires than the received data, no collisions will occur.

The reason you don't need to worry about collisions is that now it's like there is a freeway with multiple lanes instead of the single-lane road provided by half-duplex. Full-duplex Ethernet is supposed to offer 100 percent efficiency in both directions—for example, you can get 20Mbps with a 10Mbps Ethernet running full-duplex or 200Mbps for Fast Ethernet. But this rate is something known as an aggregate rate , which translates as "you're supposed to get" 100 percent efficiency. No guarantees, in networking as in life.

Full-duplex Ethernet can be used in three situations:

- With a connection from a switch to a host
- With a connection from a switch to a switch
- With a connection from a host to a host using a crossover cable

Last, remember these important points:

- There are no collisions in full-duplex mode.
- A dedicated switch port is required for each full-duplex node.
- The host network card and the switch port must be capable of operating in full-duplex mode.

Now let's take a look at how Ethernet works at the Data Link layer.

Ethernet at the Data Link Layer

Ethernet at the Data Link layer is responsible for Ethernet addressing, commonly referred to as hardware addressing or MAC addressing. Ethernet is also responsible for framing packets received from the Network layer and preparing them for transmission on the local network through the Ethernet contention media access method.

Ethernet Addressing

Here's where we get into how Ethernet addressing works. It uses the Media Access Control (MAC) address burned into each and every Ethernet network interface card (NIC). The MAC, or hardware, address is a 48-bit (6-byte) address written in a hexadecimal format. Figure 2.2 shows the 48-bit MAC addresses and how the bits are divided.

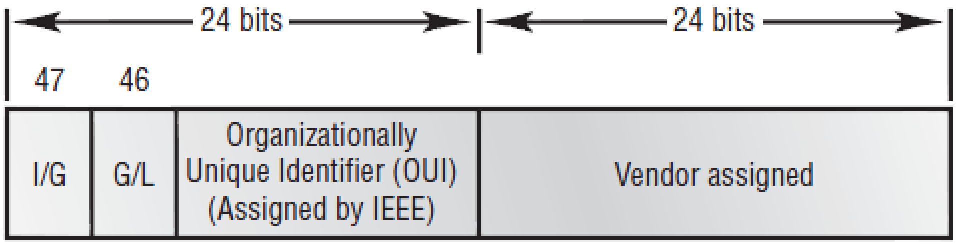


FIGURE 2 .2 Ethernet addressing using MAC addresses

The organizationally unique identifier (OUI) is assigned by the IEEE to an organization. It's composed of 24 bits, or 3 bytes. The organization, in turn, assigns a globally administered address (24 bits, or 3 bytes) that is unique (supposedly, again—no guarantees) to each and every adapter it manufactures. Look closely at the figure. The high-order bit is the Individual/ Group (I/G) bit. When it has a value of 0, we can assume that the address is the MAC address of a device and may well appear in the source portion of the MAC header. When it is a 1, we can assume that the address represents either a broadcast or multicast address in Ethernet or a broadcast or functional address in TR and FDDI (who really knows about FDDI?).

The next bit is the global/local bit, or just G/L bit (also known as U/L, where U means universal). When set to 0, this bit represents a globally administered address (as by the IEEE). When the bit is a 1, it represents a locally governed and administered address (as in what DECnet used to do). The low-order 24 bits of an Ethernet address represent a locally administered or manufacturer- assigned code. This portion commonly starts with 24 0s for the first card made and continues in order until there are twenty-four 1s for the last (16,777,216th) card made. You'll find that many manufacturers use these same six hex digits as the last six characters of their serial number on the same card.

Ethernet Frames

The Data Link layer is responsible for combining bits into bytes and bytes into frames. Frames are used at the Data Link layer to encapsulate packets handed down from the Network layer for transmission on a type of media access.

The function of Ethernet stations is to pass data frames between each other using a group of bits known as a MAC frame format. This provides error detection from a cyclic redundancy check (CRC). But remember—this is error detection, not error correction. The 802.3 frames and Ethernet frame are shown in Figure 2.3.

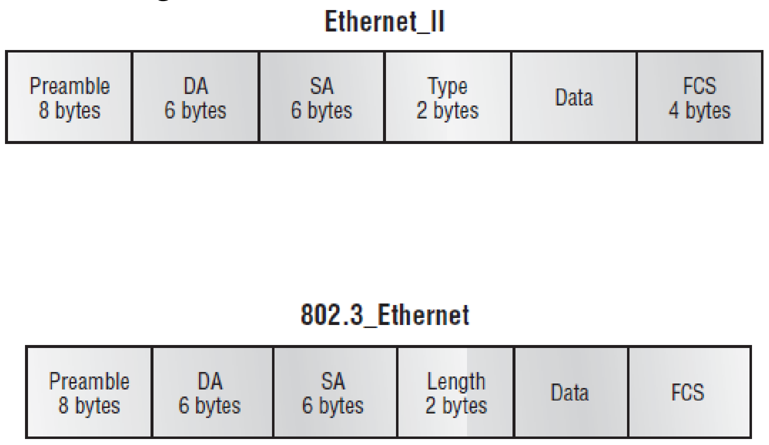


FIGURE 2.3 802.3 and Ethernet frame formats

Following are the details of the different fields in the 802.3 and Ethernet frame types: Preamble an alternating 1, 0 pattern provides a 5MHz clock at the start of each packet, which allows the receiving devices to lock the incoming bit stream.

Start Frame Delimiter (SFD)/Synch The preamble is seven octets, and the SFD is one octet (synch). The SFD is 10101011, where the last pair of 1s allows the receiver to come into the alternating 1, 0 pattern somewhere in the middle and still sync up and detect the beginning of the data.

- **Destination Address** (DA) this transmits a 48-bit value using the least significant bit (LSB) first. The DA is used by receiving stations to determine whether an incoming packet is addressed to a particular node. The destination address can be an individual address or a broadcast or multicast MAC address. Remember that a broadcast is all 1s (or Fs in hex) and is sent to all devices but a multicast is sent only to a similar subset of nodes on a network.

- **Source Address** (SA) The SA is a 48-bit MAC address used to identify the transmitting device, and it uses the LSB first. Broadcast and multicast address formats are illegal within the SA field.
- Length or Type 802.3 uses a Length field, but the Ethernet frame uses a Type field to identify the Network layer protocol. 802.3 cannot identify the upper-layer protocol and must be used with a proprietary LAN—IPX, for example.

- **Data** This is a packet sent down to the Data Link layer from the Network layer. The size can vary from 64 to 1500 bytes.

- **Frame Check Sequence (FCS)** FCS is a field at the end of the frame that's used to store the CRC.

Let's pause here for a minute and take a look at some frames caught on our trusty OmniPeek network analyzer. You can see that the frame below has only three fields: Destination, Source, and Type (shown as Protocol Type on this analyzer):

```
Destination: 00:60:f5:00:1f:27
Source: 00:60:f5:00:1f:2c
Protocol Type: 08-00 IP
```

This is an Ethernet_II frame. Notice that the type field is IP, or 08-00 (mostly just referred to as 0x800) in hexadecimal.

The next frame has the same fields, so it must be an Ethernet_II frame too:

```
Destination: ff:ff:ff:ff:ff:ff Ethernet Broadcast
Source: 02:07:01:22:de:a4
Protocol Type: 08-00 IP
```

Did you notice that this frame was a broadcast? You can tell because the destination hardware address is all 1s in binary, or all Fs in hexadecimal.

Let's take a look at one more Ethernet_II frame. You can see that the Ethernet frame is the same Ethernet_II frame we use with the IPv4 routed protocol, but the type field has 0x86dd when we are carrying IPv6 data, and when we have IPv4 data, we use 0x0800 in the protocol field:

```
Destination: IPv6-Neighbor-Discovery_00:01:00:03 (33:33:00:01:00:03)
Source: Aopen_3e:7f:dd (00:01:80:3e:7f:dd)
Type: IPv6 (0x86dd)
```

This is the beauty of the Ethernet_II frame. Because of the protocol field, we can run any Network layer routed protocol, and it will carry the data because it can identify the Network layer protocol.

Ethernet at the Physical Layer

Ethernet was first implemented by a group called DIX (Digital, Intel, and Xerox). They created and implemented the first Ethernet LAN specification, which the IEEE used to create the IEEE

802.3 Committee. This was a 10Mbps network that ran on coax and then eventually twisted-pair and fiber physical media.

The IEEE extended the 802.3 Committee to two new committees known as 802.3u (Fast Ethernet) and 802.3ab (Gigabit Ethernet on category 5) and then finally 802.3ae (10Gbps over fiber and coax).Figure 2.4 shows the IEEE 802.3 and original Ethernet Physical layer specifications.

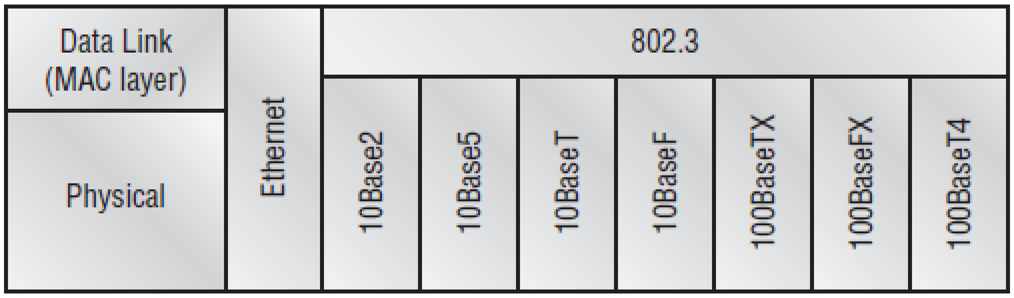


FIGURE 2.4 Ethernet Physical layer specifications

When designing your LAN, it’s really important to understand the different types of Ethernet media available to you. Sure, it would be great to run Gigabit Ethernet to each desktop and 10Gbps between switches, and although this might happen one day, justifying the cost of that network today would be pretty difficult. But if you mix and match the different types of Ethernet media methods currently available, you can come up with a cost-effective network solution that works great.

The EIA/TIA (Electronic Industries Association and the newer Telecommunications Industry Alliance) is the standards body that creates the Physical layer specifications for Ethernet. The EIA/TIA specifies that Ethernet use a registered jack (RJ) connector with a 4 5 wiring sequence on unshielded twisted-pair (UTP) cabling (RJ45). However, the industry is moving toward calling this just an 8-pin modular connector.

Each Ethernet cable type that is specified by the EIA/TIA has inherent attenuation, which is defined as the loss of signal strength as it travels the length of a cable and is measured in decibels (dB). The cabling used in corporate and home markets is measured in categories. A higher quality cable will have a higher-rated category and lower attenuation. For example, category 5 is better than category 3 because category 5 cables have more wire twists per foot and therefore less crosstalk. Crosstalk is the unwanted signal interference from adjacent pairs in the cable. Here are the original IEEE 802.3 standards:

10Base2 10Mbps, baseband technology, up to 185 meters in length. Known as thinnet and can support up to 30 workstations on a single segment. Uses a physical and logical bus with AUI connectors. The 10 means 10Mbps, Base means baseband technology (which is a signaling method for communication on the network), and the 2 means almost 200 meters. 10Base2 Ethernet cards use BNC (British Naval Connector, Bayonet Neill Concelman, or Bayonet Nut Connector) and T-connectors to connect to a network. 10Base5 10Mbps, baseband technology, up to 500 meters in length. Known as thicknet. Uses a physical and logical bus with AUI connectors. Up to 2,500 meters with repeaters and 1,024 users for all segments. 10BaseT 10Mbps using category 3 UTP wiring. Unlike with the 10Base2 and 10Base5 networks, each device must connect into a hub or switch, and you can have only one host per segment or wire. Uses an RJ45 connector (8-pin modular connector) with a physical star topology and a logical bus.

Each of the 802.3 standards defines an Attachment Unit Interface (AUI), which allows a one-bit-at-a-time transfer to the Physical layer from the Data Link media access method. This allows the MAC to remain constant but means that the Physical layer can support any existing and new technologies. The original AUI interface was a 15-pin connector, which allowed a transceiver (transmitter/receiver) that provided a 15-pin-to-twisted-pair conversion.

The thing is, the AUI interface cannot support 100Mbps Ethernet because of the high frequencies involved. So, 100BaseT needed a new interface, and the 802.3u specifications created one called the Media Independent Interface (MII), which provides 100Mbps throughput. The MII uses a nibble, defined as 4 bits. Gigabit Ethernet uses a Gigabit Media Independent Interface (GMII) and transmits 8 bits at a time.

802.3u (Fast Ethernet) is compatible with 802.3 Ethernet because they share the same physical characteristics. Fast Ethernet and Ethernet use the same maximum transmission unit (MTU), use the same MAC mechanisms, and preserve the frame format that is used by 10BaseT Ethernet.

Basically, Fast Ethernet is just based on an extension to the IEEE 802.3 specification, except that it offers a speed increase of 10 times that of 10BaseT.

Here are the expanded IEEE Ethernet 802.3 standards:

100BaseTX (IEEE 802.3u) EIA/TIA category 5, 6, or 7 UTP two-pair wiring. One user per segment; up to 100 meters long. It uses an RJ45 connector with a physical star topology and a logical bus.

100BaseFX (IEEE 802.3u) Uses fiber cabling 62.5/125-micron multimode fiber. Point-topoint topology; up to 412 meters long. It uses an ST or SC connector, which are media-interface connectors.

1000BaseCX (IEEE 802.3z) Copper twisted-pair called twinax (a balanced coaxial pair) that can only run up to 25 meters.

1000BaseT (IEEE 802.3ab) Category 5, four-pair UTP wiring up to 100 meters long. 1000BaseSX (IEEE 802.3z) MMF using 62.5- and 50-micron core; uses an 850 nanometer laser and can go up to 220 meters with 62.5 micron, 550 meters with 50 micron. 1000BaseLX (IEEE 802.3z) Single-mode fiber that uses a 9-micron core and 1300 nanometer laser and can go from 3 kilometers up to 10 kilometers.

Identify basic switching concepts and the operation of Cisco switches

Explain basic switching concepts and the operation of Cisco switches

Unlike bridges, which use software to create and manage a filter table, switches use application-specific integrated circuits (ASICs) to build and maintain their filter tables. But it’s still okay to think of a layer 2 switch as a multiport bridge because their basic reason for being is the same: to break up collision domains.

Layer 2 switches and bridges are faster than routers because they don’t take up time looking at the Network layer header information. Instead, they look at the frame’s hardware addresses before deciding to either forward, flood, or drop the frame.

Switches create private, dedicated collision domains and provide independent bandwidth on each port, unlike hubs. Figure 2.5 shows five hosts connected to a switch—all running 10Mbps half-duplex to the server. Unlike with a hub, each host has 10Mbps dedicated communication to the server.

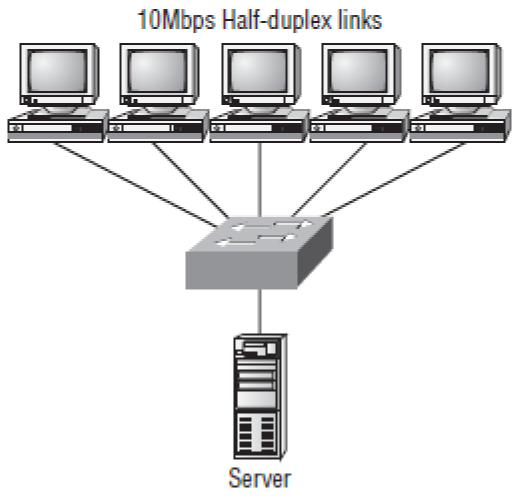


FIGURE 2.5 Switches create private domains.

Layer 2 switching provides the following:

Hardware-based bridging (ASIC)

- Wire speed
- Low latency
- Low cost

What makes layer 2 switching so efficient is that no modification to the data packet takes place. The device only reads the frame encapsulating the packet, which makes the switching process considerably faster and less error-prone than routing processes are.

And if you use layer 2 switching for both workgroup connectivity and network segmentation (breaking up collision domains), you can create a flatter network design with more network segments than you can with traditional routed networks.

Plus, layer 2 switching increases bandwidth for each user because, again, each connection (interface) into the switch is its own collision domain. This feature makes it possible for you to connect multiple devices to each interface.

I will dive deeper into the layer 2 switching technology.

Limitations of Layer 2 Switching
Since we commonly stick layer 2 switching into the same category as bridged networks, we also tend to think it has the same hang-ups and issues that bridged networks do. Keep in mind that bridges are good and helpful things if we design the network correctly, keeping their features as well as their limitations in mind. And to design well with bridges, these are the two most important considerations:

- We absolutely must break up the collision domains correctly.
- The right way to create a functional bridged network is to make sure that its users spend 80 percent of their time on the local segment.

Bridged networks break up collision domains, but remember, that network is still one large broadcast domain. Neither layer 2 switches nor bridges break up broadcast domains by default—something that not only limits your network's size and growth potential but also can reduce its overall performance.

Broadcasts and multicasts, along with the slow convergence time of spanning trees, can give you some major grief as your network grows. These are the big reasons that layer 2 switches and bridges cannot completely replace routers (layer 3 devices) in the internetwork.

Bridging vs. LAN Switching
It’s true—layer 2 switches really are pretty much just bridges that give us a lot more ports, but there are some important differences you should always keep in mind:

- Bridges are software based, while switches are hardware based because they use ASIC chips to help make filtering decisions.
- A switch can be viewed as a multiport bridge.
- There can be only one spanning-tree instance per bridge, while switches can have many. (I’m going to tell you all about spanning trees in a bit.)
- Switches have a higher number of ports than most bridges.
- Both bridges and switches forward layer 2 broadcasts.
- Bridges and switches learn MAC addresses by examining the source address of each frame received.
- Both bridges and switches make forwarding decisions based on layer 2 addresses.

Three Switch Functions at Layer 2
There are three distinct functions of layer 2 switching (you need to remember these!): address learning, forward/filter decisions, and loop avoidance.

Address learning Layer 2 switches and bridges remember the source hardware address of each frame received on an interface, and they enter this information into a MAC database called a forward/filter table.

Forward/filter decisions When a frame is received on an interface, the switch looks at the destination hardware address and finds the exit interface in the MAC database. The frame is only forwarded out the specified destination port.

Loop avoidance If multiple connections between switches are created for redundancy purposes, network loops can occur. Spanning Tree Protocol (STP) is used to stop network loops while still permitting redundancy.

Address Learning
When a switch is first powered on, the MAC forward/filter table is empty, as shown in Figure 2.6.

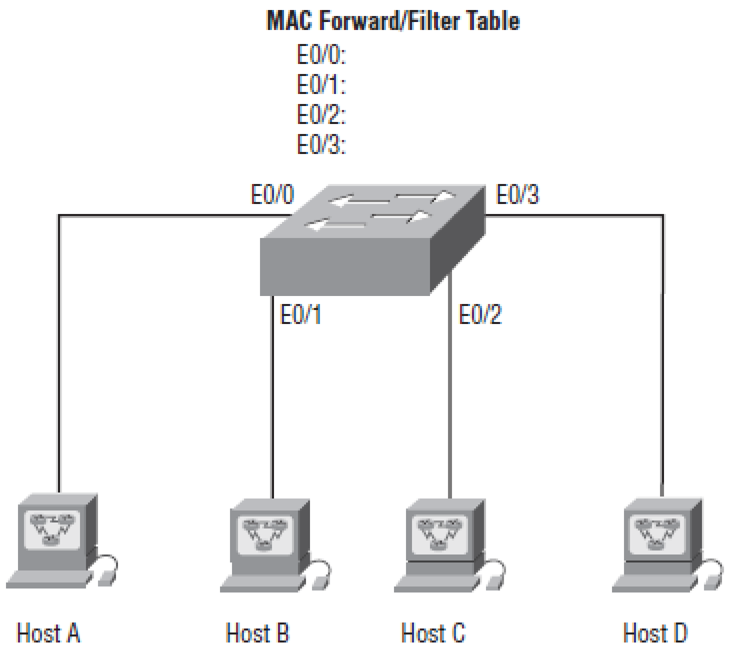


FIGURE 2.6 Empty forward/filter tables on a switch

When a device transmits and an interface receives a frame, the switch places the frame's source address in the MAC forward/filter table, allowing it to remember which interface the sending device is located on. The switch then has no choice but to flood the network with this frame out of every port except the source port because it has no idea where the destination device is actually located.

If a device answers this flooded frame and sends a frame back, then the switch will take the source address from that frame and place that MAC address in its database as well, associating this address with the interface that received the frame. Since the switch now has both of the relevant MAC addresses in its filtering table, the two devices can now make a point-to-point connection. The switch doesn't need to flood the frame as it did the first time because now the frames can and will be forwarded only between the two devices. This is exactly the thing that makes layer 2 switches better than hubs. In a hub network, all frames are forwarded out all ports every time—no matter what. Figure 2.7 shows the processes involved in building a MAC database.

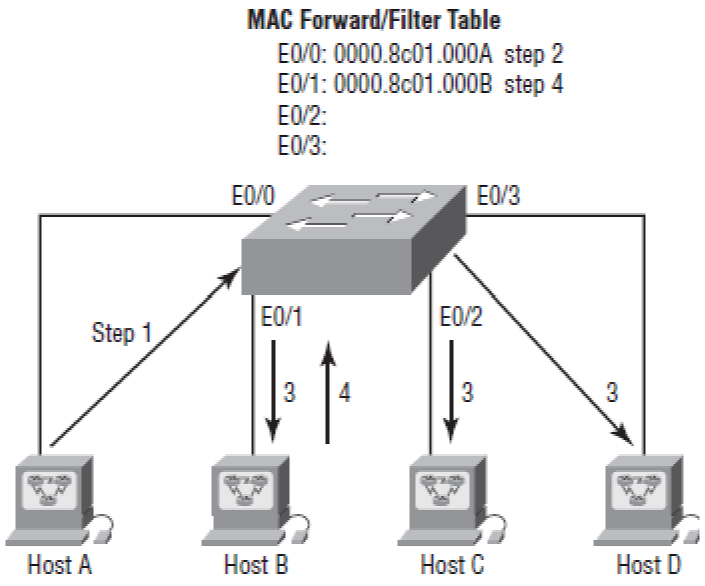


FIGURE 2.7 how switches learn hosts' locations

Let me give you an example of how a forward/filter table is populated:

1. Host A sends a frame to Host B. Host A's MAC address is 0000.8c01.000A; Host B's MAC address is 0000.8c01.000B.
2. The switch receives the frame on the E0/0 interface and places the source address in the MAC address table.
3. Since the destination address is not in the MAC database, the frame is forwarded out all interfaces—except the source port.
4. Host B receives the frame and responds to Host A. The switch receives this frame on interface E0/1 and places the source hardware address in the MAC database.
5. Host A and Host B can now make a point-to-point connection and only the two devices will receive the frames. Hosts C and D will not see the frames, nor are their MAC addresses found in the database because they haven't yet sent a frame to the switch.

If Host A and Host B don't communicate to the switch again within a certain amount of time, the switch will flush their entries from the database to keep it as current as possible.

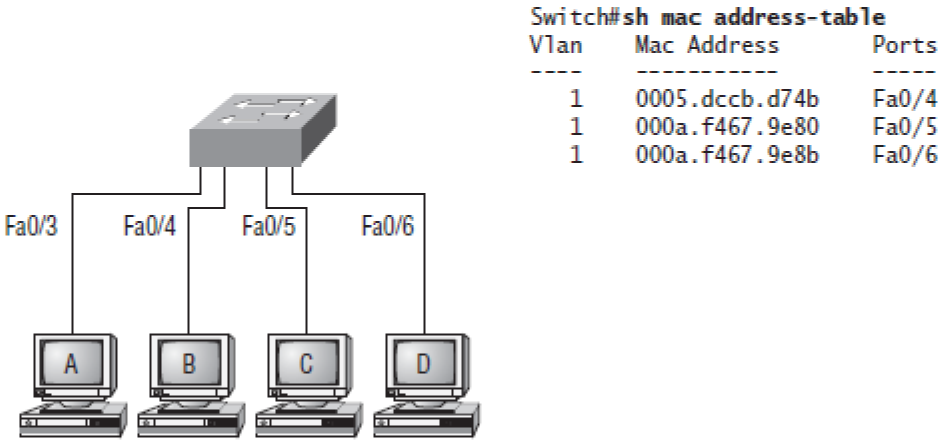
Forward/Filter Decisions

When a frame arrives at a switch interface, the destination hardware address is compared to the forward/filter MAC database. If the destination hardware address is known and listed in the database, the frame is only sent out the correct exit interface. The switch doesn't transmit the frame out any interface except for the destination interface. This preserves bandwidth on the other network segments and is called frame filtering.

But if the destination hardware address is not listed in the MAC database, then the frame is flooded out all active interfaces except the interface the frame was received on. If a device answers the flooded frame, the MAC database is updated with the device’s location (interface).

If a host or server sends a broadcast on the LAN, the switch will flood the frame out all active ports except the source port by default. Remember, the switch creates smaller collision domains, but it’s still one large broadcast domain by default.

In Figure 2.8, Host A sends a data frame to Host D. What will the switch do when it receives the frame from Host A?



```
Switch#sh mac address-table
Vlan    Mac Address       Ports
----    -----------       -----
   1    0005.dccb.d74b    Fa0/4
   1    000a.f467.9e80    Fa0/5
   1    000a.f467.9e8b    Fa0/6
```

FIGURE 2.8 Forward/filter table

Since Host A’s MAC address is not in the forward/filter table, the switch will add the source address and port to the MAC address table and then forward the frame to Host D. If Host D’s MAC address was not in the forward/filter table, the switch would have flooded the frame out all ports except for port Fa0/3.

Now let’s take a look at the output of a show mac address-table:

```
Switch#sh mac address-table
Vlan Mac Address Type Ports
---- ----------- -------- -----
1 0005.dccb.d74b DYNAMIC Fa0/1
1 000a.f467.9e80 DYNAMIC Fa0/3
1 000a.f467.9e8b DYNAMIC Fa0/4
1 000a.f467.9e8c DYNAMIC Fa0/3
1 0010.7b7f.c2b0 DYNAMIC Fa0/3
1 0030.80dc.460b DYNAMIC Fa0/3
1 0030.9492.a5dd DYNAMIC Fa0/1
1 00d0.58ad.05f4 DYNAMIC Fa0/1
```

Suppose the preceding switch received a frame with the following MAC addresses:

```
Source MAC: 0005.dccb.d74b
Destination MAC: 000a.f467.9e8c
```

Configure and verify initial switch configuration including remote access management

Perform and verify initial switch configuration tasks, including remote access management

Cisco Catalyst switches come in many flavors—some run 10Mbps, and some jam all the way up to 10Gbps switched ports with a combination of twisted-pair and fiber. These newer switches (specifically the 2960 and 3560's) have more intelligence, so they can give you data fast—video and voice services, too.

It's time to get down to it—I'm going to show you how to start up and configure a Cisco Catalyst switch using the command-line interface (CLI). After you get the basic commands down, I'll show you how to configure virtual LANs (VLANs) plus Inter-Switch Link (ISL), 802.1q routing, and Cisco's Virtual Trunk Protocol (VTP).

- Administrative functions
- Configuring the IP address and subnet mask
- Setting the IP default gateway
- Setting port security
- Setting PortFast
- Enabling BPDUGuard and BPDUFilter
- Enabling UplinkFast
- Enabling BackboneFast
- Enabling RSTP (802.1w)
- Enabling EtherChannel
- Configuring an STP root switch
- Using the CNA to configure a switch

Catalyst Switch Configuration

Figure 2.9 shows the switched network I'll be working on to show you Cisco's Catalyst switch configurations.

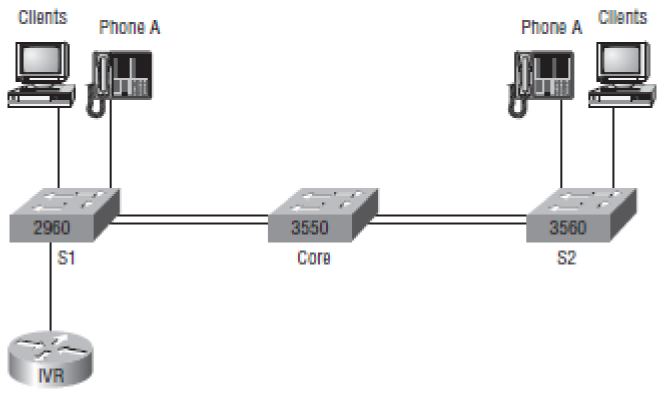


FIGURE 2.9 our switched network

I'm going to use a new 3560, a 2960, and a 3550 switch. But before we actually get into configuring one of the Catalyst switches, I've got to fill you in regarding the boot up process of these switches. Figure 2.10 shows the detail of a typical Cisco Catalyst switch, and I need to tell you about the different interfaces and features of this product.

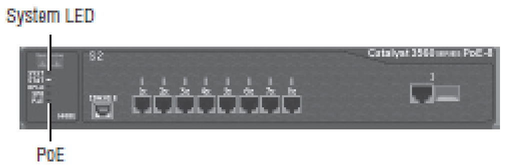
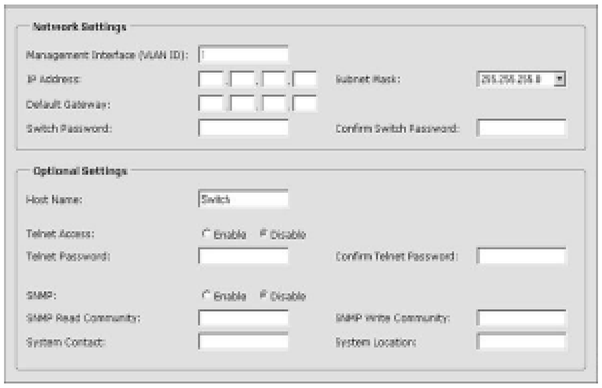


FIGURE 2.10 A Cisco Catalyst switch

The first thing I want you to know is that the console port for the Catalyst switches are typically located on the back of the switch. But on a smaller switch, like the 3560 shown in the figure, the console is right in the front to make it easier to use. (The eight-port 2960 looks exactly the same.) If the POST completes successfully, the system LED turns green; if the POST fails, it will turn amber. And seeing the amber glow is a very bad thing—typically fatal. So, you may just want to keep a spare switch around—especially in case it happens to be a production switch that's croaked! The bottom button is used to show you which lights are providing Power over Ethernet (PoE). You can see this by pressing the Mode button. The PoE is a very nice feature of these switches. It allows me to power my access point and phone by just connecting them into the switch with an Ethernet cable! Sweet.

After a switch boots up, you can use the Express Setup HTTP screen. Figure 2.11 shows the screen you'll get when you connect to a new switch and use 10.0.0.1 in the HTTP field of your browser. Oh, and obviously your host needs to be in the same subnet.



Network Settings

Management Interface (VLAN ID): 1

IP Address: . . . Subnet Mask: 255.255.255.0

Default Gateway:

Switch Password: Confirm Switch Password:

Optional Settings

Host Name: Switch

Telnet Access: ( ) Enable [✓ Disable]

Telnet Password: Confirm Telnet Password:

SNMP: ( ) Enable [✓ Disable]

SNMP Read Community: SNMP Write Community:

System Contact: System Location:

FIGURE 2.11 Express Setup HTTP screen

The screen shows us that we can set some basic functions. To me, it's easier to configure the information from the CLI, which I'll show you next, but this is actually just one of your options. You can configure the IP address, mask, and default gateway of the switch, plus the passwords. You can also configure the management VLAN on, optionally, you can configure the hostname, system contact, and location and set up Telnet access. And last, the Express Setup HTTP screen provides you with some simple help on setting the switch up with SNMP so that your Network Management System (NMS) can find it.

Now if we connect our switches to each other, as shown in Figure 2.12, remember that first we'll need a crossover cable between the switches. My 2960 and 3560 switches auto detects the connection type, so I was able to use straight-through cables. But a 2950 or 3550 switch won't auto-detect the cable type. Different switches have different needs and abilities, so just keep this in mind when connecting your various switches together.

When you first connect the switch ports to each other, the link lights are amber and then turn green indicating normal operation. This is spanning-tree converging, and as you already know, this process takes around 50 seconds with no extensions enabled. But if you connect into a switch port and the switch port LED is alternating green and amber, this means the port is experiencing errors. If this happens, check the host NIC or the cabling.

Okay—let's start our configuration by connecting into a switch and setting the administrative functions. We'll also assign an IP address to theswitch, but this isn't really necessary to make our network function. The only reason we're going to do that is so we can manage/administer it. Let's use a simple IP scheme like 192.168.10.16/28. This mask should be familiar to you!

Check out the following output:

```
Switch>en
Switch#config t
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Switch(config)#hostname S1
S1(config)#enable secret todd
S1(config)#int f0/1
S1(config-if)#description 1st Connection to Core Switch
S1(config-if)#int f0/2
S1(config-if)#description 2nd Connection to Core Switch
S1(config-if)#int f0/3
S1(config-if)#description Connection to HostA
S1(config-if)#int f0/4
S1(config-if)#description Connection to PhoneA
S1(config-if)#int f0/8
S1(config-if)#description Connection to IVR
S1(config-if)#line console 0
S1(config-line)#password console
S1(config-line)#login
S1(config-line)#exit
S1(config)#line vty 0 ?
<1-15> Last Line number
<cr>
S1(config)#line vty 0 15
S1(config-line)#password telnet
S1(config-line)#login
S1(config-line)#int vlan 1
S1(config-if)#ip address 192.168.10.17 255.255.255.240
S1(config-if)#no shut
S1(config-if)#exit
S1(config)#banner motd # This is the S1 switch #
S1(config)#exit
S1(config)#ip default-gateway 192.168.10.30
S1#copy run start
Destination filename [startup-config]? [enter]
Building configuration...
[OK]
S1#
```

The first thing to notice about this is that there's no IP address configured on the switch's interfaces. Since all ports on a switch are enabled by default, there's not so much to configure. The IP address is configured under a logical interface, called a management domain or VLAN. You would typically use the default VLAN 1 to manage a switched network just as we're doing here.

The rest of the configuration is basically the same as the process you go through for router configuration. Remember, no IP addresses on switch interfaces, no routing protocols, and so on. We're performing layer 2 switching at this point, not routing! Also, note that there is no aux port on Cisco switches.

Cisco IOS commands to perform basic switch setup

Verify network status and switch operation using basic utilities (including: ping, traceroute, Telnet, SSH, arp, ipconfig), SHOW & DEBUG commands

Before we move on to determining IP address problems and how to fix them, I just want to mention some basic DOS commands that you can use to help troubleshoot your network from both a PC and a Cisco router (the commands might do the same thing, but they are implemented differently). Packet InterNet Groper (ping) Uses ICMP echo request and replies to test if a node IP stack is initialized and alive on the network.

traceroute Displays the list of routers on a path to a network destination by using TTL time-outs and ICMP error messages. This command will not work from a DOS prompt.
tracert Same command as traceroute, but it's a Microsoft Windows command and will not work on a Cisco router.
arp -a Displays IP-to-MAC-address mappings on a Windows PC.
show ip arp Same command as arp -a, but displays the ARP table on a Cisco router.

Like the commands traceroute and tracert, they are not interchangeable through DOS and Cisco.
ipconfig /all Used only from a DOS prompt, shows you the PC network configuration. Once you've gone through all these steps and used the appropriate DOS commands, if necessary, what do you do if you find a problem? How do you go about fixing an IP address configuration error?

Checking Network Connectivity

You can use the ping and traceroute commands to test connectivity to remote devices, and both of them can be used with many protocols, not just IP.

Using the *Ping* Command

So far, you've seen many examples of pinging devices to test IP connectivity and name resolution using the DNS server. To see all the different protocols that you can use with ping, use the ping ? command like this:

```
Todd2509#ping ?
WORD Ping destination address or hostname
apollo Apollo echo
appletalk Appletalk echo
clns CLNS echo
decnet DECnet echo
ip IP echo
ipx Novell/IPX echo
srb srb echo
tag Tag encapsulated IP echo
vines Vines echo
xns XNS echo
<cr>
```

The ping output displays the minimum, average, and maximum times it takes for a Ping packet to find a specified system and return. Here’s another example:

```
Todd2509#ping todd2509
Translating "todd2509"...domain server (192.168.0.70)[OK]
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.0.121, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max
= 32/32/32 ms
Todd2509#
```

You can see that the DNS server was used to resolve the name, and the device was pinged in 32ms (milliseconds).

Verify network status and switch operation using basic utilities such as ping, telnet and ssh

Using the Traceroute Command

Traceroute (the traceroute command, or trace for short) shows the path a packet takes to get to a remote device. To see the protocols that you can use with traceroute, use the traceroute ? command, do this:

```
Todd2509#traceroute ?
WORD Trace route to destination address or
hostname
appletalk AppleTalk Trace
clns ISO CLNS Trace
ip IP Trace
ipx IPX Trace
oldvines Vines Trace (Cisco)
vines Vines Trace (Banyan)
<cr>
```

The trace command shows the hop or hops that a packet traverses on its way to a remote device. Here’s an example:

```
Todd2509#trace 2501b
Type escape sequence to abort.
Tracing the route to 2501b.lammle.com (172.16.10.2)
1 2501b.lammle.com (172.16.10.2) 16 msec * 16 msec
Todd2509#
```

You can see that the packet went through only one hop to find the destination.

Verifying Cisco Catalyst Switches

The first thing I like to do with any router or switch is to run through the configurations with a show running-config command. Why? Because doing this gives me a really great headshot of each device. Besides, we can run other commands that will still stock us with really good information.

For example, to verify the IP address set on a switch, we can use the show interface command. Here is the output:

```
S1#sh int vlan 1
Vlan1 is up, line protocol is up
Hardware is EtherSVI, address is 001b.2b55.7540 (bia 001b.2b55.7540)
Internet address is 192.168.10.17/28
MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set, reliability 255/255, txload 1/255,
rxload 1/255
[output cut]
show mac address-table
```

Using it displays the forward filter table, also called a content addressable memory (CAM) table. Here's the output from the S1 switch:

```
S1#sh mac address-table
Mac Address Table
Vlan Mac Address Type Ports
---- ----------- -------- -----
All 0100.0ccc.cccc STATIC CPU
All ffff.ffff.ffff STATIC CPU
[output cut]
1 0002.1762.b235 DYNAMIC Po1
1 0009.b79f.c080 DYNAMIC Po1
1 000d.29bd.4b87 DYNAMIC Po1
1 000d.29bd.4b88 DYNAMIC Po1
1 0016.4662.52b4 DYNAMIC Fa0/4
1 0016.4677.5eab DYNAMIC Po1
1 001a.2f52.49d8 DYNAMIC Po1
1 001a.2fe7.4170 DYNAMIC Fa0/8
1 001a.e2ce.ff40 DYNAMIC Po1
1 0050.0f02.642a DYNAMIC Fa0/3
Total Mac Addresses for this criterion: 31
S1#
```

The switches use what are called base MAC addresses that are assigned to the CPU, and the 2960s use 20. From the preceding output, you can see that we have five MAC addresses dynamically assigned to EtherChannel port 1. Ports Fa0/3, Fa0/8, and Fa0/4 only have on MAC address assigned and all ports are assigned to VLAN 1.

Let's take a look at the S2 switch CAM and see what we can find. Keep in mind that the S2 switch doesn't have EtherChannel configured as the S1 switch does, so STP will shut down one of the redundant links to the Core switch:

```
S2#sh mac address-table
Mac Address Table
-------------------------------------------
Vlan Mac Address Type Ports
---- ----------- -------- -----
All 0008.205a.85c0 STATIC CPU
All 0100.0ccc.cccc STATIC CPU
All 0100.0ccc.cccd STATIC CPU
All 0100.0cdd.dddd STATIC CPU
[output cut]
1 0002.1762.b235 DYNAMIC Fa0/3
1 000d.29bd.4b80 DYNAMIC Fa0/1
1 000d.29bd.4b85 DYNAMIC Fa0/1
1 0016.4662.52b4 DYNAMIC Fa0/1
1 0016.4677.5eab DYNAMIC Fa0/4
1 001b.2b55.7540 DYNAMIC Fa0/1
Total Mac Addresses for this criterion: 26
S2#
```

We can see in the preceding output that we have four MAC addresses assigned to Fa0/1. And of course, we can also see that we have one connection for each host on ports 3 and 4. But where's port 2? Since port 2 is a redundant link, STP placed Fa0/2 into blocking mode. I'll get into more about this again in a minute.

You can set a static MAC address in the MAC address table, but like setting static MAC port security, it's a ton of work. But in case you want to do it, here's how it's done:

```
S1#config t
S1(config)#mac-address-table static aaaa.bbbb.cccc vlan 1 int fa0/5
S1(config)#do show mac address-table
Mac Address Table
-------------------------------------------
Vlan Mac Address Type Ports
---- ----------- -------- -----
All 0100.0ccc.cccc STATIC CPU
[output cut]
```

```
1 0002.1762.b235 DYNAMIC Po1
1 0009.b79f.c080 DYNAMIC Po1
1 000d.29bd.4b87 DYNAMIC Po1
1 000d.29bd.4b88 DYNAMIC Po1
1 0016.4662.52b4 DYNAMIC Fa0/4
1 0016.4677.5eab DYNAMIC Po1
1 001a.2f52.49d8 DYNAMIC Po1
1 001a.2fe7.4170 DYNAMIC Fa0/8
1 001a.e2ce.ff40 DYNAMIC Po1
1 0050.0f02.642a DYNAMIC Fa0/3
1 aaaa.bbbb.cccc STATIC Fa0/5
Total Mac Addresses for this criterion: 31
S1(config)#
```

You can see that a static MAC address is now assigned permanently to interface Fa0/5 and that it’s also assigned to VLAN 1 only.

show spanning-tree

By this time you know that the show spanning-tree command is important. With it, you can see who the root bridge is and what our priorities are set to for each VLAN.

Understand that Cisco switches run what is called Per-VLAN Spanning Tree (PVST), which basically means that each VLAN runs its own instance of the STP protocol. If we typed show spanning-tree, we’d receive information for each VLAN, starting with VLAN 1. So, say we’ve got multiple VLANs and we want to see what’s up with VLAN 2—we’d use the command show spanning-tree vlan 2.

Here is an output from the show spanning-tree command from switch S1. Since we are only using VLAN 1, we don’t need to add the VLAN number to the command:

```
S1#sh spanning-tree
VLAN0001
Spanning tree enabled protocol ieee
Root ID Priority 32769
Address 000d.29bd.4b80
Cost 3012
Port 56 (Port-channel1)
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Bridge ID Priority 49153 (priority 49152 sys-id-ext 1)
Address 001b.2b55.7500
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 15
Uplinkfast enabled
Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- ----------
Fa0/3 Desg FWD 3100 128.3 Edge Shr
```

```
Fa0/4 Desg FWD 3019 128.4 Edge P2p
Fa0/8 Desg FWD 3019 128.8 P2p
Po1 Root FWD 3012 128.56 P2p
```

Since we only have VLAN 1 configured, there's no more output for this command, but if we had more, we would get another page for each VLAN configured on the switch. The default priority is 32768, but there's something called the system ID extension (sys-id-ext), which is the VLAN identifier. The Bridge ID priority is incremented by the number of that VLAN. And since we only have VLAN 1, we increment by one to 32769. But understand, by default, BackboneFast raises the default priority to 49152 to prevent that bridge from becoming the root.

Describe how VLANs create logically separate networks and the need for routing between them.

Describe how VLANs create logically separate networks and the need for routing between them Figure 2.12 shows how layer 2 switched networks are typically designed—as flat networks. With this configuration, every broadcast packet transmitted is seen by every device on the network regardless of whether the device needs to receive that data or not.

By default, routers allow broadcasts to occur only within the originating network, while switches forward broadcasts to all segments. Oh, and by the way, the reason it's called a flat network is because it's one broadcast domain, not because the actual design is physically flat.

In Figure 2.12 we see Host A sending out a broadcast and all ports on all switches forwarding it—all except the port that originally received it.

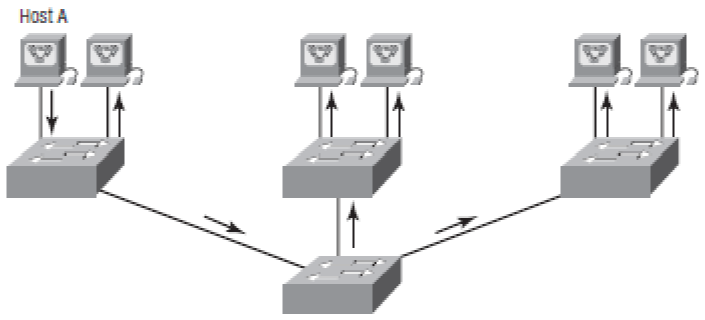


FIGURE 2.12 Flat network structure

Now check out Figure 2.13. It depicts a switched network and shows Host A sending a frame with Host D as its destination. What's important is that, as you can see, that frame is only forwarded out the port where Host D is located. This is a huge improvement over the old hub networks, unless having one collision domain by default is what you really want. (Probably not!)

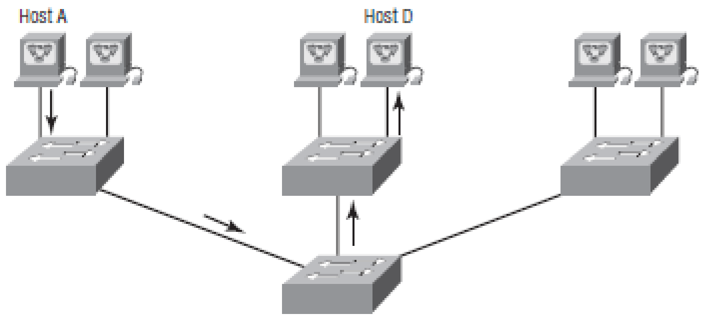


FIGURE 2.13 The benefit of a switched network

Now you already know that the largest benefit you gain by having a layer 2 switched network is that it creates individual collision domain segments for each device plugged into each port on the switch. This scenario frees us from the Ethernet distance constraints, so now larger networks can be built. But often, each new advance comes with new issues. For instance, the larger the number of users and devices, the more broadcasts and packets each switch must handle.

And here's another issue: security! This one's real trouble because within the typical layer 2 switched internetwork, all users can see all devices by default. And you can't stop devices from broadcasting, plus you can't stop users from trying to respond to broadcasts. This means your security options are dismally limited to placing passwords on your servers and other devices.

But wait—there's hope! That is, if you create a virtual LAN (VLAN). You can solve many of the problems associated with layer 2 switching with VLANs, as you'll soon see.

Here's a short list of ways VLANs simplify network management:

- Network adds, moves, and changes are achieved with ease by just configuring a port into the appropriate VLAN.
- A group of users that need an unusually high level of security can be put into its own VLAN so that users outside of the VLAN can't communicate with them.
- As a logical grouping of users by function, VLANs can be considered independent from their physical or geographic locations.
- VLANs greatly enhance network security.
- VLANs increase the number of broadcast domains while decreasing their size.

Coming up, I'm going to tell you all about switching characteristics and thoroughly describe how switches provide us with better network services than hubs can in our networks today.

Broadcast Control

Broadcasts occur in every protocol, but how often they occur depends upon three things:

The type of protocol
The application(s) running on the internetwork
How these services are used

Some older applications have been rewritten to reduce their bandwidth appetites, but there's a new generation of applications that are incredibly bandwidth greedy that will consume any and all they can find. These bandwidth gluttons are multimedia applications that use both broadcasts and multicasts extensively. And faulty equipment, inadequate segmentation, and poorly designed firewalls seriously compound the problems that these broadcast-intensive applications create. All of this has added a major new dimension to network design and presents a bunch of new challenges for an administrator. Positively making sure your network is properly segmented so that you can quickly isolate a single segment's problems to prevent them from propagating throughout your entire internetwork is imperative! And the most effective way to do that is through strategic switching and routing.

Since switches have become more affordable lately, a lot of companies are replacing their flat hub networks with pure switched network and VLAN environments. All devices within a VLAN are members of the same broadcast domain and receive all broadcasts. By default, these broadcasts are filtered from all ports on a switch that aren't members of the same VLAN. This is great because you get all the benefits you would with a switched design without getting hit with all the problems you'd have if all your users were in the same broadcast domain—sweet!

Security
Okay, I know. There's always a catch, though right? Time to get back to those security issues. A flat internetwork's security used to be tackled by connecting hubs and switches with routers. So, it was basically the router's job to maintain security. This arrangement was pretty ineffective for several reasons. First, anyone connecting to the physical network could access the network resources located on that particular physical LAN. Second, all anyone had to do to observe any and all traffic happening in that network was to simply plug a network analyzer into the hub. And similar to that last ugly fact, users could join a workgroup by just plugging their workstations into the existing hub. That's about as secure as an open barrel of honey in a bear enclosure!

But that's exactly what makes VLANs so cool. If you build them and create multiple broadcast groups, you have total control over each port and user! So, the days when anyone could just plug their workstation into any switch port and gain access to network resources are history because now you get to control each port, plus whatever resources that port can access.

What’s more, with the new 2960/3560 switches, this actually happens automatically! And it doesn’t end there, my friends, because VLANs can be created in accordance with the network resources a given user requires, plus switches can be configured to inform a network management station of any unauthorized access to network resources. And if you need inter-VLAN communication, you can implement restrictions on a router to make that happen.

You can also place restrictions on hardware addresses, protocols, and applications.

Now we’re talking security—the honey barrel is now sealed, shrouded in razor wire, and made of solid titanium!

Explain network segmentation and basic traffic management concepts

Flexibility and Scalability
If you were paying attention to what you’ve read so far, you know that layer 2 switches only read frames for filtering—they don’t look at the Network layer protocol. And by default, switches forward all broadcasts. But if you create and implement VLANs, you’re essentially creating smaller broadcast domains at layer 2.

What this means is that broadcasts sent out from a node in one VLAN won’t be forwarded to ports configured to belong to a different VLAN. So, by assigning switch ports or users to VLAN groups on a switch or group of connected switches, you gain the flexibility to add only the users you want into that broadcast domain regardless of their physical location. This setup can also work to block broadcast storms caused by a faulty NIC as well as prevent an intermediate device from propagating broadcast storms throughout the entire internetwork. Those evils can still happen on the VLAN where the problem originated, but the device with the disease will be quarantined to that one ailing VLAN.

Another advantage is that when a VLAN gets too big, you can create more VLANs to keep the broadcasts from consuming too much bandwidth—the fewer users in a VLAN, the fewer users affected by broadcasts. This is all well and good, but you seriously need to keep network services in mind and understand how the users connect to these services when you create your VLAN. It’s a good move to try to keep all services, except for the email and Internet access that everyone needs, local to all users whenever possible.

To understand how a VLAN looks to a switch, it’s helpful to begin by first looking at a traditional network. Figure 2.14 shows how a network was created by using hubs to connect physical LANs to a router.

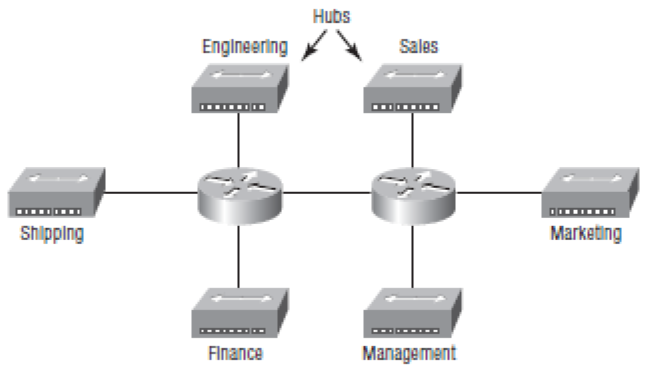


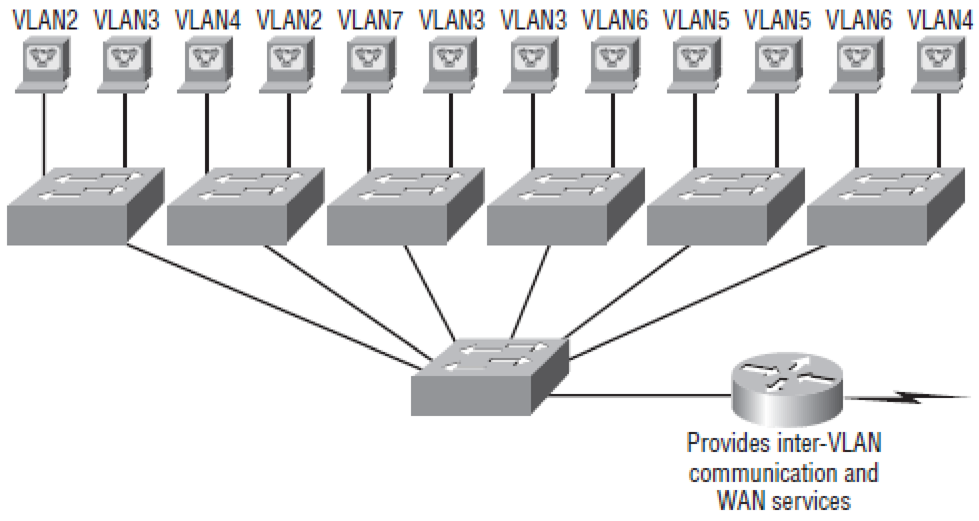
FIGURE 2.14 Physical LANs connected to a router

Here, you can see that each network is attached with a hub port to the router (each segment also has its own logical network number even though this isn't obvious from looking at the figure). Each node attached to a particular physical network has to match that network's number in order to be able to communicate on the internetwork. Notice that each department has its own LAN, so if you needed to add new users to, let's say, Sales, you would just plug them into the Sales LAN and they would automatically be part of the Sales collision and broadcast domain. This design really did work well for many years.

But there was one major flaw: What happens if the hub for Sales is full and we need to add another user to the Sales LAN? Or, what do we do if there's no more physical space where the Sales team is located for this new employee? That new Sales team member will just have to sit on the same side of the building as the Finance people, and we'll just plug the poor soul into the hub for Finance.

Doing this obviously makes the new user part of the Finance LAN, which is very bad for many reasons. First and foremost, we now have a major security issue. Because the new Sales employee is a member of the Finance broadcast domain, the newbie can see all the same servers and access all network services that the Finance folks can. Second, for this user to access the Sales network services that they need to get their job done, they would have to go through the router to log in to the Sales server—not exactly efficient!

Now let's look at what a switch accomplishes for us. Figure 2.15 demonstrates how switches come to the rescue by removing the physical boundary to solve our problem. It also shows how six VLANs (numbered 2 through 7) are used to create a broadcast domain for each department. Each switch port is then administratively assigned a VLAN membership, depending on the host and which broadcast domain it's placed in.



Marketing	VLAN2	172.16.20.0/24
Shipping	VLAN3	172.16.30.0/24
Engineering	VLAN4	172.16.40.0/24
Finance	VLAN5	172.16.50.0/24
Management	VLAN6	172.16.60.0/24
Sales	VLAN7	172.16.70.0/24

FIGURE 2.15 Switches removing the physical boundary

So now, if we needed to add another user to the Sales VLAN (VLAN 7), we could just assign the port to VLAN 7 regardless of where the new Sales team member is physically located— nice! This illustrates one of the sweetest advantages to designing your network with VLANs over the old collapsed backbone design. Now, cleanly and simply, each host that needs to be in the Sales VLAN is merely assigned to VLAN 7. And by using the new switches with the predefined macros, we can just use CNA and Smart ports to configure the port to be a Desktop connection and voilà! The port configuration is simply completed for us.

Notice that I started assigning VLANs with VLAN number 2. The number is irrelevant, but you might be wondering what happened to VLAN 1? Well that VLAN is an administrative VLAN, and even though it can be used for a workgroup, Cisco recommends that you use it for administrative purposes only. You can't delete or change the name of VLAN 1, and by default, all ports on a switch are members of VLAN 1 until you change them.

Since each VLAN is considered a broadcast domain, it's got to also have its own subnet number (refer again to Figure 2.15). And if you're also using IPv6, then each VLAN must also be assigned its own IPv6 network number. So you don't get confused, just keep thinking of VLANs as separate subnets or networks.

Now let's get back to that "because of switches, we don't need routers anymore" misconception. Looking at Figure 2.15, notice that there are seven VLANs, or broadcast domains, counting VLAN 1.

Configure and verify VLANs

Configure, verify, and troubleshoot VLANs
It may come as a surprise to you, but configuring VLANs is actually pretty easy. Figuring out which users you want in each VLAN is not; it's extremely time-consuming. But once you've decided on the number of VLANs you want to create and established which users you want to belong to each one, it's time to bring your first VLAN into the world.

To configure VLANs on a Cisco Catalyst switch, use the global config vlan command. In the following example, I'm going to demonstrate how to configure VLANs on the S1 switch by creating three VLANs for three different departments—again, remember that VLAN 1 is the native and administrative VLAN by default:

```
S1#config t
S1(config)#vlan ?
WORD ISL VLAN IDs 1-4094
internal internal VLAN
S1(config)#vlan 2
S1(config-vlan)#name Sales
S1(config-vlan)#vlan 3
S1(config-vlan)#name Marketing
S1(config-vlan)#vlan 4
S1(config-vlan)#name Accounting
S1(config-vlan)#^Z
S1#
```

From the preceding, you can see that you can create VLANs from 2 to 4094. This is only mostly true. As I said, VLANs can really only be created up to 1005, and you can't use, change, rename, or delete VLANs 1 and 1002 through 1005 because they're reserved. The VLAN numbers above that are called extended VLANs and won't be saved in the database unless your switch is set to VTP transparent mode. You won't see these VLAN numbers used too often in production.
Here's an example of setting my S1 switch to VLAN 4000 when my switch is set to VTP server mode (the default VTP mode):

```
S1#config t
S1(config)#vlan 4000
S1(config-vlan)#^Z
% Failed to create VLANs 4000
Extended VLAN(s) not allowed in current VTP mode.
%Failed to commit extended VLAN(s) changes.
```

After you create the VLANs that you want, you can use the show vlan command to check them out. But notice that, by default, all ports on the switch are in VLAN 1. To change the VLAN associated with a port, you need to go to each interface and tell it which VLAN to be a part of.

Once the VLANs are created, verify your configuration with the show vlan command (sh vlan for short):

```
S1#sh vlan
VLAN Name Status Ports
---- ------------------------------------------------------------
1 default active Fa0/3, Fa0/4, Fa0/5, Fa0/6
Fa0/7, Fa0/8, Gi0/1
2 Sales active
3 Marketing active
4 Accounting active
[output cut]
```

This may seem repetitive, but it's important, and I want you to remember it: You can't change, delete, or rename VLAN 1 because it's the default VLAN and you just can't change that—period. It's the native VLAN of all switches by default, and Cisco recommends that you use it as your administrative VLAN. Basically, any packets that aren't specifically assigned to a different VLAN will be sent down to the native VLAN.

In the preceding S1 output, you can see that ports Fa0/3 through Fa0/8 and the Gi0/1 uplink are all in VLAN 1, but where are ports 1 and 2? Ports one and two are trunked. Any port that is a trunk port won't show up in the VLAN database. You have to use the show interface trunk command to see your trunked ports.

Assigning Switch Ports to VLANs

You configure a port to belong to a VLAN by assigning a membership mode that specifies the kind of traffic the port carries, plus the number of VLANs to which it can belong. You can configure each port on a switch to be in a specific VLAN (access port) by using the interface switchport command. You can also configure multiple ports at the same time with the interface range command.

Remember that you can configure either static memberships or dynamic memberships on a port. I'm only going to cover the static flavor. In the following example, I'll configure interface fa0/3 to VLAN 3. This is the connection from the S1 switch to the HostA device:

```
S1#config t S1(config)#int fa0/3
S1(config-if)#switchport ?
access Set access mode characteristics of the interface
backup Set backup for the interface
```

```
block Disable forwarding of unknown uni/multi cast addresses
host Set port host
mode Set trunking mode of the interface
nonegotiate Device will not engage in negotiation protocol on this
interface
port-security Security related command
priority Set appliance 802.1p priority
protected Configure an interface to be a protected port
trunk Set trunking characteristics of the interface
voice Voice appliance attributes
```

Let's start with setting an access port on S1, which is probably the most widely used type of port on production switches that has VLANs configured:

```
S1(config-if)#switchport mode ?
access Set trunking mode to ACCESS unconditionally
dynamic Set trunking mode to dynamically negotiate access or
trunk mode
trunk Set trunking mode to TRUNK unconditionally
S1(config-if)#switchport mode access
S1(config-if)#switchport access vlan 3
```

By starting with the switch port mode access command, you're telling the switch that this is a layer 2 port. You can then assign a VLAN to the port with the switch port access command. Remember, you can choose many ports to configure at the same time if you use the interface range command. The dynamic and trunk commands are used for trunk ports exclusively. That's it. Well, sort of. If you plugged devices into each VLAN port, they can only talk to other devices in the same VLAN. We want to enable inter-VLAN communication, and we're going to do that, but first you need to learn a bit more about trunking.

Configure and verify trunking on Cisco switches

Configure, verify, and troubleshoot trunking on Cisco switches

The 2960 switch only runs the IEEE 802.1Q encapsulation method. To configure trunking on a Fast Ethernet port, use the interface command trunk [parameter].

The following switch output shows the trunk configuration on interface fa0/8 as set to trunk on:

```
S1#config t
S1(config)#int fa0/8
S1(config-if)#switchport mode trunk
```

The following list describes the different options available when configuring a switch interface:
switch-port mode access but this puts the interface (access port) into permanent non-trunking mode and negotiates to convert the link into a non-trunk link. The interface becomes a non-trunk interface regardless of whether the neighboring interface is a trunk interface. The port would be a dedicated layer 2 port.
switchport mode dynamic auto This mode makes the interface able to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to trunk or desirable mode. This is now the default switchport mode for all Ethernet interfaces on all new Cisco switches.

switchport mode dynamic desirable This one makes the interface actively attempt to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to trunk, desirable, or auto mode. I used to see this mode as the default on some older switches, but not any longer. The default is dynamic auto now.

switchport mode trunk Puts the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link. The interface becomes a trunk interface even if the neighboring interface isn't a trunk interface.

switchport nonegotiate Prevents the interface from generating DTP frames. You can use this command only when the interface switchport mode is access or trunk. You must manually configure the neighboring interface as a trunk interface to establish a trunk link.

Trunking with the Cisco Catalyst 3560 Switch
Okay, let's take a look at one more switch—the Cisco Catalyst 3560. The configuration is pretty much the same as it is for a 2960, with the exception that the 3560 can provide layer 3 services and the 2960 can't. Plus, the 3560 can run both the ISL and the IEEE 802.1Q trunking encapsulation methods—the 2960 can only run 802.1Q. With all this in mind, let's take a quick look at the VLAN encapsulation difference regarding the 3560 switch.

The 3560 has the encapsulation command, which the 2960 switch doesn't:

```
Core(config-if)#switchport trunk encapsulation ?
dot1q Interface uses only 802.1q trunking encapsulation
when trunking
isl Interface uses only ISL trunking encapsulation
when trunking
negotiate Device will negotiate trunking encapsulation with peer on
interface
Core(config-if)#switchport trunk encapsulation dot1q
Core(config-if)#switchport mode trunk
```

As you can see, we've got the option to add either the IEEE 802.1Q (dot1q) encapsulation or the ISL encapsulation to the 3560 switch. After you set the encapsulation, you still have to set the interface mode to trunk. Honestly, it's pretty rare that you'd continue to use the ISL encapsulation method. Cisco is moving away from ISL—its new routers don't even support it.

Defining the Allowed VLANs on a Trunk

As I've mentioned, trunk ports send and receive information from all VLANs by default, and if a frame is untagged, it's sent to the management VLAN. This applies to the extended range VLANs as well.

But we can remove VLANs from the allowed list to prevent traffic from certain VLANs from traversing a trunked link. Here's how you'd do that:

```
S1#config t
S1(config)#int f0/1
S1(config-if)#switchport trunk allowed vlan ?
WORD VLAN IDs of the allowed VLANs when this port is in
trunking mode
add add VLANs to the current list
all all VLANs
except all VLANs except the following
none no VLANs
remove remove VLANs from the current list
S1(config-if)#switchport trunk allowed vlan remove ?
WORD VLAN IDs of disallowed VLANS when this port is in trunking mode
S1(config-if)#switchport trunk allowed vlan remove 4
```

The preceding command stopped the trunk link configured on S1 port f0/1, causing it to drop all traffic sent and received for VLAN 4. You can try to remove VLAN 1 on a trunk link, but it will still send and receive management like CDP, PAgP, LACP, DTP, and VTP, so what's the point? To remove a range of VLANs, just use a hyphen:

```
S1(config-if)#switchport trunk allowed vlan remove 4-8
```

If by chance someone has removed some VLANs from a trunk link and you want to set the trunk back to default, just use this command:

```
S1(config-if)#switchport trunk allowed vlan all
Or this command to accomplish the same thing:
S1(config-if)#no switchport trunk allowed vlan
```

Next, I want to show you how to configure pruning for VLANs before we start routing between VLANs.

Changing or Modifying the Trunk Native VLAN
You really don't want to change the trunk port native VLAN from VLAN 1, but you can, and some people do it for security reasons. To change the native VLAN, use the following command:

```
S1#config t
S1(config)#int f0/1
S1(config-if)#switchport trunk ?
allowed Set allowed VLAN characteristics when interface is
in trunking mode
native Set trunking native characteristics when interface
is in trunking mode
pruning Set pruning VLAN characteristics when interface is
in trunking mode
S1(config-if)#switchport trunk native ?
vlan Set native VLAN when interface is in trunking mode
S1(config-if)#switchport trunk native vlan ?
<1-4094> VLAN ID of the native VLAN when this port is in
trunking mode
S1(config-if)#switchport trunk native vlan 40
S1(config-if)#^Z
```

So, we've changed our native VLAN on our trunk link to 40, and by using the show running-config command, we can see the configuration under the trunk link:

```
!
interface FastEthernet0/1
switchport trunk native vlan 40
switchport trunk allowed vlan 1-3,9-4094
switchport trunk pruning vlan 3,4
!
```

Hold on there partner! You didn't think it would be this easy and would just start working, did you? Sure you didn't. Here's the rub: If all switches don't have the same native VLAN configured on the trunk links, then we'll start to receive this error:

```
19:23:29: %CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch
discovered on FastEthernet0/1 (40), with Core FastEthernet0/7 (1).
19:24:29: %CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch
discovered on FastEthernet0/1 (40), with Core FastEthernet0/7 (1).
Actually, this is a good, noncryptic error, so either we go to the other end
of our trunk
link(s) and change the native VLAN or we set the native VLAN back to the
default. Here's
how we'd do that:
S1(config-if)#no switchport trunk native vlan
```

DTP

Dynamic Trunking Protocol (DTP) is used for negotiating trunking on a link between two devices, as well as negotiating the encapsulation type of either 802.1Q or ISL. I use the non-egotiate command when I want dedicated trunk bports no questions asked.

Auto negotiation

Identify, prescribe, and resolve common switched network media issues, configuration issues, auto negotiation, and switch hardware failures

A network port, also called an RJ-45 port, connects a computer to a network or VLAN. The connection speed depends on the type of network port. Standard Ethernet can transmit up to 10Mbps; however, it is very common to have Fast Ethernet which can transmit up to 100 Mbps. Gigabit Ethernet ports can transmit up to 1000 Mbps. The maximum length of network cable is 328 feet (100 meters).

Twisted-pair is a type of copper cabling that started in telephone communications and now is used in both telephony and most Ethernet networks. A pair of wires forms a circuit that can transmit data. The pair is twisted to provide protection against crosstalk, which is the noise generated by adjacent pairs of wires in the cable.

Common issues with cabling on a switched network include basic switch configuration issues, negotiating both the speed and duplex of a link from a PC to a switch, and the uncommon switch hardware failures.

The most common switch configuration error is not having a port configured into the correct switch membership. By using the show running-config command or show vlan command, you can easily see the port memberships. Always check your VLAN memberships when troubleshooting a command switch issue.

At times, you may find a host is not communicating to a switch because of mismatched speed or duplex issues. This is not as much a problem as it has been in the past because of the better hardware being produced, but it still may show up from time to time. The default on a switch and host is to use 100Mbps full-duplex. If your host or switch port does not support this configuration, you can configure the switch port with the duplex and speed command.

The port LED will be green when everything is OK, however, it will be amber if the port is blocked by STP, and it will turn from green to amber when the port experiences errors. Switches are made pretty resilient today; however, if you boot a switch and the POST completes successfully, the system LED turns green; if the POST fails, it will turn amber. And seeing the amber glow is a very bad thing—typically fatal.

IP addressing (IPv4 / IPv6)

Describe the operation and necessity of using private and public IP addresses for IPv4 addressing

Describe the operation and benefits of using private and public IP addressing
One of the most important topics in any discussion of TCP/IP is IP addressing. An IP address is a numeric identifier assigned to each machine on an IP network. It designates the specific location of a device on the network.

An IP address is a software address, not a hardware address—the latter is hard-coded on a network interface card (NIC) and used for finding hosts on a local network. IP addressing was designed to allow hosts on one network to communicate with a host on a different network regardless of the type of LANs the hosts are participating in.

Before we get into the more complicated aspects of IP addressing, you need to understand some of the basics. First, I'm going to explain some of the fundamentals of IP addressing and its terminology.

IP Terminology
Bit A bit is one digit, either a 1 or a 0.
Byte A byte is 7 or 8 bits, depending on whether parity is used.
Octet An octet, made up of 8 bits, is just an ordinary 8-bit binary number.

Network address
This is the designation used in routing to send packets to a remote network— for example, 10.0.0.0, 172.16.0.0, and 192.168.10.0.

Broadcast address
The address used by applications and hosts to send information to all nodes on a network is called the broadcast address

. Examples include 255.255.255.255, which is all networks, all nodes; 172.16.255.255, which is all subnets and hosts on network 172.16.0.0; and 10.255.255.255, which broadcasts to all subnets and hosts on network 10.0.0.0.

Class A Addresses

The designers of the IP address scheme said that the first bit of the first byte in a Class A network address must always be off, or 0. This means a Class A address must be between 0 and 127 inclusive. Consider the following network address:

```
0
xxxxxxx
If we turn the other 7 bits all off and then turn them all on, we'll find the
Class A range of
network addresses:
0
0000000 = 0
0
1111111 = 127
```

So, a Class A network is defined in the first octet between 0 and 127, and it can't be less or more. (I'll talk about illegal addresses in a minute.)

In a Class A network address, the first byte is assigned to the network address, and the three remaining bytes are used for the node addresses. The Class A format is: network.node.node.node For example, in the IP address 49.22.102.70, the 49 is the network address, and 22.102.70 is the node address. Every machine on this particular network would have the distinctive network address of 49.

Class A network addresses are 1 byte long, with the first bit of that byte reserved and the seven remaining bits available for manipulation (addressing). As a result, the maximum number of Class A networks that can be created is 128. Why? Because each of the seven bit positions can either be a 0 or a 1, thus 2 7 or 128.

To complicate matters further, the network address of all 0s (0000 0000) is reserved to designate the default route. Additionally, the address 127, which is reserved for diagnostics, can't be used either, which means that you can really only use the numbers 1 to 126 to designate Class A network addresses. This means the actual number of usable Class A network addresses is 128 minus 2, or 126.

Each Class A address has three bytes (24-bit positions) for the node address of a machine. This means there are 2 24 —or 16,777,216—unique combinations and, therefore, precisely that many possible unique node addresses for each Class A network. Because node addresses with the two patterns of all 0s and all 1s are reserved, the actual maximum usable number of nodes for a Class A network is 2 24 minus 2, which equals 16,777,214. Either way, that's a huge number of hosts on a network segment!

Class B Addresses
In a Class B network, the RFCs state that the first bit of the first byte must always be turned on, but the second bit must always be turned off. If you turn the other 6 bits all off and then all on, you will find the range for a Class B network:

10
000000 = 128
10
111111 = 191

As you can see, this means that a Class B network is defined when the first byte is configured from 128 to 191.

In a Class B network address, the first 2 bytes are assigned to the network address, and the remaining 2 bytes are used for node addresses. The format is:

network.network.node.node

For example, in the IP address 172.16.30.56, the network address is 172.16, and the node address is 30.56.

With a network address being 2 bytes (8 bits each), there would be 2 16 unique combinations. But the Internet Protocol designers decided that all Class B network addresses should start with the binary digit 1, then 0. This leaves 14 bit positions to manipulate, and therefore 16,384 (that is, 2 14) unique Class B network addresses.

A Class B address uses two bytes for node addresses. This is 2 16 minus the two reserved patterns (all 0s and all 1s), for a total of 65,534 possible node addresses for each Class B network.

Class C Addresses
For Class C networks, the RFCs define the first two bits of the first octet always turned on, but the third bit can never be on. Following the same process as the previous classes, convert from binary to decimal to find the range. Here's the range for a Class C network:

110

00000 = 192
110
11111 = 223

So, if you see an IP address that starts at 192 and goes to 223, you'll know it is a Class C IP address.

The first 3 bytes of a Class C network address are dedicated to the network portion of the address, with only one measly byte remaining for the node address. The format is network.network.network.node

Using the example IP address 192.168.100.102, the network address is 192.168.100, and the node address is 102.

In a Class C network address, the first three bit positions are always the binary 110. The calculation is such: 3 bytes, or 24 bits, minus 3 reserved positions, leaves 21 positions. Hence, there are 2 21 , or 2,097,152, possible Class C networks.

Each unique Class C network has 1 byte to use for node addresses. This leads to 2 8 or 256, minus the two reserved patterns of all 0s and all 1s, for a total of 254 node addresses for each Class C network.

Network Addresses: Special Purpose
Some IP addresses are reserved for special purposes, so network administrators can't ever assign these addresses to nodes. Table 3.1 lists the members of this exclusive little club and why they're included in it.

TABLE 3.1 Reserved IP Addresses

Address	Function
Network address of all 0s	Interpreted to mean "this network or segment."
Network address of all 1s	Interpreted to mean "all networks."
Network 127.0.0.0	Reserved for loopback tests. Designates the local node and allows that node to send a test packet to itself without generating network traffic.
Node address of all 0s	Interpreted to mean "network address." or any host on specified network.
Node address of all 1s	Interpreted to mean "all nodes" on the specified network; for example, 128.2.255.255 means "all nodes" on network 128.2 (Class B address).
Entire IP address set to all 0s	Used by Cisco routers to designate the default route. Could also mean "any network."
Entire IP address set to all 1s (same as 255.255.255.255)	Broadcast to all nodes on the current network; sometimes called an "all 1s broadcast" or limited broadcast.

Private IP Addresses

The people who created the IP addressing scheme also created what we call private IP addresses. These addresses can be used on a private network, but they're not routable through the Internet. This is designed for the purpose of creating a measure of much-needed security, but it also conveniently saves valuable IP address space.

If every host on every network had to have real routable IP addresses, we would have run out of IP addresses to hand out years ago. But by using private IP addresses, ISPs, corporations, and home users only need a relatively tiny group of bona fide IP addresses to connect their networks to the Internet. This is economical because they can use private IP addresses on their inside networks and get along just fine.

To accomplish this task, the ISP and the corporation—the end user, no matter who they are—need to use something called Network Address Translation (NAT), which basically takes a private IP address and converts it for use on the Internet. Many people can use the same real IP address to transmit out onto the Internet. Doing things this way saves megatons of address space—good for us all!

So, What Private IP Address Should I Use?

That's a really great question: Should you use Class A, Class B, or even Class C private addressing when setting up your network? Let's take Acme Corporation in SF as an example.

This company is moving into a new building and needs a whole new network (what a treat this is!). It has 14 departments, with about 70 users in each. You could probably squeeze one or two Class C addresses to use, or maybe you could use a Class B, or even a Class A just for fun.

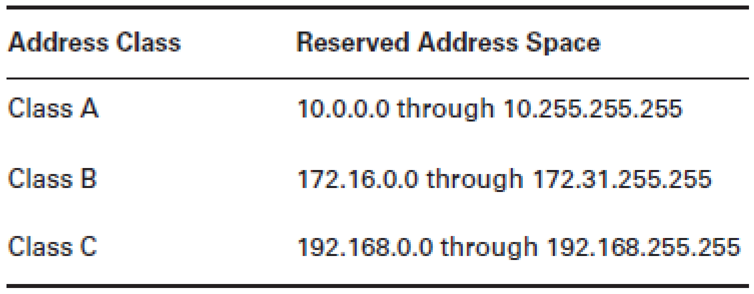
The rule of thumb in the consulting world is, when you're setting up a corporate network—regardless of how small it is—you should use a Class A network address because it gives you the most flexibility and growth options. For example, if you used the 10.0.0.0 network address with a /24 mask, then you'd have 65,536 networks, each with 254 hosts. Lots of room for growth with that network!

But if you're setting up a home network, you'd opt for a Class C address because it is the easiest for people to understand and configure. Using the default Class C mask gives you one network with 254 hosts—plenty for a home network.

With the Acme Corporation, a nice 10.1.

X .0 with a /24 mask (the X is the subnet for each department) makes this easy to design, install, and troubleshoot. The reserved private addresses are listed in Table 3.2.

TABLE 3 . 2 Reserved IP Address Space



Address Class	Reserved Address Space
Class A	10.0.0.0 through 10.255.255.255
Class B	172.16.0.0 through 172.31.255.255
Class C	192.168.0.0 through 192.168.255.255

Identify the appropriate IPv6 addressing scheme to satisfy addressing requirements in a LAN/WAN environment

IPv6 Address Plan Considerations

IPv6 provides a significantly larger address space than IPv4, which enables lots of flexibility in how you can define logical and practical addressing plans. You can assign subnet prefixes based on various logical schemes that involve factors detailed in the IP Addressing Guide and on additional factors that pertain to IPv6 such as:

- Using existing IP addressing schemes
  - Translating the existing subnet numbers into IPv6 subnet IDs
  - Translating the VLAN IDs into IPv6 subnet IDs
- Redesigning your IP addressing scheme
  - Allocating IPv6 addresses according to your needs

When redesigning IP addressing schemes, you can allocate according to your need. Such logical addressing plans have the potential to simplify network operations and service offerings as well as network management and troubleshooting.

The addressing plan must take the following into consideration:

- **Prefix aggregation**: The large IPv6 addresses can lead to large routing tables unless network designers actively pursue aggregation.
- **Network growth**: It is important to design the address infrastructure to take network growth into account
- **Use of Unique Local Addresses (RFC4193)**: As in IPv4, there is private address space within IPv6. The main difference is that in IPv4 every organization has the same private address space to choose from. In IPv6 the address space is just for the one network and is globally unique. This private address space can be used to address devices and services that do not need to connect to the Internet.

Prefix Sizing Considerations
The IPv6 specification prescribes a /64 prefix length for the normal IPv6 unicast addresses. Because there is a very large address space available for IPv6, you may want to use a different prefix length than /64.

A prefix length other than a /64 in IPv6 will break the operation of the following technologies:

- Neighbor Discovery (ND)
- Secure Neighbor Discovery (SEND) [RFC3971]
- Privacy extensions [RFC4941]
- Parts of Mobile IPv6 [RFC4866]
- Protocol Independent Multicast - Sparse Mode (PIM-SM) with Embedded-RP [RFC3956]
- Site Multihoming by IPv6 Intermediation (SHIM6) [SHIM6]

The /64 Prefix
The 64-bit prefix should be used for the traditional LAN/WAN interfaces of network devices.

The /126 Prefix

The 126-bit prefix is typically used for point-to-point links similar to the IPv4 address-conservative /30 allocation for point-to-point links. However, the address space in IPv6 is significantly larger than the IPv4 address space.

The general recommendation is to use /64 on point-to-point links.

The /127 Prefix
Using the /127 prefix, the equivalent of the IPv4 /31 on point-to-point links (RFC 3021), is considered harmful according to RFC3627. This allocation is just like the /126 allocation for a point-to-point link, but is driven by address conservation. For operational simplicity, consider using the /64 prefix for point-to-point links.

The /128 Prefix
The 128-bit prefix may be used in those situations where one address is required. An example of this type of address is the loopback address of a network device.

IPv6 Address Space Assignments for Internet Connectivity
IPv6 is not very different from IPv4. For an agency to connect to the Internet using IPv6 addresses, it must acquire a block of IPv6 addresses from the routable Internet space. IPv6 address blocks are distributed just as IPv4 bl Provider-assigned (PA) address space is not portable between service providers and stays in the service provider's region. Service providers may include provider- assigned address space as part of the service. Unless the service itself is multi-homed, PA address space is sufficient. If an agency is multi-homed, it should procure provider-independent (PI) address space from its Regional Internet Registry. Different registries have different policies and cost structures relating to PI address space.

IPv6 Transition Technologies
The success of IPv6 originally was thought to depend on the new applications that run over it. However, it is becoming very clear that the exhaustion of IPv4 will ultimately end up being the driver for IPv6 adoption. A key part of any good IPv6 design is its ability to integrate into and coexist with existing IPv4 networks. IPv4 and IPv6 hosts need to coexist for a substantial length of time during the steady migration from IPv4 to IPv6, and the development of transition strategies, tools, and mechanisms has been part of the basic IPv6 design from the start.

There are three IPv6 transition technologies: dual-stack, tunneling, and translation.

Dual Stack
Dualstack is the basic strategy to use for large agencies that are adopting IPv6. It involves configuring devices to be able to run IPv4 and IPv6 simultaneously. IPv4 communication uses the IPv4 protocol stack, and IPv6 communication uses the IPv6 protocol stack.

Applications choose between using IPv4 or IPv6 based on the response to DNS requests. The application selects the correct address based on the type of IP traffic. Because dual stack allows hosts to simultaneously reach existing IPv4 content and IPv6 content as it becomes available, dual stack offers a very flexible adoption strategy. However, because IPv4 addresses are still required, dual stack is not a long-term solution to address exhaustion.

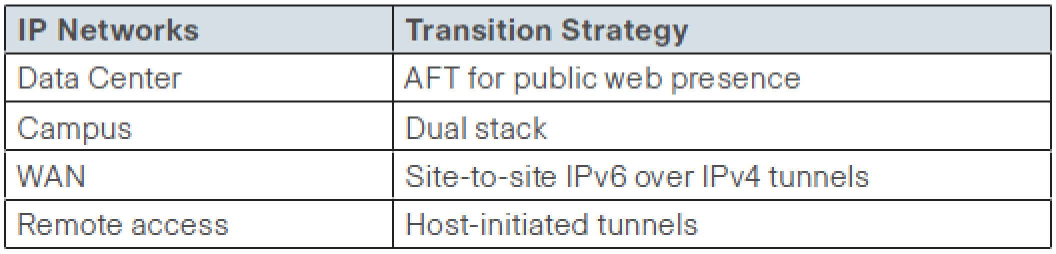
Dual stack also avoids the need to translate between protocol stacks. Translation is a valid adoption mechanism, but it introduces operational complexity and lower performance. Because a host automatically selects the right transport to use to reach a destination based on DNS information, there should not be a need to translate between an IPv6 host and an IPv4 server.

Tunneling
Tunnels encapsulate IPv6 traffic within IPv4 packets, and are primarily used for communication between IPv6 (or dual-stack) sites or for connection to remote IPv6 networks or hosts over an IPv4 backbone. There are many different tunneling techniques, including 6to4, ISATAP, Teredo, 6PE, 6VPE, and mGRE v6 over v4. Tunnels may be manually configured or automatically configured. Most modern operating systems include support for tunneling in addition to dual stack.

Translation
Address Family Translation (AFT) is the process of translating addresses from one address-family to another. During the adoption phase, AFT is primarily used to translate between IPv6 hosts and IPv4 content. AFT may be stateless, where reserved portions of the IPv6 address space are automatically mapped to IPv4, or it may be stateful, with addresses from a configured range used to map packets between address families.

Nearly all enterprise deployments of IPv6 use dual-stack internally. Dual stack offers a non-disruptive way to learn about and gain operational experience with a new address-family, which is an important part of successfully managing the transition. Pilots and trials depend on specific requirements. An example strategy is shown below in table 3.1.

Table 3.1 Sample Transition Strategies



IP Networks	Transition Strategy
Data Center	AFT for public web presence
Campus	Dual stack
WAN	Site-to-site IPv6 over IPv4 tunnels
Remote access	Host-initiated tunnels

Identify the appropriate IPv4 addressing scheme using VLSM and summarization to satisfy addressing requirements in a LAN/WAN environment.

Determine the appropriate classless addressing scheme using VLSM and summarization to satisfy addressing requirements in a LAN/WAN environment

To create VLSMs quickly and efficiently, you need to understand how block sizes and charts work together to create the VLSM masks. Table 3.3 shows you the block sizes used when creating VLSMs with Class C networks. For example, if you need 25 hosts, then you'll need a block size of 32. If you need 11 hosts, you'll use a block size of 16. Need 40 hosts? Then you'll need a block of 64. You cannot just make up block sizes—they've got to be the block sizes shown in Table 3.2. So, memorize the block sizes in this table—it's easy. They're the same numbers we used with sub netting!

TABLE 3.2 Block Sizes

Prefix	Mask	Hosts	Block Size
/25	128	126	128
/26	192	62	64
/27	224	30	32
/28	240	14	16
/29	248	6	8
/30	252	2	4

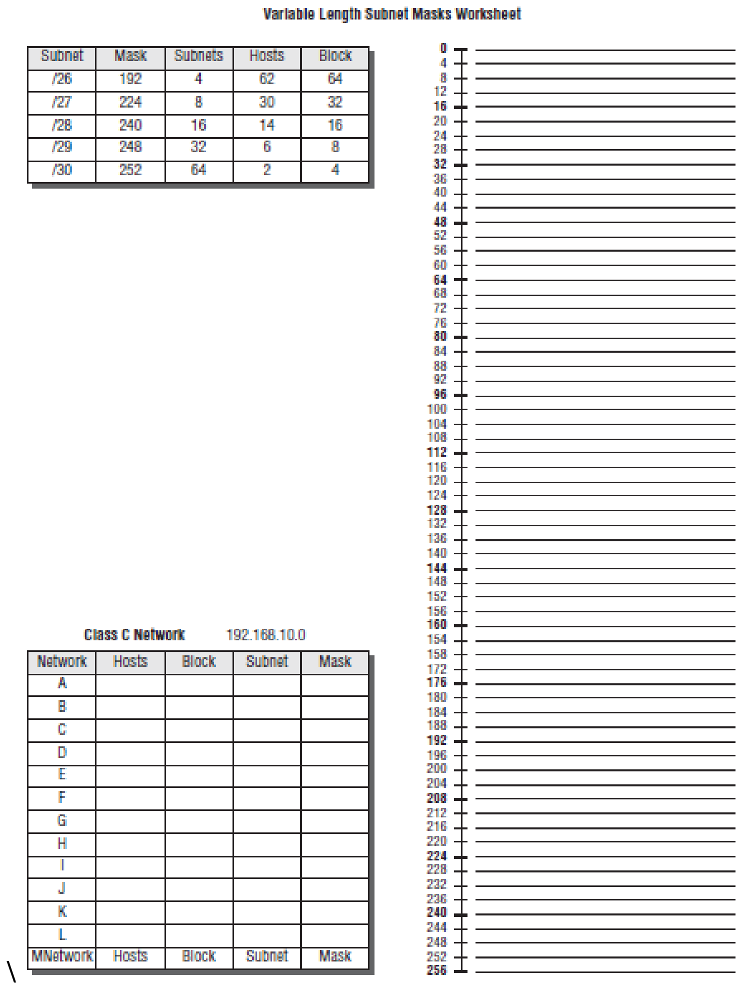
The next step is to create a VLSM table. Figure 3.3 shows you the table used in creating a VLSM network. The reason that we use this table is so that we don't accidentally overlap networks. You'll find the sheet shown in Figure 3.3 very valuable because it lists every block size you can use for a network address. Notice that the block sizes are listed starting from a block size of 4 all the way to a block size of 128. If you have two networks with block sizes of 128, you'll quickly see that you can have only two networks. With a block size of 64, you can have only four networks, and so on, all the way to having 64 networks if you use only block sizes of 4.

Remember that this takes into account that you are using the command ip subnet-zero in your network design. Now, just fill in the chart in the lower-left corner, and then add the subnets to the worksheet and you're good to go.

So, let’s take what we’ve learned so far about our block sizes and VLSM table and create a VLSM using a Class C network address 192.168.10.0 for the network in Figure 3.4. Then fill out the VLSM table, as shown in Figure 3.5.

In Figure 3.4, we have four WAN links and four LANs connected together. We need to create a VLSM network that will allow us to save address space. Looks like we have two block sizes of 32, a block size of 16, and a block size of 8, and our WANs each have a block size of 4. Take a look and see how I filled out our VLSM chart in Figure 3.5.

We still have plenty of room for growth with this VLSM network design. We never could accomplish that with one subnet mask using classful routing. Let’s do another one. Figure 3.6 shows a network with 11 networks, two block sizes of 64, one of 32, five of 16, and three of 4.



Variable Length Subnet Masks Worksheet

Subnet	Mask	Subnets	Hosts	Block
/26	192	4	62	64
/27	224	8	30	32
/28	240	16	14	16
/29	248	32	6	8
/30	252	64	2	4

Class C Network 192.168.10.0

Network	Hosts	Block	Subnet	Mask
A				
B				
C				
D				
E				
F				
G				
H				
I				
J				
K				
L				
MNetwork	Hosts	Block	Subnet	Mask

0
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76
80
84
88
92
96
100
104
108
112
116
120
124
128
132
136
140
144
148
152
156
160
154
158
172
176
180
184
188
192
196
200
204
208
212
216
220
224
228
232
236
240
244
248
252
256

\

FIGURE 3.3 The VLSM table

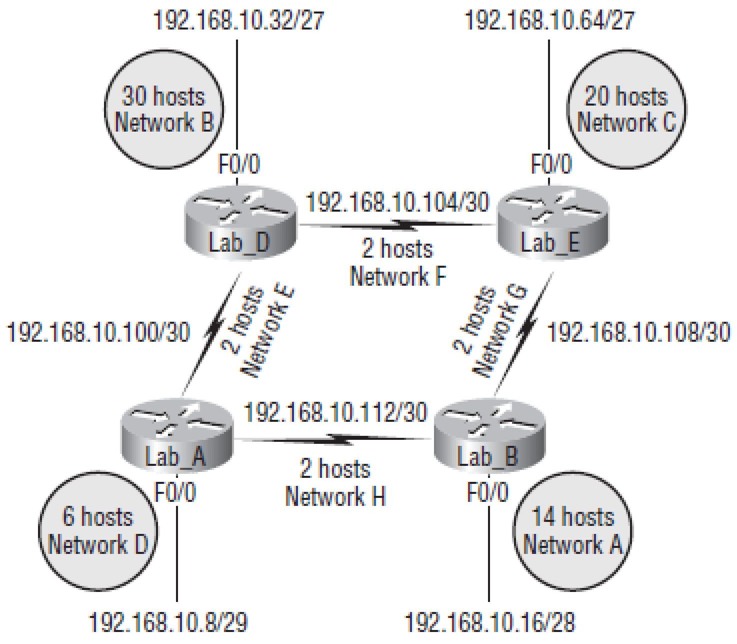


FIGURE 3.4 A VLSM network, example one

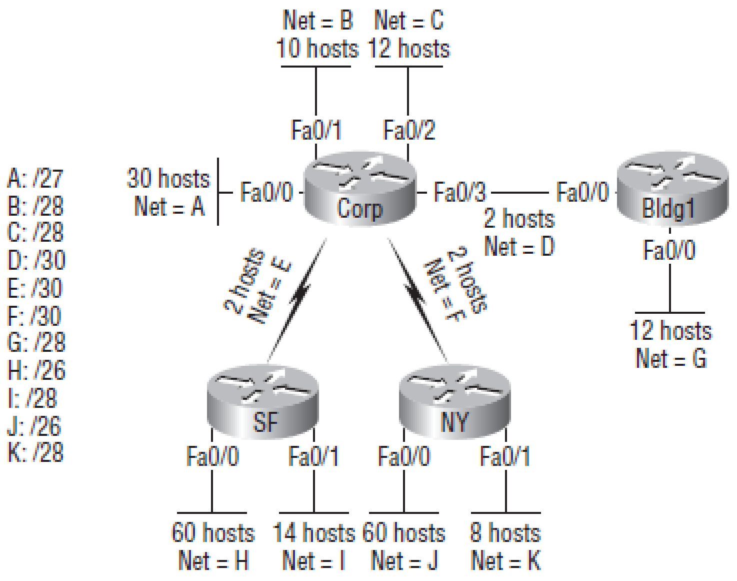
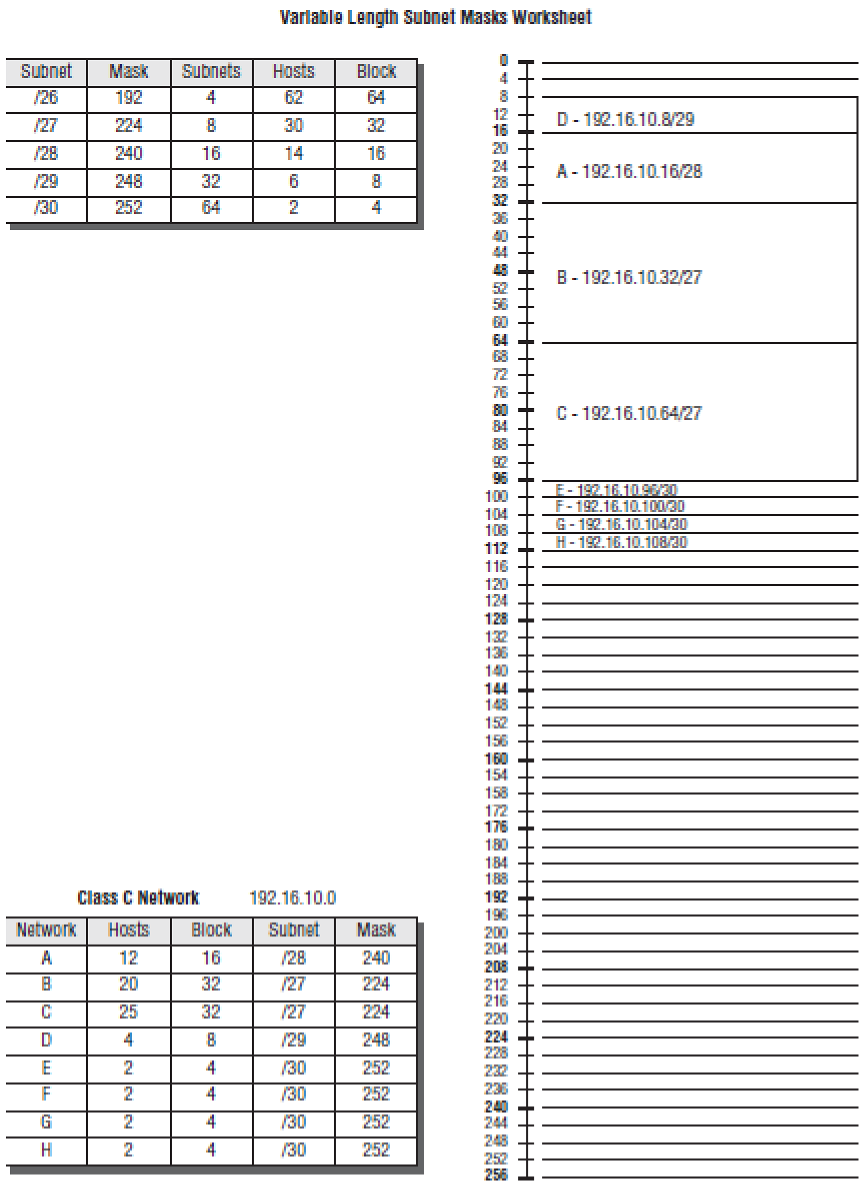


FIGURE 3.5 VLSM network, example two

First, create your VLSM table and use your block size chart to fill in the table with the subnets you need. Figure 3.7 shows a possible solution.

Notice that we filled in this entire chart and only have room for one more block size of 4! Only with a VLSM network can you provide this type of address space savings.



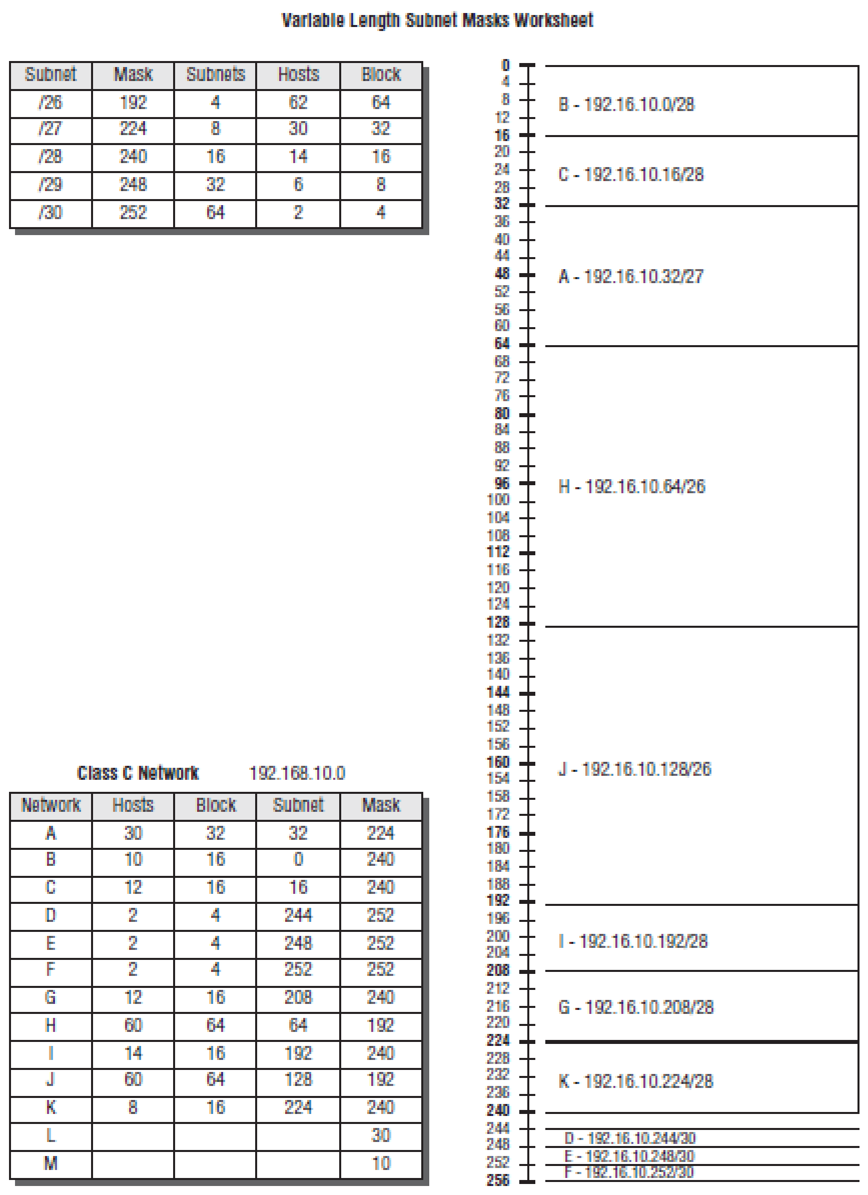
Variable Length Subnet Masks Worksheet

Subnet	Mask	Subnets	Hosts	Block
/26	192	4	62	64
/27	224	8	30	32
/28	240	16	14	16
/29	248	32	6	8
/30	252	64	2	4

Class C Network 192.16.10.0

Network	Hosts	Block	Subnet	Mask
A	12	16	/28	240
B	20	32	/27	224
C	25	32	/27	224
D	4	8	/29	248
E	2	4	/30	252
F	2	4	/30	252
G	2	4	/30	252
H	2	4	/30	252

FIGURE 3. 6 A VLSM table, example one


Variable Length Subnet Masks Worksheet

Subnet	Mask	Subnets	Hosts	Block
/26	192	4	62	64
/27	224	8	30	32
/28	240	16	14	16
/29	248	32	6	8
/30	252	64	2	4

Class C Network 192.168.10.0

Network	Hosts	Block	Subnet	Mask
A	30	32	32	224
B	10	16	0	240
C	12	16	16	240
D	2	4	244	252
E	2	4	248	252
F	2	4	252	252
G	12	16	208	240
H	60	64	64	192
I	14	16	192	240
J	60	64	128	192
K	8	16	224	240
L				30
M				10

Figure 3. 7 VLSM table, example two

Keep in mind that it doesn't matter where you start your block sizes as long as you always count from zero. For example, if you had a block size of 16, you must start at 0 and count from there—0, 16, 32, 48, etc. You can't start a block size of 16 from, say, 40 or anything other than increments of 16.

Here's another example. If you have block sizes of 32, you must start at zero like this: 0, 32, 64, 96, and so on. Just remember that you don't get to start wherever you want; you must always start counting from zero. In the example in Figure 3.7, I started at 64 and 128, with my two block sizes of 64. I didn't have much choice, because my options are 0, 64, 128, and 192. However, I was able to add the block sizes of 32, 16, 8, and 4 wherever I wanted just as long as they were in the correct increments of that block size.

Okay—you have three locations you need to address, and the IP network you have received is 192.168.55.0 to use as the addressing for the entire network. You'll use ip subnet-zero and RIPv2 as the routing protocol. (RIPv2 supports VLSM networks, RIPv1 does not.) Figure 3.8 shows the network diagram and the IP address of the RouterA S0/0 interface.

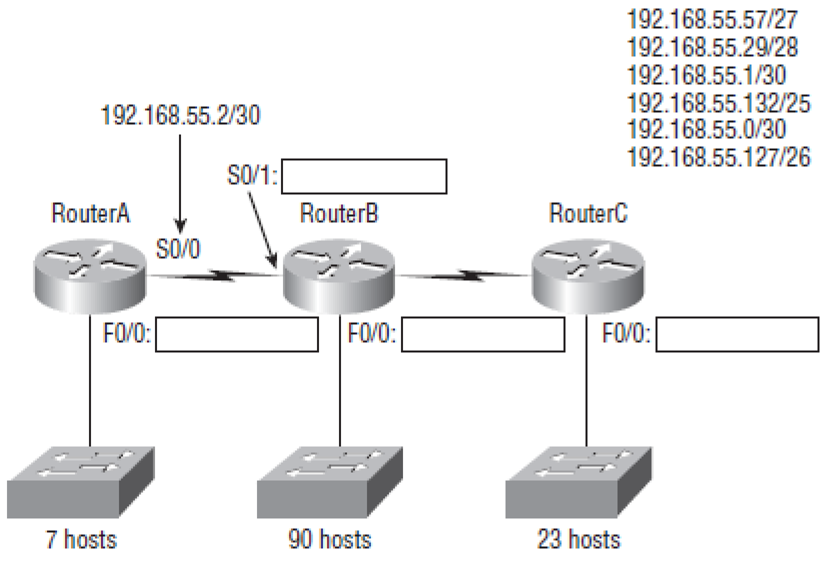


FIGURE 3. 8 VLSM design example

From the list of IP addresses on the right of the figure, which IP address will be placed in each router's FastEthernet 0/0 interface and serial 0/1 of RouterB? To answer this question, first look for clues in Figure 3.9. The first clue is that interface S0/0 on RouterA has IP address 192.168.55.2/30 assigned, which makes for an easy answer. A /30, as you know, is 255.255.255.252, which gives you a block size of 4. Your subnets are 0, 4, 8, and so on. Since the known host has an IP address of 2, the only other valid host in the zero subnet is 1, so the third answer down is what you want for the s0/1 interface of RouterB.

The next clues are the listed number of hosts for each of the LANs. RouterA needs 7 hosts, a block size of 16 (/28); RouterB needs 90 hosts, a block size of 128 (/25); and RouterC needs 23 hosts, a block size of 32 (/27).

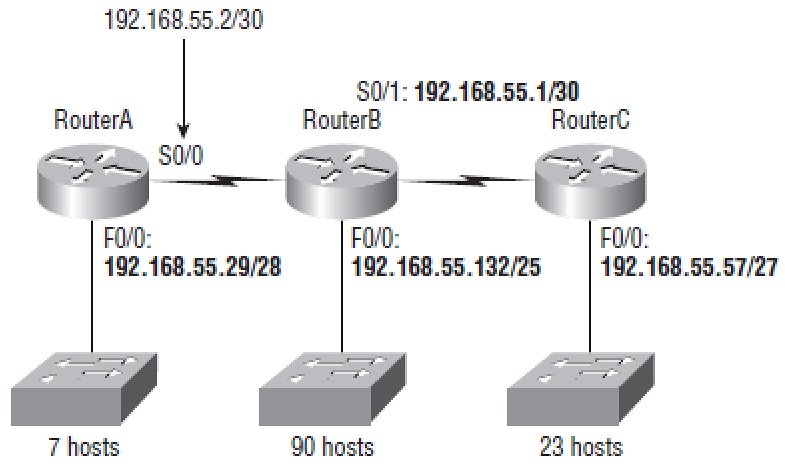


Figure 3.9 shows the answers to this question.

Describe the technological requirements for running IPv6 in conjunction with IPv4 such as dual stack

Describe the technological requirements for running IPv6 in conjunction with IPv4 (including protocols, dual stack, tunneling, etc)

The IPv6 header and address structure has been completely overhauled, and many of the features that were basically just afterthoughts and addendums in IPv4 are now included as full blown standards in IPv6. It’s seriously well equipped, poised, and ready to manage the mind blowing demands of the Internet to come.

Why Do We Need IPv6?

Well, the short answer is, because we need to communicate, and our current system isn’t really cutting it anymore—rather like how the Pony Express can’t compete with airmail. Just look at how much time and effort we’ve invested in coming up with slick new ways to conserve bandwidth and IP addresses. We’ve even come up with VLSMs in our struggle to overcome the worsening address drought.

It’s reality—the number of people and devices that connect to networks increases each and every day. That’s not a bad thing at all—we’re finding new and exciting ways to communicate with more people all the time, and that’s a good thing. In fact, it’s a basic human need. But the forecast isn’t exactly blue skies and sunshine because introduction, IPv4, upon which our ability to communicate is presently dependent, is going to run out of addresses for us to use. IPv4 has only about 4.3 billion addresses available—in theory, and we know that we don’t even get to use all of those. There really are only about 250 million addresses that can be assigned to devices. Sure, the use of Classless Inter-Domain Routing (CIDR) and NAT has helped to extend the inevitable dearth of addresses, but we will run out of them, and it's going to happen within a few years. China is barely online, and we know there's a huge population of people and corporations there that surely want to be. There are a lot of reports that give us all kinds of numbers, but all you really need to think about to convince yourself that I'm not just being an alarmist is the fact that there are about 6.5 billion people in the world today, and it's estimated that just over 10 percent of that population is connected to the Internet—wow!

That statistic is basically screaming at us the ugly truth that based on IPv4's capacity, every person can't even have a computer—let alone all the other devices we use with them. I have more than one computer, and it's pretty likely you do too. And I'm not even including in the mix phones, laptops, game consoles, fax machines, routers, switches, and a mother lode of other devices we use every day! So, I think I've made it pretty clear that we've got to do something before we run out of addresses and lose the ability to connect with each other as we know it. And that "something" just happens to be implementing IPv6.

The Benefits and Uses for IPv6
So, what's so fabulous about IPv6? Is it really the answer to our coming dilemma? Is it really worth it to upgrade from IPv4? All good questions—you may even think of a few more. Of course, there's going to be that group of people with the time-tested and well-known "resistance to change syndrome," but don't listen to them. If we had done that years ago, we'd still be waiting weeks, even months for our mail to arrive via horseback. Instead, just know that the answer is a resounding YES! Not only does IPv6 give us lots of addresses (3.4 x 10^38 = definitely enough), but there are many other features built into this version that make it well worth the cost, time, and effort required to migrate to it.

Today's networks, as well as the Internet, have a ton of unforeseen requirements that simply were not considerations when IPv4 was created. We've tried to compensate with a collection of add-ons that can actually make implementing them more difficult than they would be if they were applied according to a standard. By default, IPv6 has improved upon and included many of those features as standard and mandatory. One of these sweet new standards is IPSec. Another little beauty is known as mobility, and as its name suggests, it allows a device to roam from one network to another without dropping connections.

But it's the efficiency features that are really going to rock the house! For starters, the header in an IPv6 packet have half the fields, and they are aligned to 64 bits, which gives us some seriously souped-up processing speed—compared to IPv4, lookups happen at light speed! Most of the information that used to be bound into the IPv4 header was taken out, and now you can choose to put it, or parts of it, back into the header in the form of optional extension headers that follow the basic header fields.

And, of course, there's that whole new universe of addresses (3.4 x 10^38) we talked about already. But where did we get them? Did that Criss Angel—Mindfreak dude just show up and, Blammo? I mean, that huge proliferation of address had to come from somewhere! Well it just so happens that IPv6 gives us a substantially larger address space, meaning the address is whole lot bigger—four times bigger as a matter of fact! An IPv6 address is actually 128 bits in length. For now, let me just say that all that additional room permits more levels of hierarchy inside the address space and a more flexible address architecture. It also makes routing much more efficient and scalable because the addresses can be aggregated a lot more effectively.

And IPv6 also allows multiple addresses for hosts and networks. This is especially important for enterprises jonesing for availability. Plus, the new version of IP now includes an expanded use of multicast communication (one device sending to many hosts or to a select group), which will also join in to boost efficiency on networks because communications will be more specific.

IPv4 uses broadcasts very prolifically, causing a bunch of problems, the worst of which is of course the dreaded broadcast storm—an uncontrolled deluge of forwarded broadcast traffic that can bring an entire network to its knees and devour every last bit of bandwidth. Another nasty thing about broadcast traffic is that it interrupts each and every device on the network. When a broadcast is sent out, every machine has to stop what it's doing and respond to the traffic, whether the broadcast is meant for it or not.

But smile everyone: There is no such thing as a broadcast in IPv6 because it uses multicast traffic instead. And there are two other types of communication as well: unicast, which is the same as it is in IPv4, and a new type called anycast. Anycast communication allows the same address to be placed on more than one device so that when traffic is sent to one device addressed in this way, it is routed to the nearest host that shares the same address.

Dual Stacking

This is the most common type of migration strategy because, well, it's the easiest on us—it allows our devices to communicate using either IPv4 or IPv6. Dual stacking lets you upgrade your devices and applications on the network one at a time. As more and more hosts and devices on the network are upgraded, more of your communication will happen over IPv6, and after you've arrived—everything's running on IPv6, and you get to remove all the old IPv4 protocol stacks you no longer need.

Plus, configuring dual stacking on a Cisco router is amazingly easy—all you have to do is enable IPv6 forwarding and apply an address to the interfaces already configured with IPv4. It'll look something like this:

```
Corp(config)#ipv6 unicast-routing
Corp(config)#interface fastethernet 0/0
Corp(config-if)#ipv6 address 2001:db8:3c4d:1::/64 eui-64
Corp(config-if)#ip address 192.168.255.1 255.255.255.0
```

But to be honest, it's really a good idea to understand the various tunneling techniques because it'll probably be awhile before we all start running IPv6 as a solo routed protocol.

6to4 Tunneling

6to4 tunneling is really useful for carrying IPv6 data over a network that's still IPv4. It's quite possible that you'll have IPv6 subnets or other portions of your network that are all IPv6, and those networks will have to communicate with each other. Not so complicated, but when you consider that you might find this happening over a WAN or some other network that you don't control, well, that could be a bit ugly. So, what do we do about this if we don't control the whole tamale? Create a tunnel that will carry the IPv6 traffic for us across the IPv4 network, that's what.

The whole idea of tunneling isn't a difficult concept, and creating tunnels really isn't as hard as you might think. All it really comes down to is snatching the IPv6 packet that's happily traveling across the network and sticking an IPv4 header onto the front of it. It's kind of like catch-and-release fishing, except that the fish doesn't get something plastered on its face before being thrown back into the stream. To get a picture of this, take a look at Figure 3.10.

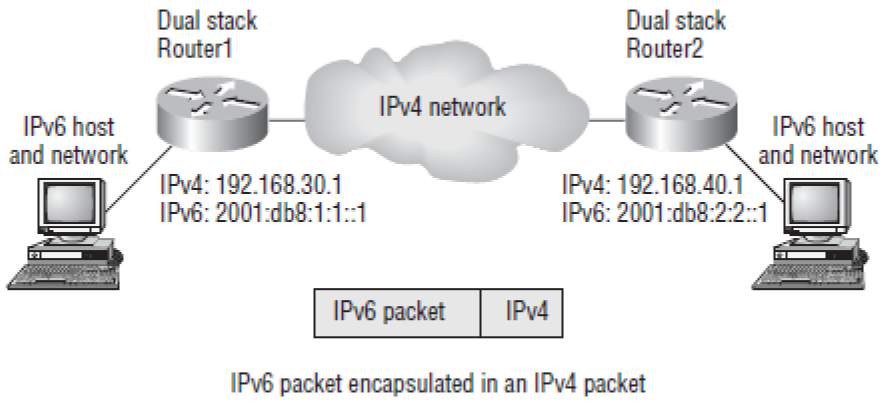


FIGURE 3.10 Creating a 6to4 tunnel

Nice—but to make this happen we're going to need a couple of dual-stacked routers, which I just demonstrated for you, so you should be good to go. Now we have to add a little configuration to place a tunnel between those routers. Tunnels are pretty simple—we just have to tell each router where the tunnel begins and where we want it to end up. Referring again to Figure 3.11, we'll configure the tunnel on each router:

```
Router1(config)#int tunnel 0
Router1(config-if)#ipv6 address 2001:db8:1:1::1/64
```

```
Router1(config-if)#tunnel source 192.168.30.1
Router1(config-if)#tunnel destination 192.168.40.1
Router1(config-if)#tunnel mode ipv6ip
Router2(config)#int tunnel 0
Router2(config-if)#ipv6 address 2001:db8:2:2::1/64
Router2(config-if)#tunnel source 192.168.40.1
Router2(config-if)#tunnel destination 192.168.30.1
Router2(config-if)#tunnel mode ipv6ip
```

With this in place, our IPv6 networks can now communicate over the IPv4 network. Now, I've got to tell you that this is not meant to be a permanent configuration; your end goal should still be to run a total, complete IPv6 network end to end.

One important note here—if the IPv4 network that you're traversing in this situation has a NAT translation point, it would absolutely break the tunnel encapsulation we've just created! Over the years, NAT has been upgraded a lot so that it can handle specific protocols and dynamic connections, and without one of these upgrades, NAT likes to demolish most connections. And since this transition strategy isn't present in most NAT implementations, that means trouble. But there is a way around this little problem and it's called Teredo, which allows all your tunnel traffic to be placed in UDP packets. NAT doesn't blast away at UDP packets, so they won't get broken as other protocols packets do. So, with Teredo in place and your packets disguised under their UDP cloak, the packets will easily slip by NAT alive and well!

Describe IPv6 addresses

Describe IPv6 addresses
Just as understanding how IP addresses are structured and used is critical with IPv4 addressing, it's also vital when it comes to IPv6. You've already read about the fact that at 128 bits, an IPv6 address is much larger than an IPv4 address. Because of this, as well as the new ways the addresses can be used, you've probably guessed that IPv6 will be more complicated to manage. But no worries! As I said, I'll break down the basics and show you what the address looks like, how you can write it, and what many of its common uses are. It's going to be a little weird at first, but before you know it, you'll have it nailed!

So, let's take a look at Figure 3.12.

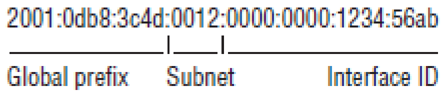


FIGURE 3.12 IPv6 address example

So as you can now see, the address is truly much larger—but what else is different? Well, first, notice that it has eight groups of numbers instead of four and also that those groups are separated by colons instead of periods. And hey wait a second . . . there are letters in that address! Yep, the address is expressed in hexadecimal just like a MAC address is, so you could say this address has eight 16-bit hexadecimal colon-delimited blocks. That's already quite a mouthful, and you probably haven't even tried to say the address out loud yet!

One other thing I want to point out is useful for when you set up your test network to play with IPv6, because I know you're going to want to do that. When you use a web browser to make an HTTP connection to an IPv6 device, you have to type the address into the browser with brackets around the literal address. Why? Well a colon is already being used by the browser for specifying a port number.

Now obviously if you can, you would rather use names to specify a destination (like www.lammle.com), but even though it's definitely going to be a pain in the rear, we just have to accept the fact that sometimes we have to bite the bullet and type in the address number. So, it should be pretty clear that DNS is going to become extremely important when implementing IPv6.

Shortened Expression

The good news is there are a few tricks to help rescue us when writing this monster addresses. For one thing, you can actually leave out parts of the address to abbreviate it, but to get away with doing that you have to follow a couple of rules. First, you can drop any leading zeros in each of the individual blocks. The sample address from earlier would then look like this:

2001:db8:3c4d:12:0:0:1234:56ab

Okay, that's a definite improvement—at least we don't have to write all of those extra zeros! But what about whole blocks that don't have anything in them except zeros? Well, we can lose those, too—at least some of them. Again referring to our sample address, we can remove the two blocks of zeros by replacing them with double colons, like this:

2001:db8:3c4d:12::1234:56ab

Cool—we replaced the blocks of all zeros with double colons. The rule you have to follow to get away with this is that you can only replace one contiguous block of zeros in an address. So, if my address has four blocks of zeros and each of them were separated, I just don't get to replace them all. Check out this example:

2001:0000:0000:0012:0000:0000:1234:56ab

And just know that you can't do this:

2001::12::1234:56ab

Instead, this is the best that you can do:

2001::12:0:0:1234:56ab

The reason why the above example is our best shot is that if we remove two sets of zeros, the device looking at the address will have no way of knowing where the zeros go back in. Basically, the router would look at the incorrect address and say, "Well, do I place two blocks into the first set of double colons and two into the second set, or do I place three blocks into the first set and one block into the second set?" And on and on it would go because the information the router needs just isn't there.

Address Types

We're all familiar with IPv4's unicast, broadcast, and multicast addresses that basically define who or at least how many other devices we're talking to. But as I mentioned, IPv6 adds to that trio and introduces the anycast. Broadcasts, as we know them, have been eliminated in IPv6 because of their cumbersome inefficiency.

So, let's find out what each of these types of IPv6 addressing and communication methods do for us. Unicast Packets addressed to a unicast address are delivered to a single interface. For load balancing, multiple interfaces can use the same address. There are a few different types of unicast addresses, but we don't need to get into that here.

Global unicast addresses These are your typical publicly routable addresses, and they're the same as they are in IPv4.

Link-local addresses These are like the private addresses in IPv4 in that they're not meant to be routed. Think of them as a handy tool that gives you the ability to throw a temporary LAN together for meetings or for creating a small LAN that's not going to be routed but still needs to share and access files and services locally.

Unique local addresses These addresses are also intended for nonrouting purposes, but they are nearly globally unique, so it's unlikely you'll ever have one of them overlap. Unique local addresses were designed to replace site-local addresses, so they basically do almost exactly what IPv4 private addresses do—allow communication throughout a site while being routable to multiple local networks. Site-local addresses were denounced as of September 2004.

Multicast Again, same as in IPv4, packets addressed to a multicast address are delivered to all interfaces identified by the multicast address. Sometimes people call them one-to-many addresses. It's really easy to spot a multicast address in IPv6 because they always start with FF. Anycast Like multicast addresses, an anycast address identifies multiple interfaces, but there's a big difference: the anycast packet is only delivered to one address—actually, to the first one it finds defined in terms of routing distance. And again, this address is special because you can apply a single address to more than one interface. You could call them one-to-one-of many addresses, but just saying "anycast" is a lot easier.

You're probably wondering if there are any special, reserved addresses in IPv6 because you know they're there in IPv4. Well there are—plenty of them! Let's go over them now.

Special Addresses
I'm going to list some of the addresses and address ranges that you should definitely make a point to remember because you'll eventually use them. They're all special or reserved for specific use, but unlike IPv4, IPv6 gives us a galaxy of addresses, so reserving a few here and there doesn't hurt a thing!

0:0:0:0:0:0:0:0 Equals ::. This is the equivalent of IPv4's 0.0.0.0, and is typically the source address of a host when you're using stateful configuration.

0:0:0:0:0:0:0:1 Equals ::1. The equivalent of 127.0.0.1 in IPv4.
0:0:0:0:0:0:192.168.100.1 This is how an IPv4 address would be written in a mixed IPv6/ IPv4 network environment.

2000::/3 The global unicast address range.
FC00::/7 The unique local unicast range.
FE80::/10 The link-local unicast range.
FF00::/8 The multicast range.
3FFF:FFFF::/32 Reserved for examples and documentation.
2001:0DB8::/32 Also reserved for examples and documentation.
2002::/16 Used with 6to4, which is the transition system—the structure that allows IPv6 packets to be transmitted over an IPv4 network without the need to configure explicit tunnels.

IP Routing Technologies

Describe basic routing concepts

Describe basic routing concepts (including packet forwarding, router lookup process)
Once you create an internetwork by connecting your WANs and LANs to a router, you'll need to configure logical network addresses, such as IP addresses, to all hosts on the internetwork so that they can communicate across that internetwork.

The term routing is used for taking a packet from one device and sending it through the network to another device on a different network. Routers don't really care about hosts—they only care about networks and the best path to each network. The logical network address of the destination host is used to get packets to a network through a routed network, and then the hardware address of the host is used to deliver the packet from a router to the correct destination host. If your network has no routers, then it should be apparent that you are not routing. Routers route traffic to all the networks in your internetwork. To be able to route packets, a router must know, at a minimum, the following:

- Destination address
- Neighbor routers from which it can learn about remote networks
- Possible routes to all remote networks
- The best route to each remote network
- How to maintain and verify routing information

The router learns about remote networks from neighbor routers or from an administrator.
The router then builds a routing table (a map of the internetwork) that describes how to find the remote networks. If a network is directly connected, then the router already knows how to get to it.

If a network isn't directly connected to the router, the router must use one of two ways to learn how to get to the remote network:

Static routing, meaning that someone must hand-type all network locations into the routing table, or something called dynamic routing. In dynamic routing, a protocol on one router communicates with the same protocol running on neighbor routers. The routers then update each other about all the networks they know about and place this information into the routing table. If a change occurs in the network, the dynamic routing protocols automatically inform all routers about the event. If static routing is used, the administrator is responsible for updating all changes by hand into all routers. Typically, in a large network, a combination of both dynamic and static routing is used.

Before we jump into the IP routing process, let’s take a look at a simple example that demonstrates how a router uses the routing table to route packets out of an interface.
Figure 4.1 shows a simple two-router network. Lab_A has one serial interface and three LAN interfaces.

Looking at Figure 4.1, can you see which interface Lab_A will use to forward an IP datagram to a host with an IP address of 10.10.10.10?

By using the command show ip route , we can see the routing table (map of the internetwork) that Lab_A uses to make forwarding decisions:

```
Lab_A#
sh ip route
[output cut]
Gateway of last resort is not set
C 10.10.10.0/24 is directly connected, FastEthernet0/0
C 10.10.20.0/24 is directly connected, FastEthernet0/1
C 10.10.30.0/24 is directly connected, FastEthernet0/2
C 10.10.40.0/24 is directly connected, Serial 0/0
```

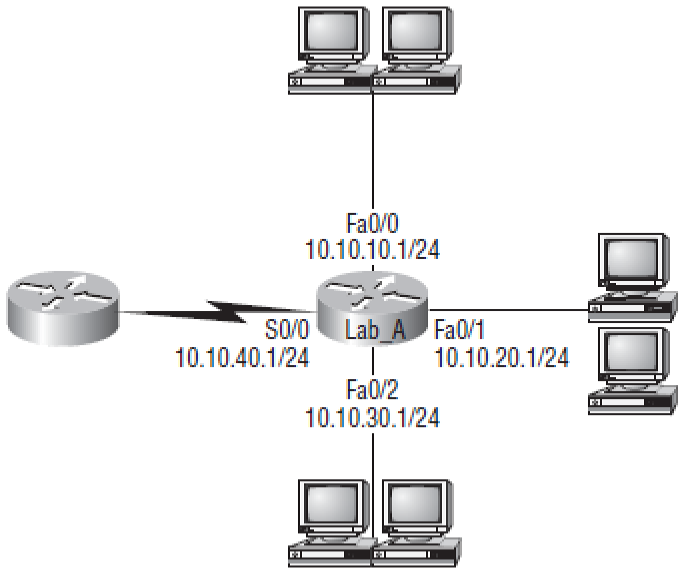


FIGURE 4. 1 A simple routing example

The C in the routing table output means that the networks listed are “directly connected,” and until we add a routing protocol—something like RIP, EIGRP, and so on—to the routers in our internetwork (or use static routes), we’ll have only directly connected networks in our routing table.
So, let’s get back to the original question: By looking at the figure and the output of the routing table, can you tell what IP will do with a received packet that has a destination IP address of

10.10.10.10? The routers will packet-switch the packet to interface FastEthernet 0/0, and this interface will frame the packet and then send it out on the network segment.

Because we can, let's do another example: Based on the output of the next routing table, which interface will a packet with a destination address of 10.10.10.14 be forwarded from?

```
Lab_A#
sh ip route
[output cut]
Gateway of last resort is not set
C 10.10.10.16/28 is directly connected, FastEthernet0/0
C 10.10.10.8/29 is directly connected, FastEthernet0/1
C 10.10.10.4/30 is directly connected, FastEthernet0/2
C 10.10.10.0/30 is directly connected, Serial 0/0
```

First, you can see that the network is subnetted and each interface has a different mask. And I have to tell you—you just can't answer this question if you can't subnet! 10.10.10.14 would be a host in the 10.10.10.8/29 subnet connected to the FastEthernet0/1 interface.

Using DNS to Resolve Names

If you have a lot of devices and don't want to create a host table in each device, you can use a DNS server to resolve hostnames.

Any time a Cisco device receives a command it doesn't understand, it will try to resolve it through DNS by default. Watch what happens when I type the special command todd at a Cisco router prompt:

```
Corp#
todd
Translating "todd"...domain server (255.255.255.255)
Translating "todd"...domain server (255.255.255.255)
Translating "todd"...domain server (255.255.255.255)
% Unknown command or computer name, or unable to find
computer address
Corp#
```

It doesn't know my name or what command I am trying to type, so it tries to resolve this through DNS. This is really annoying because I need to hang out and wait for the name lookup to time out. You can get around this and prevent a time-consuming DNS lookup by using the no ip domain-lookup command on your router from global configuration mode.

If you have a DNS server on your network, you need to add a few commands to make DNS name resolution work:

The first command is ip domain-lookup, which is turned on by default. It needs to be entered only if you previously turned it off (with the no ip domain-lookup
Command). The command can be used without the hyphen as well (ip domain lookup).

The second command is ip name-server. This sets the IP address of the DNS server. You can enter the IP addresses of up to six servers.

The last command is ip domain-name. Although this command is optional, it really should be set. It appends the domain name to the hostname you type in. Since DNS uses a fully qualified domain name (FQDN) system, you must have a full DNS name, in the form domain.com

Here's an example of using these three commands:

```
Corp#
config t
Corp(config)#
ip domain-lookup
Corp(config)#
ip name-server ?
A.B.C.D Domain server IP address (maximum of 6)
Corp(config)#
ip name-server 192.168.0.70
Corp(config)#
ip domain-name lammle.com
Corp(config)#
^Z
Corp#
After the DNS configurations are set, you can test the DNS server by using a
hostname to
ping or telnet a device like this:
Corp#
ping R1
Translating "R1"...domain server (192.168.0.70) [OK]
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.2.2.2, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max
= 28/31/32 ms
Notice that the router uses the DNS server to resolve the name.
```

Configure and verify utilizing the CLI to set basic Router configuration

Describe the operation of Cisco routers (including router bootup process, POST, router components)

To configure and troubleshoot a Cisco internetwork, you need to know the major components of Cisco routers and understand what each one does. Table 4.2 describes the major Cisco router components.

TABLE 4.2 Cisco Router Components

Component	Description
Bootstrap	Stored in the microcode of the ROM, the bootstrap is used to bring a router up during initialization. It will boot the router and then load the IOS.
POST (power-on self-test)	Stored in the microcode of the ROM, the POST is used to check the basic functionality of the router hardware and determines which interfaces are present.
ROM monitor	Stored in the microcode of the ROM, the ROM monitor is used for manufacturing, testing, and troubleshooting.
Mini-IOS	Called the RXBOOT or bootloader by Cisco, the mini-IOS is a small IOS in ROM that can be used to bring up an interface and load a Cisco IOS into flash memory. The mini-IOS can also perform a few other maintenance operations.
RAM (random access memory)	Used to hold packet buffers, ARP cache, routing tables, and also the software and data structures that allow the router to function. `Running-config` is stored in RAM, and most routers expand the IOS from flash into RAM upon boot.
ROM (read-only memory)	Used to start and maintain the router. Holds the POST and the Bootstrap program, as well as the mini-IOS.
Flash memory	Stores the Cisco IOS by default. Flash memory is not erased when the router is reloaded. It is EEPROM (electronically erasable programmable read-only memory) created by Intel.
NVRAM (nonvolatile RAM)	Used to hold the router and switch configuration. NVRAM is not erased when the router or switch is reloaded. Does not store an IOS. The configuration register is stored in NVRAM.
Configuration register	Used to control how the router boots up. This value can be found as the last line of the `show version` command output and by default is set to 0x2102, which tells the router to load the IOS from flash memory as well as to load the configuration from NVRAM.

The Router Boot Sequence

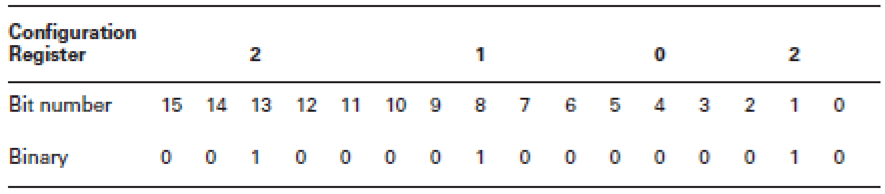
When a router boots up, it performs a series of steps, called the *boot sequence*, to test the hardware and load the necessary software. The boot sequence consists of the following steps:

1. The router performs a POST. The POST tests the hardware to verify that all components of the device are operational and present. For example, the POST checks for the different interfaces on the router. The POST is stored in and run from *ROM (read-only memory)*.
2. The bootstrap then looks for and loads the Cisco IOS software. The *bootstrap* is a program in ROM that is used to execute programs. The bootstrap program is responsible for finding where each IOS program is located and then loading the file. By default, the IOS software is loaded from flash memory in all Cisco routers
3. The IOS software looks for a valid configuration file stored in NVRAM. This file is called startup-config and is only there if an administrator copies the running-config file into NVRAM. (Cisco's new Integrated Services Router (ISR) have a small startupconfig file preloaded.)
4. If a startup-config file is in NVRAM, the router will copy this file and place it in RAM and call the file running-config. The router will use this file to run the router. The router should now be operational. If a startup-config file is not in NVRAM, the router will broadcast out any interface that detects carrier detect (CD) for a TFTP host looking for a configuration, and when that fails (typically it will fail—most people won't even realize the router has attempted this process), it will start the setup mode configuration process.

Managing Configuration Register
All Cisco routers have a 16-bit software register that's written into NVRAM. By default, the configuration register is set to load the Cisco IOS from flash memory and to look for and load the startup-config file from NVRAM.

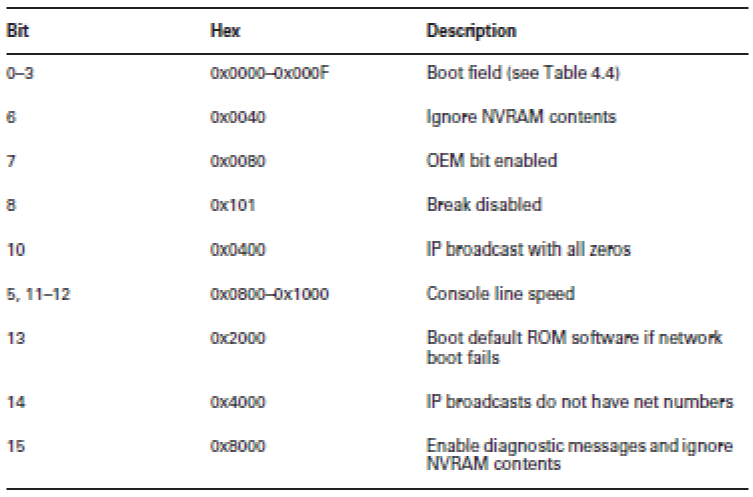
Understanding the Configuration Register Bits
The 16 bits (2 bytes) of the configuration register are read from 15 to 0, from left to right. The default configuration setting on Cisco routers is 0x2102. This means that bits 13, 8, and 1 are on, as shown in Table 4.3. Notice that each set of 4 bits (called a nibble) is read in binary with a value of 8, 4, 2, 1.

TABLE 4 . 3 The Configuration Register Bit Numbers



Configuration Register			2					1				0			2	
Bit number	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Binary	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0

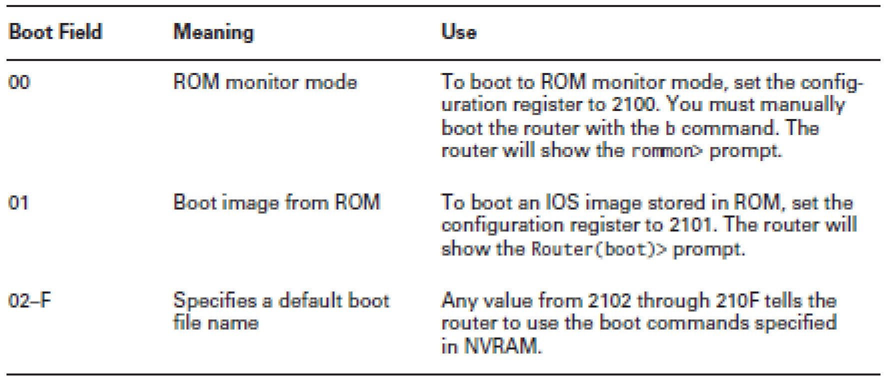
TABLE 4. 4 Software Configuration Meanings



Bit	Hex	Description
0–3	0x0000–0x000F	Boot field (see Table 4.4)
6	0x0040	Ignore NVRAM contents
7	0x0080	OEM bit enabled
8	0x101	Break disabled
10	0x0400	IP broadcast with all zeros
5, 11–12	0x0800–0x1000	Console line speed
13	0x2000	Boot default ROM software if network boot fails
14	0x4000	IP broadcasts do not have net numbers
15	0x8000	Enable diagnostic messages and ignore NVRAM contents

The boot field, which consists of bits 0–3 in the configuration register, controls the router boot sequence. Table 4.5 describes the boot field bits.

TABLE 4. 5 The Boot Field (Configuration Register Bits 00–03)



Boot Field	Meaning	Use
00	ROM monitor mode	To boot to ROM monitor mode, set the configuration register to 2100. You must manually boot the router with the b command. The router will show the rommon> prompt.
01	Boot image from ROM	To boot an IOS image stored in ROM, set the configuration register to 2101. The router will show the Router(boot)> prompt.
02–F	Specifies a default boot file name	Any value from 2102 through 210F tells the router to use the boot commands specified in NVRAM.

Cisco IOS commands to perform basic router setup

Cisco IOS Commands to perform basic router setup

When you boot up your Cisco router for the first time, you notice some basic configuration has already been performed. Use the **show running-config** command to view the initial configuration, as shown in the following example.

```
Router# show running-config
Building configuration...
Current configuration : 723 bytes
!
version 12.4
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname Router
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
!
no aaa new-model
!
no ipv6 cef
ip source-route
ip cef
!
!
!
!
multilink bundle-name authenticated
!
!
archive
 log config
  hidekeys
!
!
!
!
!
interface GigabitEthernet0/0
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface GigabitEthernet0/1
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface GigabitEthernet0/2
 no ip address
 shutdown
 duplex auto
 speed auto
!
ip forward-protocol nd
```

```
!
no ip http server
!
!
!
!
!
control-plane
!
!
line con 0
line aux 0
line vty 0 3
 login
!
exception data-corruption buffer truncate
scheduler allocate 20000 1000
end
```

Configuring Global Parameters

To configure the global parameters for your router, follow these steps.

SUMMARY STEPS

1. **configure terminal**
2. **hostname** *name*
3. **enable secret** *password*
4. **no ip domain-lookup**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	Example:
	`Router> enable`
	`Router# configure terminal`
	`Router(config)#`
		Enters global configuration mode, when using the console port.
		Use the following to connect to the router with a remote terminal:
		`telnet` *`router name or address`*
		`Login:` *`login id`*
		`Password: *********`
		`Router> enable`

Step 2	**hostname** *name* **Example:** `Router(config)# hostname Router`	Specifies the name for the router.

	`Router(config)#`	

Step 3	**enable secret** *password*
	Example:
	`Router(config)# enable secret cr1ny5ho`
	`Router(config)#`
		Specifies an encrypted password to prevent unauthorized access to the router.

Step 4	**no ip domain-lookup** **Example:** `Router(config)# no ip domain-lookup Router(config)#`	Disables the router from translating unfamiliar words (typos) into IP addresses.

For complete information on global parameter commands, see the Cisco IOS Release configuration guide documentation set.

Configuring I/O Memory Allocation

To reallocate the percentage of DRAM in use for I/O memory and processor memory on Cisco 3925E and Cisco 3945E routers, use the**memory-size iomem** *i/o-memory-percentage* command in global configuration mode. To revert to the default memory allocation, use the **no**form of this command. This procedure enables **smartinit**.

Syntax	Description
i/o-memory-percentage	The percentage of DRAM allocated to I/O memory. The values permitted are 5, 10, 15, 20, 25, 30, 40, and 50. A minimum of 201 MB of memory is required for I/O memory.

Configure and verify operation status of an Ethernet interface

Connect, configure, and verify the operational status of a device interface

Interface configuration is one of the most important router configurations, because without interfaces, a router is pretty much a completely useless object. Plus, interface configurations must be totally precise to enable communication with other devices. Network layer addresses, media type, bandwidth, and other administrator commands are all used to configure an interface.

Different routers use different methods to choose the interfaces used on them. Use the question mark to find your interface numbers:

```
Router(config)#int serial ?
<0-9> Serial interface number
```

Now it’s time to choose the interface you want to configure. Once you do that, you will be in interface configuration for that specific interface. The following command would be used to choose serial port 5, for example:

```
Router(config)#int serial 5
Router(config)-if)
```

The above router has one Ethernet 10BaseT port, and typing interface ethernet 0 can configure that interface, as shown here:

```
Router(config)#int ethernet ?
<0-0> Ethernet interface number
Router(config)#int ethernet 0
Router(config-if)#
```

As I showed you above, the 2500 router is a fixed configuration router. This means that when you buy that model, you’re stuck with that physical configuration—a huge reason why I don’t use them much. I certainly never would use them in a production setting anymore.

To configure an interface, we always used the interface type number sequence, but the 2600 and 2800 series routers (actually, any ISR router for that matter), use a physical slot in the router, with a port number on the module plugged into that slot. So on a modular router, the configuration would be interface type slot/port, as shown here:

```
Router(config)#int fastethernet ?
<0-1> FastEthernet interface number
Router(config)#int fastethernet 0
% Incomplete command.
Router(config)#int fastethernet 0?
/
Router(config)#int fastethernet 0/?
<0-1> FastEthernet interface number
```

Make note of the fact that you can’t just type int fastethernet 0. You must type the full command: type slot/port, or int fastethernet 0/0 (or int fa 0/0).

For the ISR series, it’s basically the same, only you get even more options. For example, the built-in Fast Ethernet interfaces work with the same configuration we used with the 2600 series:

```
Todd(config)#int fastEthernet 0/?
```

```
<0-1> FastEthernet interface number
Todd(config)#int fastEthernet 0/0
Todd(config-if)#
```

But the rest of the modules are different—they use three numbers instead of two. The first 0 is the router itself, and then you choose the slot, and then the port. Here's an example of a serial interface on my 2811:

```
Todd(config)#interface serial ?
<0-2> Serial interface number
Todd(config)#interface serial 0/0/?
<0-1> Serial interface number
Todd(config)#interface serial 0/0/0
Todd(config-if)#
```

This can look a little dicey, I know, but I promise it's really not that hard! It helps to remember that you should always view a running-config output first so that you know what interfaces you have to deal with. Here's my 2801 output:

```
Todd(config-if)#do show run
Building configuration...
[output cut]
!
interface FastEthernet0/0
no ip address
shutdown
duplex auto
speed auto
!
interface FastEthernet0/1
no ip address
shutdown
duplex auto
speed auto
!
interface Serial0/0/0
no ip address
shutdown
no fair-queue
!
interface Serial0/0/1
no ip address
shutdown
!
interface Serial0/1/0
no ip address
shutdown
```

```
!
interface Serial0/2/0
no ip address
shutdown
clock rate 2000000
!
[output cut]
```

For the sake of brevity I didn't include my complete running-config, but I've displayed all you need. You can see the two built-in Fast Ethernet interfaces, the two serial interfaces in slot 0 (0/0/0 and 0/0/1), the serial interface in slot 1 (0/1/0), and the serial interface in slot 2 (0/2/0). Once you see the interfaces like this, it makes it a lot easier for you to understand how the modules are inserted into the router.
Just understand that if you type interface e0 on a 2500, interface fastethernet 0/0 on a 2600, or interface serial 0/1/0 on a 2800, all you're doing is choosing an interface to configure, and basically, they're all configured the same way after that.

I'm going to continue with our router interface and I'll include how to bring up the interface and set an IP address on a router interface.

Bringing Up an Interface

You can disable an interface with the interface command shutdown, and enable it with the no shutdown command. If an interface is shut down, it'll display administratively down when using the show interfaces command (sh int for short):

```
Todd#sh int f0/1
FastEthernet0/1 is administratively down, line protocol is down
[output cut]
Another way to check an interface's status is via the show running-config
command. All
interfaces are shut down by default. You can bring up the interface with the
no shutdown
command (no shut for short):
Todd#config t
Todd(config)#int f0/1
Todd(config-if)#no shutdown
Todd(config-if)#
*Feb 28 22:45:08.455: %LINK-3-UPDOWN: Interface FastEthernet0/1,
changed state to up
Todd(config-if)#do show int f0/1
FastEthernet0/1 is up, line protocol is up
[output cut]
```

Configuring an IP Address on an Interface

Even though you don't have to use IP on your routers, it's most often what people actually do use. To configure IP addresses on an interface, use the ip address command from interface configuration mode:

```
Todd(config)#int f0/1
Todd(config-if)#ip address 172.16.10.2 255.255.255.0
```

Don't forget to enable the interface with the no shutdown command. Remember to look at the command show interface int to see if it's administratively shut down or not. Show running-config will also give you this information.

Serial Interface Commands

Wait! Before you just jump in and configure a serial interface, you need some key information—like knowing that the interface will usually be attached to a CSU/DSU type of device that provides clocking for the line to the router, as I've shown in Figure 4.6.

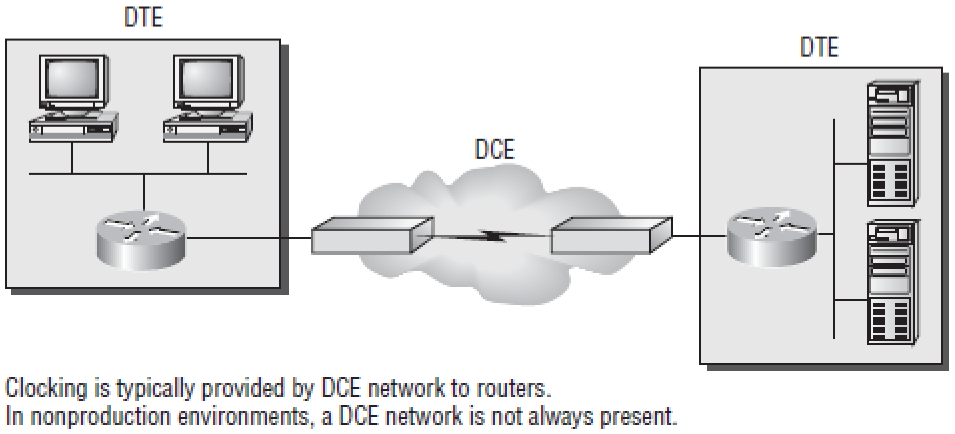

DTE
DCE
DTE
Clocking is typically provided by DCE network to routers.
In nonproduction environments, a DCE network is not always present.

FIGURE 4.6 A typical WAN connection

Here you can see that the serial interface is used to connect to a DCE network via a CSU/DSU that provides the clocking to the router interface. But if you have a back-to-back configuration, (for example, one that's used in a lab environment like I've shown you in Figure 4.7), one end—the data communication equipment (DCE) end of the cable—must provide clocking!

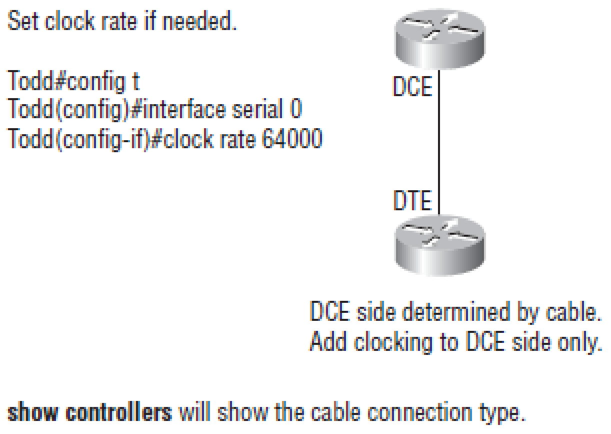

Set clock rate if needed.
Todd#config t
Todd(config)#interface serial 0
Todd(config-if)#clock rate 64000
DCE
DTE
DCE side determined by cable.
Add clocking to DCE side only.
show controllers will show the cable connection type.

FIGURE 4.7 providing clocking on a nonproduction network

By default, Cisco routers are all data terminal equipment (DTE) devices, which means that you must configure an interface to provide clocking if you need it to act like a DCE device. Again, you would not provide clocking on a production T1 connection, for example, because you would have a CSU/DSU connected to your serial interface, as Figure 4.8 shows.

You configure a DCE serial interface with the clock rate command:

```
Todd#config t
Enter configuration commands, one per line. End with CNTL/Z.
Todd(config)#int s0/0/0
Todd(config-if)#clock rate 1000000
```

The clock rate command is set in bits per second. Besides looking at the cable end to check for a label of DCE or DTE, you can see if a router’s serial interface has a DCE cable connected with the show controllers int command:

```
Todd#sh controllers s0/0/0
Interface Serial0/0/0
Hardware is GT96K
DTE V.35idb at 0x4342FCB0, driver data structure at 0x434373D4
```

Here is an example of an output that shows a DCE connection:

```
Todd#sh controllers s0/2/0
Interface Serial0/2/0
Hardware is GT96K
DCE V.35, clock rate 1000000
```

The next command you need to get acquainted with is the bandwidth command. Every Cisco router ships with a default serial link bandwidth of T1 (1.544Mbps). But this has nothing to do with how data is transferred over a link. The bandwidth of a serial link is used by routing protocols such as EIGRP and OSPF to calculate the best cost (path) to a remote network. So if you're using RIP routing, then the bandwidth setting of a serial link is irrelevant since RIP uses only hop count to determine that. Here's an example of using the bandwidth command:

```
Todd#config t
Todd(config)#int s0/0/0
Todd(config-if)#bandwidth ?
<1-10000000> Bandwidth in kilobits
inherit Specify that bandwidth is inherited
receive Specify receive-side bandwidth
Todd(config-if)#bandwidth 1000
```

Did you notice that, unlike the clock rate command, the bandwidth command is configured in kilobits?

Viewing, Saving, and Erasing Configurations

You can manually save the file from DRAM to NVRAM by using the copy running-config startup-config command (you can use the shortcut copy run start also):

```
Todd#copy running-config startup-config
Destination filename [startup-config]? [press enter]
Building configuration...
[OK]
Todd#
Building configuration...
```

When you see a question with an answer in [], it means that if you just press Enter, you're choosing the default answer.
Also, when the command asked for the destination filename, the default answer was startup-config. The reason it asks is because you can copy the configuration pretty much anywhere you want. Take a look:

```
Todd#copy running-config ?
archive: Copy to archive: file system
flash: Copy to flash: file system
ftp: Copy to ftp: file system
http: Copy to http: file system
https: Copy to https: file system
ips-sdf Update (merge with) IPS signature configuration
null: Copy to null: file system
nvram: Copy to nvram: file system
rcp: Copy to rcp: file system
running-config Update (merge with) current system configuration
scp: Copy to scp: file system
startup-config Copy to startup configuration
```

```
syslog: Copy to syslog: file system
system: Copy to system: file system
tftp: Copy to tftp: file system
xmodem: Copy to xmodem: file system
ymodem: Copy to ymodem: file system
```

You can view the files by typing show running-config or show startup-config from privileged mode. The sh run command, which is a shortcut for show running-config, tells us that we are viewing the current configuration:

```
Todd#show running-config
Building configuration...
Current configuration : 3343 bytes
!
version 12.4
[output cut]
```

The sh start command—one of the shortcuts for the show startup-config command— shows us the configuration that will be used the next time the router is reloaded. It also tells us how much NVRAM is being used to store the startup-config file. Here's an example:

```
Todd#show startup-config
Using 1978 out of 245752 bytes
!
version 12.4
[output cut]
```

Verifying Your Configuration

Obviously, show running-config would be the best way to verify your configuration and show startup-config would be the best way to verify the configuration that'll be used the next time the router is reloaded—right?

Well, once you take a look at the running-config, if all appears well, you can verify your configuration with utilities such as ping and Telnet. Ping is Packet Internet Groper, a program that uses ICMP echo requests and replies. Ping sends a packet to a remote host, and if that host responds, you know that the host is alive. But you don't know if it's alive and also well—just because you can ping a Microsoft server does not mean you can log in! Even so, ping is an awesome starting point for troubleshooting an internetwork.

Did you know that you can ping with different protocols? You can, and you can test this by typing ping ? at either the router user-mode or privileged-mode prompt:

```
Router#ping ?
WORD Ping destination address or hostname
```

```
appletalk Appletalk echo
clns CLNS echo
decnet DECnet echo
ip IP echo
ipv6 IPv6 echo
ipx Novell/IPX echo
srb srb echo
tag Tag encapsulated IP echo
<cr>
```

If you want to find a neighbor's Network layer address, either you need to go to the router or switch itself or you can type show cdp entry * protocol to get the Network layer addresses you need for pinging.

Traceroute uses ICMP with IP time to live (TTL) time-outs to track the path a packet takes through an internetwork, in contrast to ping, which just finds the host and responds. And traceroute can also be used with multiple protocols.

```
Router#traceroute ?
WORD Trace route to destination address or hostname
appletalk AppleTalk Trace
clns ISO CLNS Trace
ip IP Trace
ipv6 IPv6 Trace
ipx IPX Trace
<cr>
```

Telnet, FTP, or HTTP are really the best tools because they use IP at the Network layer and TCP at the Transport layer to create a session with a remote host. If you can telnet, ftp, or http into a device, your IP connectivity just has to be good.

```
Router#telnet ?
WORD IP address or hostname of a remote system
<cr>
```

From the router prompt, you just type a hostname or IP address and it will assume you want to telnet—you don't need to type the actual command, telnet.

I am going to show you how to verify the interface statistics.

Verifying with the show interface Command
Another way to verify your configuration is by typing show interface commands, the first of which is show interface ?. That will reveal all the available interfaces to configure.

This command is very useful for verifying and troubleshooting router and network issues.

The following output is from my freshly erased and rebooted 2811 router:

```
Router#sh int ?
Async Async interface
BVI Bridge-Group Virtual Interface
CDMA-Ix CDMA Ix interface
CTunnel CTunnel interface
Dialer Dialer interface
FastEthernet FastEthernet IEEE 802.3
Loopback Loopback interface
MFR Multilink Frame Relay bundle interface
Multilink Multilink-group interface
Null Null interface
Port-channel Ethernet Channel of interfaces
Serial Serial
Tunnel Tunnel interface
Vif PGM Multicast Host interface
Virtual-PPP Virtual PPP interface
Virtual-Template Virtual Template interface
Virtual-TokenRing Virtual TokenRing
accounting Show interface accounting
counters Show interface counters
crb Show interface routing/bridging info
dampening Show interface dampening info
description Show interface description
etherchannel Show interface etherchannel information
irb Show interface routing/bridging info
mac-accounting Show interface MAC accounting info
mpls-exp Show interface MPLS experimental accounting info
precedence Show interface precedence accounting info
pruning Show interface trunk VTP pruning information
rate-limit Show interface rate-limit info
stats Show interface packets & octets, in & out, by switching
path
status Show interface line status
summary Show interface summary
switching Show interface switching
switchport Show interface switchport information
trunk Show interface trunk information
| Output modifiers
<cr>
```

The only “real” physical interfaces are Fast Ethernet, Serial, and Async; the rest are all logical interfaces or commands to verify with.

The next command is show interface fastethernet 0/0. It reveals to us the hardware address, logical address, and encapsulation method, as well as statistics on collisions, as shown here:

```
Router#sh int f0/0
FastEthernet0/0 is up, line protocol is up
Hardware is MV96340 Ethernet, address is 001a.2f55.c9e8 (bia 001a.2f55.c9e8)
Internet address is 192.168.1.33/27
MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
Auto-duplex, Auto Speed, 100BaseTX/FX
ARP type: ARPA, ARP Timeout 04:00:00
Last input never, output 00:02:07, 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
0 packets input, 0 bytes
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
0 watchdog
0 input packets with dribble condition detected
16 packets output, 960 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 babbles, 0 late collision, 0 deferred
0 lost carrier, 0 no carrier
0 output buffer failures, 0 output buffers swapped out
Router#
```

As you probably guessed, we're going to discuss the important statistics from this output, but first, I've got to ask you what subnet is the FastEthernet 0/0 a member of and what's the broadcast address and valid host range?

Just in case you didn't, the address is 192.168.1.33/27. And I've gotta be honest—if you don't know what a /27 is at this point, you'll need a miracle to pass the exam. (A /27 is 255.255.255.224.) The fourth octet is a block size of 32. The subnets are 0, 32, 64, . . .; the Fast Ethernet interface is in the 32 subnet; the broadcast address is 63; and the valid hosts are 33–62.

The preceding interface is working and looks to be in good shape. The show interfaces command will show you if you are receiving errors on the interface, and it will show you the maximum transmission units (MTUs), bandwidth (BW), reliability (255/255 means perfect!), and load (1/255 means no load).

Continuing to use the output from above, what is the bandwidth of the interface? Well, other than the easy giveaway of the interface being called a "Fast Ethernet" interface, we can see the bandwidth is 100000Kbit, which is 100,000,000 (Kbit means to add three zeros), which is 100Mbits per second, or Fast Ethernet. Gigabit would be 1,000,000Kbits per second. The most important statistic of the show interface command is the output of the line and data-link protocol status. If the output reveals that Fast Ethernet 0/0 is up and the line protocol is up, then the interface is up and running:

```
Router#sh int fa0/0
FastEthernet0/0 is up, line protocol is up
```

The first parameter refers to the Physical layer, and it's up when it receives carrier detect. The second parameter refers to the Data Link layer, and it looks for keepalives from the connecting end. (Keepalives are used between devices to make sure that connectivity has not dropped.) Here's an example of where the problem usually is found—on serial interfaces:

```
Router#sh int s0/0/0
Serial0/0 is up, line protocol is down
```

If you see that the line is up but the protocol is down, as shown above, you're experiencing a clocking (keepalive) or framing problem—possibly an encapsulation mismatch. Check the keepalives on both ends to make sure that they match, that the clock rate is set, if needed, and that the encapsulation type is the same on both ends. The output above would be considered a Data Link layer problem.

If you discover that both the line interface and the protocol are down, it's a cable or interface problem. The following output would be considered a Physical layer problem:

```
Router#sh int s0/0/0
Serial0/0 is down, line protocol is down
```

If one end is administratively shut down (as shown next), the remote end would present as down and down:

```
Router#sh int s0/0/0
Serial0/0 is administratively down, line protocol is down
```

To enable the interface, use the command no shutdown from interface configuration mode.
The next show interface serial 0/0/0 command demonstrates the serial line and the maximum transmission unit (MTU)—1,500 bytes by default. It also shows the default bandwidth

(BW) on all Cisco serial links: 1.544Kbps. This is used to determine the bandwidth of the line for routing protocols such as EIGRP and OSPF. Another important configuration to notice is the keep alive, which is 10 seconds by default. Each router sends a keepalive message to its neighbor every 10 seconds, and if both routers aren't configured for the same keepalive time, it won't work.

```
Router#sh int s0/0/0
Serial0/0 is up, line protocol is up
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set, keepalive set
(10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored,
0 abort
0 packets output, 0 bytes, 0 underruns
0 output errors, 0 collisions, 16 interface resets
0 output buffer failures, 0 output buffers swapped out
0 carrier transitions
DCD=down DSR=down DTR=down RTS=down CTS=down
You can clear the counters on the interface by typing the command
clear counters:
Router#clear counters ?
Async Async interface
BVI Bridge-Group Virtual Interface
CTunnel CTunnel interface
Dialer Dialer interface
FastEthernet FastEthernet IEEE 802.3
Group-Async Async Group interface
Line Terminal line
Loopback Loopback interface
MFR Multilink Frame Relay bundle interface
Multilink Multilink-group interface
Null Null interface
Serial Serial
Tunnel Tunnel interface
```

```
Vif PGM Multicast Host interface
Virtual-Template Virtual Template interface
Virtual-TokenRing Virtual TokenRing
<cr>
Router#clear counters s0/0/0
Clear "show interface" counters on this interface
[confirm][Enter]
Router#
00:17:35: %CLEAR-5-COUNTERS: Clear counter on interface
Serial0/0/0 by console
Router#
Verifying with the show ip interface Command
The show ip interface command will provide you with information
regarding the layer 3
configurations of a router’s interfaces:
Router#sh ip interface
FastEthernet0/0 is up, line protocol is up
Internet address is 1.1.1.1/24
Broadcast address is 255.255.255.255
Address determined by setup command
MTU is 1500 bytes
Helper address is not set
Directed broadcast forwarding is disabled
Outgoing access list is not set
Inbound access list is not set
Proxy ARP is enabled
Security level is default
Split horizon is enabled
[output cut]
```

The status of the interface, the IP address and mask, information on whether an access list is set on the interface, and basic IP information are included in this output.

Using the show ip interface brief Command
The show ip interface brief command is probably one of the most helpful commands that you can ever use on a Cisco router. This command provides a quick overview of the router’s interfaces, including the logical address and status:

```
Router#sh ip int brief
Interface IP-Address OK? Method Status Protocol
FastEthernet0/0 unassigned YES unset up up
FastEthernet0/1 unassigned YES unset up up
Serial0/0/0 unassigned YES unset up down
Serial0/0/1 unassigned YES unset administratively down down
Serial0/1/0 unassigned YES unset administratively down down
```

```
Serial0/2/0 unassigned YES unset administratively down down
```

Remember, the administratively down means that you need to type no shutdown under the interface. Notice that Serial0/0/0 is up/down, which means that the physical layer is good and carrier detect is sensed, but no keepalives are being received from the remote end. In a nonproduction network, like the one I am working with, the clock rate isn't set.

Verify router configuration and network connectivity

Verify device configuration and network connectivity using ping, traceroute, Telnet, SSH, or other utilities

Before we move on to determining IP address problems and how to fix them, I just want to mention some basic DOS commands that you can use to help troubleshoot your network from both a PC and a Cisco router (the commands might do the same thing, but they are implemented differently). Packet InterNet Groper (ping) Uses ICMP echo request and replies to test if a node IP stack is initialized and alive on the network. traceroute Displays the list of routers on a path to a network destination by using TTL time-outs and ICMP error messages. This command will not work from a DOS prompt. tracert Same command as traceroute, but it's a Microsoft Windows command and will not work on a Cisco router.telnet Connects a device as a dumb terminal to another device and allows this dumb terminal to run programs on the connected device. All information, including login information, is sent in clear text. secure shell (SSH) Same as telnet, however, with one large difference: the connection is a secure connection and certificates are used to authenticate the connection. SSH also secures or encrypts the connection which is what the keys are for. arp -a Displays IP-to-MAC-address mappings on a Windows PC. show ip arp Same command as arp -a, but displays the ARP table on a Cisco router. Like the commands traceroute and tracert, they are not interchangeable through DOS and Cisco. ipconfig /all Used only from a DOS prompt, shows you the PC network configuration. Once you've gone through all these steps and used the appropriate DOS commands, if necessary, what do you do if you find a problem? How do you go about fixing an IP address configuration error?

Checking Network Connectivity

You can use the ping and traceroute commands to test connectivity to remote devices, and both of them can be used with many protocols, not just IP.

Using the ping Command

So far, you've seen many examples of pinging devices to test IP connectivity and name resolution using the DNS server. To see all the different protocols that you can use with ping, use the ping ? command like this:

```
Todd2509#ping ?
```

```
WORD Ping destination address or hostname
apollo Apollo echo
appletalk Appletalk echo
clns CLNS echo
decnet DECnet echo
ip IP echo
ipx Novell/IPX echo
srb srb echo
tag Tag encapsulated IP echo
vines Vines echo
xns XNS echo
<cr>
```

The ping output displays the minimum, average, and maximum times it takes for a ping packet to find a specified system and return. Here's another example:

```
Todd2509#ping todd2509
Translating "todd2509"...domain server (192.168.0.70)[OK]
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.0.121, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max
= 32/32/32 ms
Todd2509#
```

You can see that the DNS server was used to resolve the name, and the device was pinged in 32ms (milliseconds).

Using the traceroute Command

Traceroute (the traceroute command, or trace for short) shows the path a packet takes to get to a remote device. To see the protocols that you can use with traceroute, use the traceroute ? command, do this:

```
Todd2509#traceroute ?
WORD Trace route to destination address or
hostname
appletalk AppleTalk Trace
clns ISO CLNS Trace
ip IP Trace
ipx IPX Trace
oldvines Vines Trace (Cisco)
vines Vines Trace (Banyan)
<cr>
```

The trace command shows the hop or hops that a packet traverses on its way to a remote device. Here's an example:

```
Todd2509#trace 2501b
Type escape sequence to abort.
Tracing the route to 2501b.lammle.com (172.16.10.2)
1 2501b.lammle.com (172.16.10.2) 16 msec * 16 msec
Todd2509#
```

Using Telnet and SSH

Telnet and SSH are not necessarily used to test network connectivity like ping and traceroute; however, if you can connect to a remote device using Telnet or SSH, this means that you do have good connectivity to the device. This can be considered a better network test than using ping and Telnet because Telnet and SSH are Application layer protocols, whereas ping and traceroute are Network layer protocols.

Telnet, part of the TCP/IP protocol suite, is a virtual terminal protocol that allows you to make connections to remote devices, gather information, and run programs.

After your routers and switches are configured, you can use the Telnet program to reconfigure and/or check up on your routers and switches without using a console cable. You run the Telnet program by typing telnet from any command prompt (DOS or Cisco). You need to have VTY passwords set on the routers for this to work.

Remember, you can't use CDP to gather information about routers and switches that aren't directly connected to your device. But you can use the Telnet application to connect to your neighbor devices and then run CDP on those remote devices to get information on them. You can issue the telnet command from any router prompt like this:

```
Corp#telnet 10.2.2.2
Trying 10.2.2.2 ... Open
Password required, but none set
[Connection to 10.2.2.2 closed by foreign host]
Corp#
```

As you can see, I didn't set my passwords—how embarrassing! Remember that the VTY ports on a router are configured as login, meaning that we have to either set the VTY passwords or use the no login command.

On a Cisco router, you don't need to use the telnet command; you can just type in an IP address from a command prompt, and the router will assume that you want to telnet to the device. Here's how that looks by using just the IP address:

```
Corp#10.2.2.2
Trying 10.2.2.2 ... Open
User Access Verification
Password:
R1>
```

Remember that the VTY password is the user-mode password, not the enable-mode password. Watch what happens when I try to go into privileged mode after telnetting into router R1:

```
R1>en
% No password set
R1>
```

It is basically saying, “No way!” This is a really good security feature because you don’t want anyone telnetting into your device and being able to just type the enable command to get into privileged mode. You’ve got to set your enable-mode password or enable secret password to use Telnet to configure remote devices!

It’s different if you want to set the router to use HTTPS and SSH—you need to add a few more commands.

First, enable the HTTP/HTTPS server (your router won’t support HTTPS if it doesn’t have the advanced services IOS):

```
Router(config)#ip http server
Router(config)#ip http secure-server
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
Router(config)#ip http authentication local
Second, create a user account using privilege level 15 (the highest level):
Router(config)#username cisco privilege ?
<0-15> User privilege level
Router(config)#username cisco privilege 15 password ?
0 Specifies an UNENCRYPTED password will follow
7 Specifies a HIDDEN password will follow
LINE The UNENCRYPTED (cleartext) user password
Router(config)#username cisco privilege 15 password 0 cisco
Last, configure console, SSH, and Telnet to provide local login
authentication at privilege
level access:
Router(config)#line console 0
Router(config-line)#login local
Router(config-line)#exit
Router(config)#line vty 0 ?
<1-1180> Last Line number
<cr>
```

```
Router(config)#line vty 0 1180
Router(config-line)#privilege level 15
Router(config-line)#login local
Router(config-line)#transport input telnet
Router(config-line)#transport input telnet ssh
Router(config-line)#^Z
```

Configure and verify routing configuration for a static or default route given specific routing requirements

Perform and verify routing configuration tasks for a static or default route given specific routing requirements

You must have a good foundation of routing to pass the CCNA exam.
Static routing occurs when you manually add routes in each router's routing table. There are pros and cons to static routing, but that's true for all routing processes. Static routing has the following benefits:

- There is no overhead on the router CPU, which means that you could possibly buy a cheaper router than you would use if you were using dynamic routing.
- There is no bandwidth usage between routers, which means that you could possibly save money on WAN links.
- It adds security, because the administrator can choose to allow routing access to certain networks only

Static routing has the following disadvantages:
The administrator must really understand the internetwork and how each router is connected in order to configure routes correctly.

If a network is added to the internetwork, the administrator has to add a route to it on all routers—by hand.

It's not feasible in large networks because maintaining it would be a full-time job in itself.
Okay—that said, here's the command syntax you use to add a static route to a routing table:

```
ip route [destination_network] [mask] [next-hop_address or
exitinterface] [administrative_distance] [permanent]
```

This list describes each command in the string:

ip route The command used to create the static route. destination network The network you’re placing in the routing table. Mask The subnet mask being used on the network. next-hop_address The address of the next-hop router that will receive the packet and forward it to the remote network. This is a router interface that’s on a directly connected network.

You must be able to ping the router interface before you add the route. If you type in the wrong next-hop address or the interface to that router is down, the static route will show up in the router’s configuration but not in the routing table. Exit interface Used in place of the next-hop address if you want, and shows up as a directly connected route. Administrative distance By default, static routes have an administrative distance of 1 (or even 0 if you use an exit interface instead of a next-hop address). You can change the default value by adding an administrative weight at the end of the command. Permanent If the interface is shut down or the router can’t communicate to the next-hop router, the route will automatically be discarded from the routing table. Choosing the permanent option keeps the entry in the routing table no matter what happens.

Before we dive into configuring static routes, let’s take a look at a sample static route and see what we can find out about it.

```
Router(config)#ip route 172.16.3.0 255.255.255.0 192.168.2.4
 The ip route command tells us simply that it is a static route.
 172.16.3.0 is the remote network we want to send packets to.
 255.255.255.0 is the mask of the remote network.
192.168.2.4 is the next hop, or router, we will send packets to.
```

However, suppose the static route looked like this:

```
Router(config)#ip route 172.16.3.0 255.255.255.0 192.168.2.4 150
```

The 150 at the end changes the default administrative distance (AD) of 1 to 150. No worries—I’ll talk much more about AD when we get into dynamic routing. For now, just remember that the AD is the trustworthiness of a route, where 0 is best and 255 is worst.

One more example, then we’ll start configuring:

```
Router(config)#ip route 172.16.3.0 255.255.255.0 s0/0/0
```

Instead of using a next-hop address, we can use an exit interface that will make the route show up as a directly connected network. Functionally, the next hop and exit interface work exactly the same.

We use default routing to send packets with a remote destination network not in the routing table to the next-hop router. You should only use default routing on stub networks—those with only one exit path out of the network. You can easily create loops with default routing, so be careful!

To configure a default route, you use wildcards in the network address and mask locations of a static route. In fact, you can just think of a default route as a static route that uses wildcards instead of network and mask information.

By using a default route, you can just create one static route entry instead. This sure is easier than typing in all those routes!

```
Router(config)#ip route 0.0.0.0 0.0.0.0 10.1.11.1
Router(config)#ip classless
Router(config)#do show ip route
Gateway of last resort is 10.1.11.1 to network 0.0.0.0
10.0.0.0/24 is subnetted, 2 subnets
C 10.1.11.0 is directly connected, Vlan1
C 10.1.12.0 is directly connected, Dot11Radio0
S* 0.0.0.0/0 [1/0] via 10.1.11.1
Router(config)#
```

If you look at the routing table, you'll see only the two directly connected networks plus an S*, which indicates that this entry is a candidate for a default route. I could have completed the default route command another way:

```
Router(config)#ip route 0.0.0.0 0.0.0.0 vlan1
```

What this is telling us is that if you don't have an entry for a network in the routing table, just forward it out Vlan1 (which will send it out FastEthernet0/0). You can choose the IP address of the next-hop router or the exit interface—either way, it will work the same.

Remember, I used this exit interface configuration with the R3 static route configs. Notice also in the routing table that the gateway of last resort is now set. Even so, there's one more command you must be aware of when using default routes: the ip classless command.

All Cisco routers are classful routers, meaning that they expect a default subnet mask on each interface of the router. When a router receives a packet for a destination subnet that's not in the routing table, it will drop the packet by default. If you're using default routing, you must use the ip classless command because it is possible that no remote subnets will be in the routing table. Since I have version 12.4 of the IOS on my routers, the ip classless command is on by default. If you're using default routing and this command isn't in your configuration, you will need to add it if you have subnetted networks on your routers. The command is shown here:

Router(config)#ip classless

Notice that it's a global configuration mode command. The interesting part of the ip classless command is that default routing sometimes works without it but sometimes doesn't. To be on the safe side, you should always turn on the ip classless command when you use default routing. There's another command you can use to configure a gateway of last resort—the ip default-network command. Figure 4.7 shows a network that needs to have a gateway of last resort statement configured.

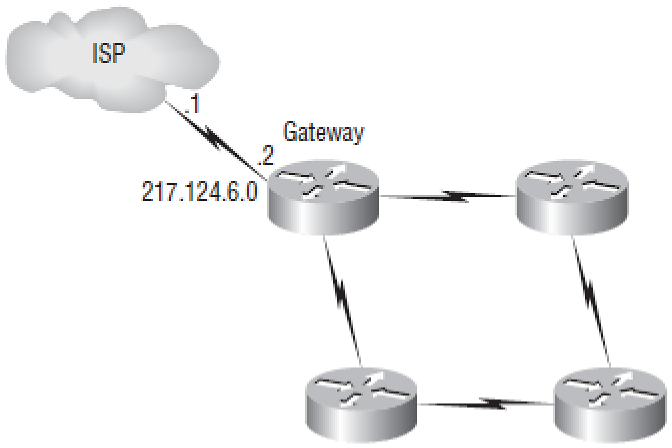


FIGURE 4.7 Configuring a gateway of last resort

Here are three commands (all providing the same solution) for adding a gateway of last resort on the gateway router to the ISP.

```
Gateway(config)#ip route 0.0.0.0 0.0.0.0 217.124.6.1
Gateway(config)#ip route 0.0.0.0 0.0.0.0 s0/0
Gateway(config)#ip default-network 217.124.6.0
```

As I said before, all three of these commands would accomplish the goal of setting the gateway of last resort, but there are some small differences between them. First, the exit interface solution would be used over the other two solutions because it has an AD of 0. Also, the ip default-network command would advertise the default network when you configure an IGP (like RIP) on the router. This is so other routers in your internetwork will receive this route as a default route automatically.

Differentiate methods of routing and routing protocols

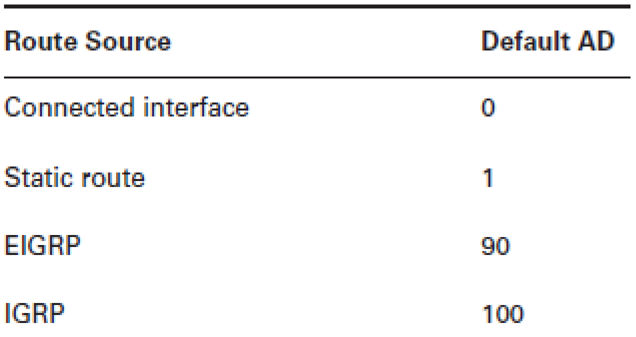
Compare and contrast methods of routing and routing protocols
A routing protocol is used by routers to dynamically find all the networks in the internetwork and to ensure that all routers have the same routing table. Basically, a routing protocol determines the path of a packet through an internetwork. Examples of routing protocols are RIP, RIPv2, EIGRP, and OSPF.

Once all routers know about all networks, a routed protocol can be used to send user data (packets) through the established enterprise. Routed protocols are assigned to an interface and determine the method of packet delivery. Examples of routed protocols are IP and IPv6.
The administrative distance (AD) is used to rate the trustworthiness of routing information received on a router from a neighbor router. An administrative distance is an integer from 0 to 255, where 0 is the most trusted and 255 means no traffic will be passed via this route.

If a router receives two updates listing the same remote network, the first thing the router checks is the AD. If one of the advertised routes has a lower AD than the other, then the route with the lowest AD will be placed in the routing table.

If both advertised routes to the same network have the same AD, then routing protocol metrics (such as hop count or bandwidth of the lines) will be used to find the best path to the remote network. The advertised route with the lowest metric will be placed in the routing table. But if both advertised routes have the same AD as well as the same metrics, then the routing protocol will load-balance to the remote network (which means that it sends packets down each link).
Table 4.5 shows the default administrative distances that a Cisco router uses to decide which route to take to a remote network.

TABLE 4.5 Default Administrative Distances



Route Source	Default AD
Connected interface	0
Static route	1
EIGRP	90
IGRP	100

OSPF	110
RIP	120
External EIGRP	170
Unknown	255 (this route will never be used)

If a network is directly connected, the router will always use the interface connected to the network. If you configure a static route, the router will then believe that route over any other learned routes. You can change the administrative distance of static routes, but, by default, they have an AD of 1. In our static route configuration, the AD of each route is set at 150 or 151. This lets us configure routing protocols without having to remove the static routes. They'll be used as backup routes in case the routing protocol experiences a failure of some type.

For example, if you have a static route, a RIP-advertised route, and an IGRP-advertised route listing the same network, then, by default, the router will always use the static route unless you change the AD of the static route—which we did.

Routing Protocols

There are three classes of routing protocols:

Distance vector The distance-vector protocols find the best path to a remote network by judging distance. Each time a packet goes through a router, that's called a hop. The route with the least number of hops to the network is determined to be the best route. The vector indicates the direction to the remote network. Both RIP and IGRP are distance-vector routing protocols. They send the entire routing table to directly connected neighbors.

Link state In link-state protocols, also called shortest-path-first protocols, the routers each create three separate tables. One of these tables keeps track of directly attached neighbors, one determines the topology of the entire internetwork, and one is used as the routing table. Linkstate routers know more about the internetwork than any distance-vector routing protocol. OSPF is an IP routing protocol that is completely link state. Link-state protocols send updates containing the state of their own links to all other routers on the network. Hybrid protocols use aspects of both distance vector and link state—for example, EIGRP.

There's no set way of configuring routing protocols for use with every business. This is something you really have to do on a case-by-case basis. If you understand how the different routing protocols work, you can make good, solid decisions that truly meet the individual needs of any business.

Configure and verify OSPF (single area)

Configure, verify, and troubleshoot OSPF
Open Shortest Path First (OSPF) is an open standard routing protocol that's been implemented by a wide variety of network vendors, including Cisco. If you have multiple routers and not all of them are Cisco (what!), then you can't use EIGRP, can you? So, your remaining CCNA objective options are basically RIP, RIPv2, and OSPF.

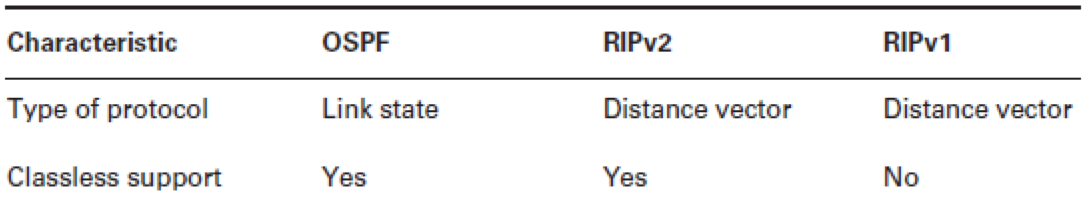
OSPF works by using the Dijkstra algorithm. First, a shortest path tree is constructed, and then the routing table is populated with the resulting best paths. OSPF converges quickly, although perhaps not as quickly as EIGRP, and it supports multiple, equal-cost routes to the same destination. Like EIGRP, it does support both IP and IPv6 routed protocols.

OSPF provides the following features:

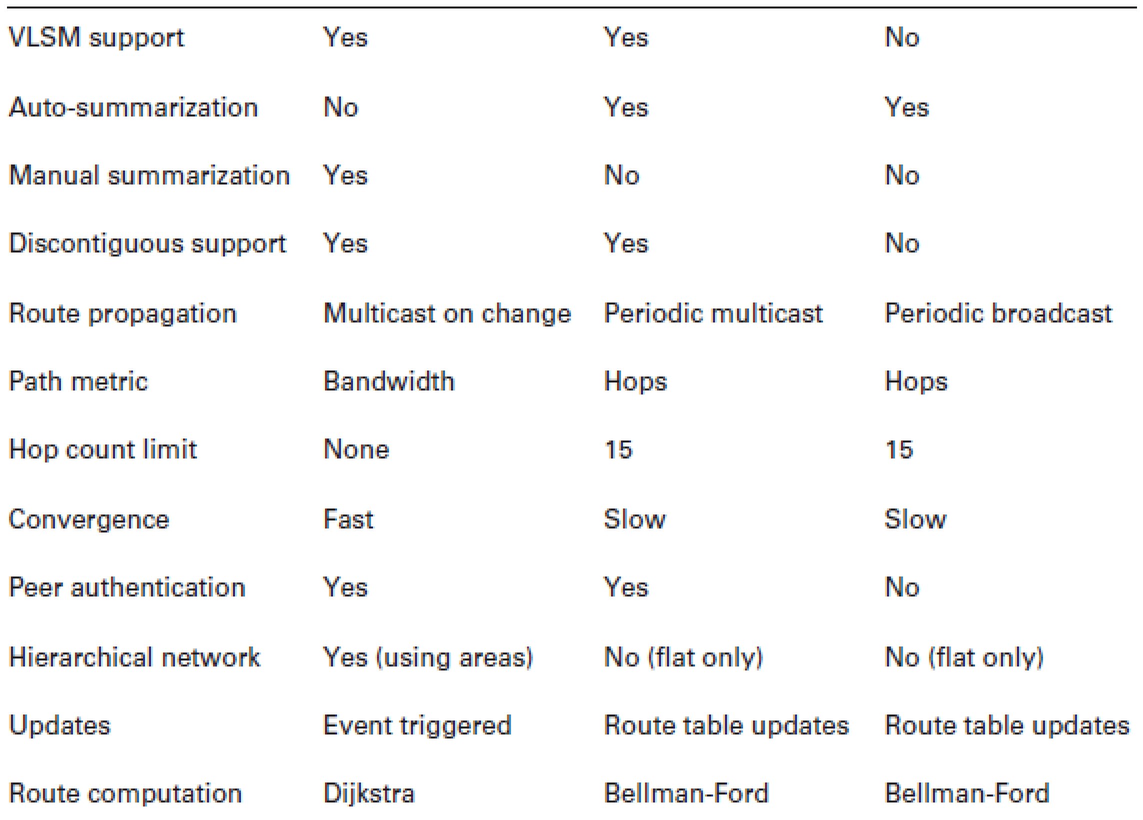
- Consists of areas and autonomous systems
- Minimizes routing update traffic
- Allows scalability
- Supports VLSM/CIDR
- Has unlimited hop count
- Allows multi-vendor deployment (open standard)

OSPF is the first link-state routing protocol that most people are introduced to, so it's useful to see how it compares to more traditional distance-vector protocols such as RIPv2 and RIPv1. Table 4.6 gives you a comparison of these three protocols.

TABLE 4 . 6 OSPF and RIP comparison



Characteristic	OSPF	RIPv2	RIPv1
Type of protocol	Link state	Distance vector	Distance vector
Classless support	Yes	Yes	No



VLSM support	Yes	Yes	No
Auto-summarization	No	Yes	Yes
Manual summarization	Yes	No	No
Discontiguous support	Yes	Yes	No
Route propagation	Multicast on change	Periodic multicast	Periodic broadcast
Path metric	Bandwidth	Hops	Hops
Hop count limit	None	15	15
Convergence	Fast	Slow	Slow
Peer authentication	Yes	Yes	No
Hierarchical network	Yes (using areas)	No (flat only)	No (flat only)
Updates	Event triggered	Route table updates	Route table updates
Route computation	Dijkstra	Bellman-Ford	Bellman-Ford

OSPF has many features beyond the few I've listed in Table 4.6, and all of them contribute to a fast, scalable, and robust protocol that can be actively deployed in thousands of production networks.

OSPF is supposed to be designed in a hierarchical fashion, which basically means that you can separate the larger internetwork into smaller internetworks called areas. This is the best design for OSPF.

The following are reasons for creating OSPF in a hierarchical design:

- To decrease routing overhead
- To speed up convergence
- To confine network instability to single areas of the network

This does not make configuring OSPF easier, but more elaborate and difficult. Figure 4.8 shows a typical OSPF simple design. Notice how each router connects to the backbone—called area 0, or the backbone area. OSPF must have an area 0, and all other areas should connect to this area. Routers that connect other areas to the backbone area within an AS are called Area Border Routers (ABRs). Still, at least one interface of the ABR must be in area 0.

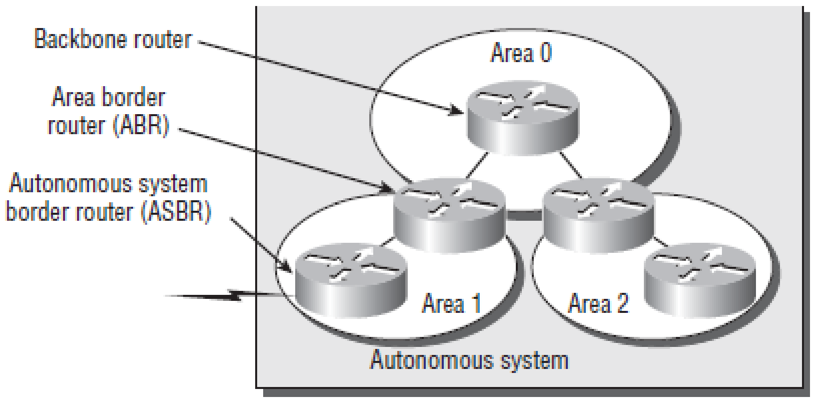


FIGURE 4.8 OSPF design example

OSPF runs inside an autonomous system, but it can also connect multiple autonomous systems. The router that connects this ASs is called an Autonomous System Boundary Router (ASBR). Ideally, you would create other areas of networks to help keep route updates to a minimum and to keep problems from propagating throughout the network.

Configuring OSPF Areas

After identifying the OSPF process, you need to identify the interfaces that you want to activate OSPF communications on as well as the area in which each resides. This will also configure the networks you're going to advertise to others. OSPF uses wildcards in the configuration—which are also used in access-list configurations.

Here's an OSPF basic configuration example for you:

```
Lab_A#config t
Lab_A(config)#router ospf 1
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area ?
<0-4294967295> OSPF area ID as a decimal value
A.B.C.D OSPF area ID in IP address format
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area 0
```

Remember, the OSPF Process ID number is irrelevant. It can be the same on every router on the network, or it can be different—doesn't matter. It's locally significant and just enables the OSPF routing on the router.

We're going to start by taking a quick look at the routing table of the Corp router: So, let's issue a show ip route command on the Corp router:

```
10.0.0.0/24 is subnetted, 12 subnets
O 10.1.11.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
```

```
O 10.1.10.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
O 10.1.9.0 [110/74] via 10.1.4.2, 00:01:31, Serial0/1/0
O 10.1.8.0 [110/65] via 10.1.4.2, 00:01:31, Serial0/1/0
O 10.1.12.0 [110/66] via 10.1.5.2, 00:01:31, Serial0/2/0
C 10.1.3.0 is directly connected, Serial0/0/1
C 10.1.2.0 is directly connected, Serial0/0/0
C 10.1.1.0 is directly connected, FastEthernet0/1
O 10.1.7.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
O 10.1.6.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
C 10.1.5.0 is directly connected, Serial0/2/0
C 10.1.4.0 is directly connected, Serial0/1/0
```

The Corp router shows the found routes for all 12 of our networks, with the O representing OSPF internal routes (the Cs are obviously our directly connected networks). It also found the dual routes to networks 10.1.6.0 and 10.1.7.0. I removed the bandwidth and delay commands from under the interface, so the defaults are being used to determine the metric. But remember, OSPF only uses bandwidth to determine the best path to a network.

The show ip ospf Command

The show ip ospf command is used to display OSPF information for one or all OSPF processes running on the router. Information contained therein includes the Router ID, area information, SPF statistics, and LSA timer information. Let’s check out the output from the Corp router:

```
Corp#sh ip ospf
Routing Process "ospf 132" with ID 10.1.5.1
Start time: 04:32:04.116, Time elapsed: 01:27:10.156
Supports only single TOS(TOS0) routes
Supports opaque LSA
Supports Link-local Signaling (LLS)
Supports area transit capability
Router is not originating router-LSAs with maximum metric
Initial SPF schedule delay 5000 msecs
Minimum hold time between two consecutive SPFs 10000 msecs
Maximum wait time between two consecutive SPFs 10000 msecs
Incremental-SPF disabled
Minimum LSA interval 5 secs
Minimum LSA arrival 1000 msecs
LSA group pacing timer 240 secs
Interface flood pacing timer 33 msecs
Retransmission pacing timer 66 msecs
Number of external LSA 0. Checksum Sum 0x000000
Number of opaque AS LSA 0. Checksum Sum 0x000000
Number of DCbitless external and opaque AS LSA 0
Number of DoNotAge external and opaque AS LSA 0
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
Number of areas transit capable is 0
External flood list length 0
Area BACKBONE(0)
Number of interfaces in this area is 5
Area has no authentication
SPF algorithm last executed 00:14:52.220 ago
SPF algorithm executed 14 times
Area ranges are
Number of LSA 6. Checksum Sum 0x03C06F
Number of opaque link LSA 0. Checksum Sum 0x000000
Number of DCbitless LSA 0
Number of indication LSA 0
Number of DoNotAge LSA 0
Flood list length 0
```

Notice the Router ID (RID) of 10.1.5.1, which is the highest IP address configured on the router.

The show ip ospf database Command

Using the show ip ospf database command will give you information about the number of routers in the internetwork (AS) plus the neighboring router's ID (this is the topology database I mentioned earlier). Unlike the show ip eigrp topology command, this command shows the "OSPF routers," not each and every link in the AS as EIGRP does.

The output is broken down by area. Here's a sample output, again from Corp:

```
Corp#sh ip ospf database
OSPF Router with ID (10.1.5.1) (Process ID 132)
Router Link States (Area 0)
Link ID ADV Router Age Seq# Checksum Link count
10.1.5.1 10.1.5.1 72 0x80000002 0x00F2CA 9
10.1.7.1 10.1.7.1 83 0x80000004 0x009197 6
10.1.9.1 10.1.9.1 73 0x80000001 0x00DA1C 4
10.1.11.1 10.1.11.1 67 0x80000005 0x00666A 4
10.1.12.1 10.1.12.1 67 0x80000004 0x007631 2
Net Link States (Area 0)
Link ID ADV Router Age Seq# Checksum
10.1.11.2 10.1.12.1 68 0x80000001 0x00A337
```

You can see all five routers and the RID of each router (the highest IP address on each router). The router outputs shows the link ID—remember that an interface is also a link—and the RID of the router on that link under the ADV router, or advertising router.

The show ip ospf interface Command

The show ip ospf interface command displays all interface-related OSPF information. Data is displayed about OSPF information for all interfaces or for specified interfaces. (I'll bold some of the important things.)

```
Corp#sh ip ospf interface f0/1
FastEthernet0/1 is up, line protocol is up
Internet Address 10.1.1.1/24, Area 0
Process ID 132, Router ID 10.1.5.1, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DR, Priority 1
Designated Router (ID) 10.1.5.1, Interface address 10.1.1.1
No backup designated router on this network
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
oob-resync timeout 40
Hello due in 00:00:01
Supports Link-local Signaling (LLS)
Index 1/1, flood queue length 0
Next 0x0(0)/0x0(0)
Last flood scan length is 0, maximum is 0
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 0, Adjacent neighbor count is 0
Suppress hello for 0 neighbor(s)
```

The following information is displayed by this command:

- Interface IP address
- Area assignment
- Process ID
- Router ID
- Network type
- Cost
- Priority
- DR/BDR election information (if applicable)
- Hello and Dead timer intervals
- Adjacent neighbor information

The reason I used the show ip ospf interface f0/1 command is that I knew that there would be a designated router elected on the FastEthernet broadcast multi-access network. We'll get into DR and DBR elections in detail in a minute.

The show ip ospf neighbor Command

The show ip ospf neighbor command is super-useful because it summarizes the pertinent OSPF information regarding neighbors and the adjacency state. If a DR or BDR exists, that information will also be displayed. Here's a sample:

```
Corp#sh ip ospf neighbor
Neighbor ID Pri State Dead Time Address Interface
10.1.11.1 0 FULL/ - 00:00:37 10.1.5.2 Serial0/2/0
10.1.9.1 0 FULL/ - 00:00:34 10.1.4.2 Serial0/1/0
10.1.7.1 0 FULL/ - 00:00:38 10.1.3.2 Serial0/0/1
10.1.7.1 0 FULL/ - 00:00:34 10.1.2.2 Serial0/0/0
```

This is a super-important command to understand because it's extremely useful in production networks. Let's take a look at the R3 and 871W routers outputs:

```
R3#sh ip ospf neighbor
Neighbor ID Pri State Dead Time Address Interface
10.1.5.1 0 FULL/ - 00:00:39 10.1.5.1 Serial0/0/1
10.1.11.2 1 FULL/BDR 00:00:31 10.1.11.2 FastEthernet0/1
871W#sh ip ospf nei
Neihbor ID Pri State Dead Time Address Interface
10.1.11.1 1 FULL/DR 00:00:30 10.1.11.1 Vlan1
```

Since there's an Ethernet link (broadcast multi-access) on the Corp router, there's going to be an election to determine who will be the designated router and who will be the non-designated router. You can see that the 871W became the designated router, and it won because it had the highest IP address on the network. You can change this, but that's the default.

The reason that the Corp connections to R1, R2, and R3 don't have a DR or BDR listed in the output is that by default, elections don't happen on point-to-point links. But you can see that the Corp router is fully adjacent to all three routers (and on both connections to R1) from its output.

Configure and verify inter-VLAN routing (Router on a stick)

Configure, verify, and troubleshoot inter-VLAN routing

By default, only hosts that are members of the same VLAN can communicate. To change this and allow inter-VLAN communication, you need a router or a layer 3 switch. I'm going to start with the router approach.

To support ISL or 802.1Q routing on a Fast Ethernet interface, the router's interface is divided into logical interfaces—one for each VLAN. These are called sub-interfaces. From a Fast Ethernet or Gigabit interface, you can set the interface to trunk with the encapsulation command:

```
 ISR#config t
ISR(config)#int f0/0.1
ISR(config-subif)#encapsulation ?
```

```
dot1Q IEEE 802.1Q Virtual LAN
ISR(config-subif)#encapsulation dot1Q ?
<1-4094> IEEE 802.1Q VLAN ID
```

Notice that my 2811 router (named ISR) only supports 802.1Q. We'd need an older-model router to run the ISL encapsulation, but why bother? The sub-interface number is only locally significant, so it doesn't matter which sub-interface numbers are configured on the router. Most of the time, I'll configure a sub-interface with the same number as the VLAN I want to route. It's easy to remember that way, since the sub-interface number is used only for administrative purposes.

It's really important that you understand that each VLAN is a separate subnet. True, I know—they don't have to be. But it really is a good idea to configure your VLANs as separate subnets, so just does that.

Now, I need to make sure you're fully prepared to configure inter-VLAN routing, as well as determine the port IP addresses of hosts connected in a switched VLAN environment. And as always, it's also a good idea to be able to fix any problems that may arise. To set you up for success, let me give you few examples.

you should be able to determine the IP address, masks, and default gateways of each of the hosts in the VLANs.

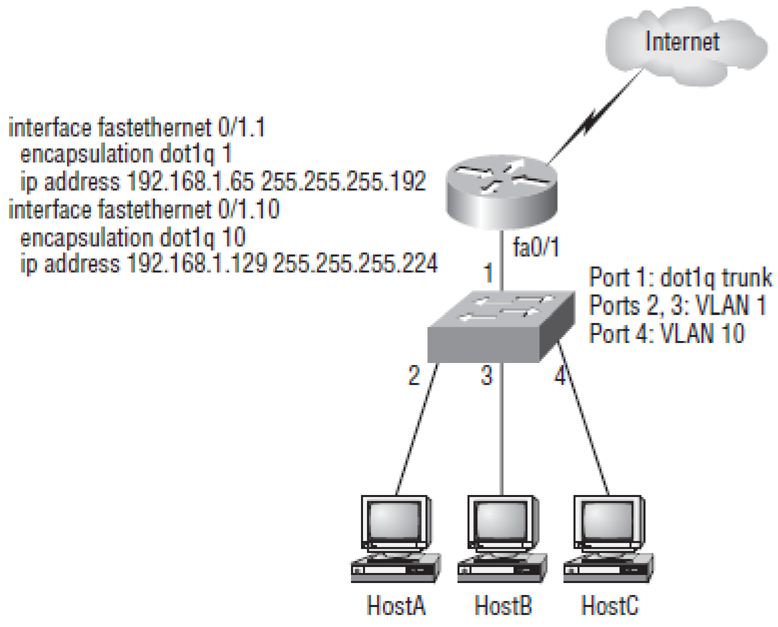

Internet
interface fastethernet 0/1.1
encapsulation dot1q 1
ip address 192.168.1.65 255.255.255.192
interface fastethernet 0/1.10
encapsulation dot1q 10
ip address 192.168.1.129 255.255.255.224
fa0/1
1
Port 1: dot1q trunk
Ports 2, 3: VLAN 1
Port 4: VLAN 10
2
3
4
HostA
HostB
HostC

FIGURE 4.9 Configuring Inter-VLAN example 1

The next step after that is to figure out which subnets are being used. By looking at the router configuration in the figure, you can see that we're using 192.168.1.64/26 with VLAN 1 and 192.168.1.128/27 with VLAN 10. And by looking at the switch configuration, you can see that ports 2 and 3 are in VLAN 1 and port 4 is in VLAN 10. This means that HostA and HostB are in VLAN 1, and HostC is in VLAN 10.

Here's what the hosts' IP addresses should be:

```
HostA: 192.168.1.66, 255.255.255.192, default gateway 192.168.1.65
HostB: 192.168.1.67, 255.255.255.192, default gateway 192.168.1.65
HostC: 192.168.1.130, 255.255.255.224, default gateway 192.168.1.129
```

The hosts could be any address in the range—I just choose the first available IP address after the default gateway address. That wasn't so hard, was it?

Now, again using Figure 4.10, let's go through the commands necessary to configure switch port 1 to establish a link with the router and provide inter-VLAN communication using the IEEE version for encapsulation. Keep in mind that the commands can vary slightly depending on what type of switch you're dealing with.

For a 2960 switch, use the following:

```
2960#config t
2960(config)#interface fa0/1
2960(config-if)#switchport mode trunk
```

As you already know, the 2960 switch can only run the 802.1Q encapsulation, so there's no need to specify it. You can't anyway! For a 3560, it's basically the same, but since it can run ISL and 802.1Q, you have to specify the trunking protocol you're going to use.

Let's take a look at Figure 4.10 and see what we can learn from it. This figure shows three VLANs, with two hosts in each of them.

The router in Figure 4.10 is connected to the fa0/1 switch port, and VLAN 2 is configured on port f0/6. Looking at the diagram, these are the things that Cisco expects you to know:

- The router is connected to the switch using sub-interfaces.
- The switch port connecting to the router is a trunk port.
- The switch ports connecting to the clients and the hub are access ports, not trunk ports.

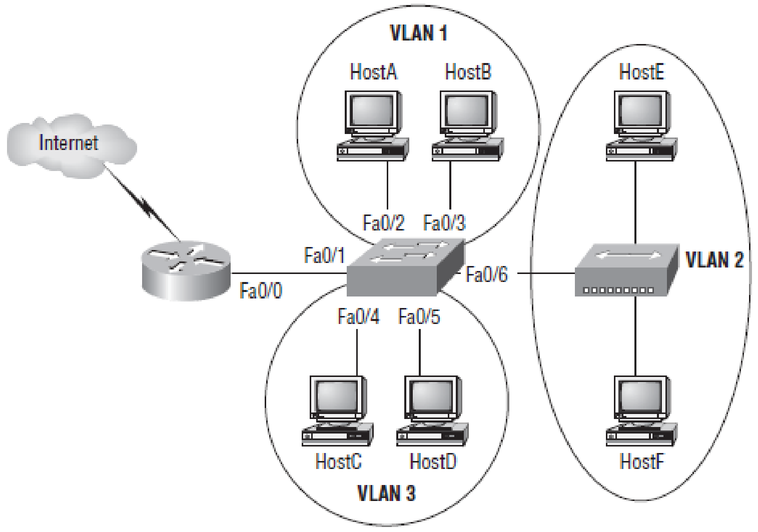


FIGURE 4.10 Inter-VLAN example 2

The configuration of the switch would look something like this:

```
2960#config t
2960(config)#int f0/1
2960(config-if)#switchport mode trunk
2960(config-if)#int f0/2
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/3
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/4
2960(config-if)#switchport access vlan 3
2960(config-if)#int f0/5
2960(config-if)#switchport access vlan 3
2960(config-if)#int f0/6
2960(config-if)#switchport access vlan 2
Before we configure the router, we need to design our logical network:
VLAN 1: 192.168.10.16/28
VLAN 2: 192.168.10.32/28
VLAN 3: 192.168.10.48/28
```

The configuration of the router would then look like this:

```
ISR#config t
ISR(config)#int f0/0
ISR(config-if)#no ip address
ISR(config-if)#no shutdown
ISR(config-if)#int f0/0.1
ISR(config-subif)#encapsulation dot1q 1
ISR(config-subif)#ip address 192.168.10.17 255.255.255.240
ISR(config-subif)#int f0/0.2
ISR(config-subif)#encapsulation dot1q 2
ISR(config-subif)#ip address 192.168.10.33 255.255.255.240
ISR(config-subif)#int f0/0.3
ISR(config-subif)#encapsulation dot1q 3
ISR(config-subif)#ip address 192.168.10.49 255.255.255.240
```

The hosts in each VLAN would be assigned an address from their subnet range, and the default gateway would be the IP address assigned to the router's sub-interface in that VLAN. Now, let's take a look at another figure and see if you can determine the switch and router configurations without looking at the answer—no cheating! Figure 4.11 shows a router connected to a 2960 switch with two VLANs. One host in each VLAN is assigned an IP address. What are your router and switch configurations based on these IP addresses?

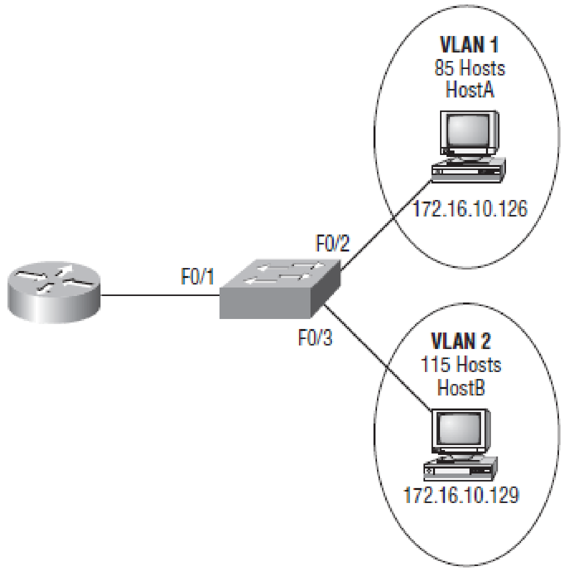


FIGURE 4.11 Inter-VLAN examples 3

Since the hosts don't list a subnet mask, you have to look for the number of hosts used in each VLAN to figure out the block size. VLAN 1 has 85 hosts and VLAN 2 has 115 hosts. Each of these will fit in a block size of 128, which is a /25 mask, or 255.255.255.128. You should know by now that the subnets are 0 and 128; the 0 subnet (VLAN 1) has a host range of 1–126, and the 128 subnet (VLAN 2) has a range of 129–254. You can almost be fooled since HostA has an IP address of 126, which makes it almost seem that HostA and B are in the same subnet. But they're not, and you're way too smart by now to be fooled by this one!

Here is the switch configuration:

```
2960#config t
2960(config)#int f0/1
2960(config-if)#switchport mode trunk
2960(config-if)#int f0/2
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/3
```

```
2960(config-if)#switchport access vlan 2
```

Here is the router configuration:

```
ISR#config t
ISR(config)#int f0/0
ISR(config-if)#no ip address
ISR(config-if)#no shutdown
ISR(config-if)#int f0/0.1
ISR(config-subif)#encapsulation dot1q 1
ISR(config-subif)#ip address 172.16.10.1 255.255.255.128
ISR(config-subif)#int f0/0.2
ISR(config-subif)#encapsulation dot1q 2
ISR(config-subif)#ip address 172.16.10.254 255.255.255.128
```

I used the first address in the host range for VLAN 1 and the last address in the range for VLAN 2, but any address in the range would work. You just have to configure the host's default gateway to whatever you make the router's address.

Now, before we go on to the next example, I need to make sure that you know how to set the IP address on the switch. Since VLAN 1 is typically the administrative VLAN, we'll use an IP address from that pool of addresses. Here's how to set the IP address of the switch (I'm not nagging, but you really should already know this!):

```
2960#config t
2960(config)#int vlan 1
2960(config-if)#ip address 172.16.10.2 255.255.255.128
2960(config-if)#no shutdown
```

Yes, you have to do a no shutdown on the VLAN interface. One more example, and then we'll move on to VTP—another important subject that you definitely don't want to miss! In Figure 2.26 there are two VLANs. By looking at the router configuration, what's the IP address, mask, and default gateway of HostA? Use the last IP address in the range for HostA's address:

If you really look carefully at the router configuration (the hostname in this figure is just Router), there is a simple and quick answer. Both subnets are using a /28, or 255.255.255.240 mask, which is a block size of 16. The router's address for VLAN 1 is in subnet 128. The next subnet is 144, so the broadcast address of VLAN 1 is 143 and the valid host range is 129–142. So, the host address would be this:

```
IP Address: 192.168.10.142
Mask: 255.255.255.240
Default Gateway: 192.168.10.129
```

Configure SVI interfaces

Configure SVI Intefaces

Cisco offers three types of integrated switching modules for the modular Cisco 3800, 2800, and 1800 Series Integrated Services Routers: the 16- and 36-port Cisco EtherSwitch® modules, the Cisco EtherSwitch 4- and 9-port high-speed WAN interface cards (HWICs), and the Cisco EtherSwitch service modules. In addition, the Cisco 1800 Series fixed-configuration Integrated Services Routers are integrated with an 8-port switch. The Cisco 870 and 850 Series Integrated Services Routers are integrated with a 4-port switch.

The integrated switch ports for the fixed-configuration Integrated Services Routers and the switch ports on the HWICs do not natively support Layer 3 addresses or Layer 3 features. They must be assigned to a SVI and use a VLAN interface for Layer 3 features. SVI represents a logical Layer 3 interface on a switch. In addition to basic routing, SVI can be used to support additional features for the network that the SVI represents.

SVI on Cisco Integrated Services Routers is designed to provide basic Layer 3 functions for the Layer 2 switch ports that belong to a specific VLAN. The SVI does not provide the same feature set and functions as the integrated Layer 3 Ethernet ports of the integrated services routers and should not be used to entirely replace the Layer 3 Ethernet ports. Customer who needs additional Layer 3 Ethernet ports for their Integrated Services Routers may consider the use of 1- and 2-Port Fast Ethernet High-Speed WIC for Cisco 1841, 2800, and 3800 series.

IP Services

Configure and verify DHCP (IOS Router)

Configure, verify, and troubleshoot DHCP and DNS operation on a router (including CLI/SDM)

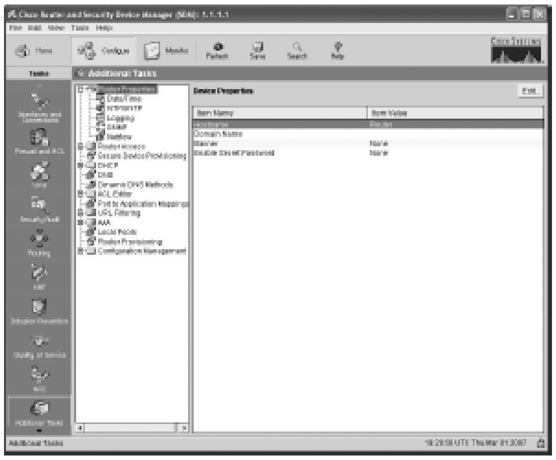
Instead of using a server to provide IP addresses to clients, you can configure a router with a DHCP scope that will hand out IP addresses to hosts on a connected interface. Here is an example:

```
R2#config t
R2(config)#ip dhcp pool Admin
R2(dhcp-config)#network 10.1.8.0 255.255.255.0
R2(dhcp-config)#default-router 10.1.8.1
R2(dhcp-config)#exit
R2(config)#ip dhcp excluded-address 10.1.8.1
R2(config)#
```

Creating DHCP pools on a router is actually a pretty simple process. To do so, you just create the pool name, add the network/subnet and the default gateway, and exclude any addresses you don't want handed out (like the default gateway address). And you'd usually add a DNS server as well. Note that the excluded addresses are set from global configuration mode. But wait, we're not done. Even though we created the pool, which hosts can use this pool? Now this is where the interface configuration comes in. Notice the default-router address in the pool configuration? Any DHCP client connected off of interface fastethernet 0/0 will grab a DHCP address from the pool. Here is the finished configuration:

```
R2(config)#interface fa0/0
R2(config-if)#ip address 10.1.8.1 255.255.255.0
```

Configuring a pool from SDM is actually just as easy, if not easier, than doing so from the CLI. After you open SDM, look down at the bottom left portion of the wizard page and click the Additional Tasks button. From there, just click the Router Properties icon.



Here, you can set the hostname, MOTD banner, and enable secret password. Last, I clicked on the DHCP folder, then the DHCP pool icon. I then clicked Add and created a DHCP pool on my router.

Now, let’s take a look at the configuration on the router:

```
Todd#sh run
Building configuration...
[output cut]
hostname Todd
!
ip domain name lammle.com
[output cut]
ip dhcp excluded-address 172.16.10.1
ip dhcp excluded-address 172.16.10.11 172.16.10.254
!
ip dhcp pool Todd's_LAN
import all
network 172.16.10.0 255.255.255.0
!
```

Using DNS to Resolve Names

If you have a lot of devices and don’t want to create a host table in each device, you can use a DNS server to resolve hostnames.

Any time a Cisco device receives a command it doesn’t understand, it will try to resolve it through DNS by default. Watch what happens when I type the special command todd at a Cisco router prompt:

```
Corp#todd
Translating "todd"...domain server (255.255.255.255)
Translating "todd"...domain server (255.255.255.255)
Translating "todd"...domain server (255.255.255.255)
% Unknown command or computer name, or unable to find
computer address
Corp#
```

It doesn’t know my name or what command I am trying to type, so it tries to resolve this through DNS. This is really annoying for two reasons: first, because it doesn’t know my name, and second, because I need to hang out and wait for the name lookup to time out. You can get around this and prevent a time-consuming DNS lookup by using the no ip domain-lookup command on your router from global configuration mode.

If you have a DNS server on your network, you need to add a few commands to make DNS name resolution work:

- The first command is ip domain-lookup, which is turned on by default. It needs to be entered only if you previously turned it off (with the no ip domain-lookup command). The command can be used without the hyphen as well (ip domain lookup).
- The second command is ip name-server. This sets the IP address of the DNS server. You can enter the IP addresses of up to six servers.
- The last command is ip domain-name. Although this command is optional, it really should be set. It appends the domain name to the hostname you type in. Since DNS uses a FQDN system, you must have a full DNS name, in the form domain.com.

Here's an example of using these three commands:

```
Corp#config t
Corp(config)#ip domain-lookup
Corp(config)#ip name-server ?
A.B.C.D Domain server IP address (maximum of 6)
Corp(config)#ip name-server 192.168.0.70
Corp(config)#ip domain-name lammle.com
Corp(config)#^Z
Corp#
```

After the DNS configurations are set, you can test the DNS server by using a hostname to ping or telnet a device like this:

```
Corp#ping R1
Translating "R1"...domain server (192.168.0.70) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.2.2.2, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max
= 28/31/32 ms
```

Notice that the router uses the DNS server to resolve the name.

Describe the types, features, and applications of ACLs

Describe the purpose and types of ACLs

An access list is essentially a list of conditions that categorize packets. They can be really helpful when you need to exercise control over network traffic. An access list would be your tool of choice for decision making in these situations.

One of the most common and easiest to understand uses of access lists is filtering unwanted packets when implementing security policies. For example, you can set them up to make very specific decisions about regulating traffic patterns so that they'll allow only certain hosts to access web resources on the Internet while restricting others. With the right combination of access lists, network managers arm themselves with the power to enforce nearly any security policy they can invent.

Access lists can even be used in situations that don't necessarily involve blocking packets. For example, you can use them to control which networks will or won't be advertised by dynamic routing protocols. How you configure the access list is the same. The difference here is simply how you apply it—to a routing protocol instead of an interface. When you apply an access list in this way, it's called a distribute list, and it doesn't stop routing advertisements, it just controls their content. You can also use access lists to categorize packets for queuing or QoS-type services and for controlling which types of traffic can activate a pricey ISDN link.

Creating access lists is really a lot like programming a series of if-then statements—if a given condition is met, then a given action is taken. If the specific condition isn't met, nothing happens and the next statement is evaluated. Access-list statements are basically packet filters that packets are compared against, categorized by, and acted upon accordingly. Once the lists are built, they can be applied to either inbound or outbound traffic on any interface. Applying an access list causes the router to analyze every packet crossing that interface in the specified direction and take the appropriate action.

There are a few important rules that a packet follows when it's being compared with an access list:

- It's always compared with each line of the access list in sequential order—that is, it'll always start with the first line of the access list, then go to line 2, then line 3, and so on.
- It's compared with lines of the access list only until a match is made. Once the packet matches the condition on a line of the access list, the packet is acted upon and no further comparisons take place.
- There is an implicit "deny" at the end of each access list—this means that if a packet doesn't match the condition on any of the lines in the access list, the packet will be discarded. Each of these rules has some powerful implications when filtering IP packets with access lists, so keep in mind that creating effective access lists truly takes some practice.

There are two main types of access lists:

Standard access lists

These use only the source IP address in an IP packet as the condition test. All decisions are made based on the source IP address. This means that standard access lists basically permit or deny an entire suite of protocols. They don't distinguish among any of the many types of IP traffic such as web, Telnet, UDP, and so on.

Extended access lists
Extended access lists can evaluate many of the other fields in the layer 3 and layer 4 headers of an IP packet. They can evaluate source and destination IP addresses, the protocol field in the Network layer header, and the port number at the Transport layer header.

This gives extended access lists the ability to make much more granular decisions when controlling traffic.

Named access lists
Hey, wait a minute—I said there were two types of access lists but listed three! Well, technically there really are only two since named access lists are either standard or extended and not actually a new type. I'm just distinguishing them because they're created and referred to differently than standard and extended access lists, but they're functionally the same.

Configure and verify ACLs in a network environment

Configure and apply ACLs based on network filtering requirements (including CLI/SDM)
For the CCNA exam prep, we will look at two types of access lists; standard IP access lists, extended IP access lists, and named access lists, which is another way of configuring standard and extended access lists. We will also look at a technique for specifying ranges of addressing called wildcard masking that can be used with all three types of access list. For now, let's get started on standard access lists!

Standard IP Access Lists
Standard IP access lists filter network traffic by examining the source IP address in a packet. You create a standard IP access list by using the access-list numbers 1–99 or 1300–1999 (expanded range). Access-list types are generally differentiated using a number. Based on the number used when the access list is created, the router knows which type of syntax to expect as the list is entered.

By using numbers 1–99 or 1300–1999, you're telling the router that you want to create a standard IP access list, so the router will expect syntax specifying only the source IP address in the test lines.

The following is an example of the many access-list number ranges that you can use to filter traffic on your network (the protocols for which you can specify access lists depend on your IOS version):

```
Corp(config)#
access-list ?
<1-99> IP standard access list
<100-199> IP extended access list
<1100-1199> Extended 48-bit MAC address access list
<1300-1999> IP standard access list (expanded range)
<200-299> Protocol type-code access list
<2000-2699> IP extended access list (expanded range)
<700-799> 48-bit MAC address access list
compiled Enable IP access-list compilation
dynamic-extended Extend the dynamic ACL absolute timer
rate-limit Simple rate-limit specific access list
```

Let’s take a look at the syntax used when creating a standard access list:

```
Corp(config)#
access-list 10 ?
deny Specify packets to reject
permit Specify packets to forward
remark Access list entry comment
```

As I said, by using the access-list numbers 1–99 or 1300–1999, you’re telling the router that you want to create a standard IP access list.

After you choose the access-list number, you need to decide whether you’re creating a permit or deny statement. For this example, you will create a deny statement:

```
Corp(config)#
access-list 10 deny ?
Hostname or A.B.C.D Address to match
any Any source host
host A single host address
```

The next step requires a more detailed explanation. There are three options available. You can use the any parameter to permit or deny any host or network, you can use an IP address to specify either a single host or a range of them, or you can use the host command to specify a specific host only. The any command is pretty obvious—any source address matches the statement, so every packet compared against this line will match. The host command is relatively simple. Here’s an example using it:

```
Corp(config)#
```

```
access-list 10 deny host ?
Hostname or A.B.C.D Host address
Corp(config)#
access-list 10 deny host 172.16.30.2
```

This tells the list to deny any packets from host 172.16.30.2. The default parameter is
Host In other words, if you type access-list 10 deny 172.16.30.2 , the router assumes you that mean host 172.16.30.2.

But there's another way to specify either a particular host or a range of hosts—you can use wildcard masking. In fact, to specify any range of hosts, you have to use wildcard masking in the access list.

Wildcard Masking

Wildcards are used with access lists to specify an individual host, a network, or a certain range of a network or networks. To understand a wildcard, you need to understand what a block size is; it's used to specify a range of addresses. Some of the different block sizes available are 64, 32, 16, 8, and 4.

When you need to specify a range of addresses, you choose the next-largest block size for your needs. For example, if you need to specify 34 networks, you need a block size of 64. If you want to specify 18 hosts, you need a block size of 32. If you only specify 2 networks, then a block size of 4 would work.

Wildcards are used with the host or network address to tell the router a range of available addresses to filter. To specify a host, the address would look like this:

172.16.30.5 0.0.0.0

The four zeros represent each octet of the address. Whenever a zero is present, it means that octet in the address must match exactly. To specify that an octet can be any value, the value of 255 is used. As an example, here's how a /24 subnet is specified with a wildcard:

172.16.30.0 0.0.0.255

This tells the router to match up the first three octets exactly, but the fourth octet can be any value.

Now, that was the easy part. What if you want to specify only a small range of subnets? This is where the block sizes come in. You have to specify the range of values in a block size. In other words, you can't choose to specify 20 networks. You can only specify the exact amount as the block size value. For example, the range would have to be either 16 or 32, but not 20. Let's say that you want to block access to part of the network that is in the range from 172.16.8.0 through 172.16.15.0. That is a block size of 8. Your network number would be 172.16.8.0, and the wildcard would be 0.0.7.255. Whoa! What is that? The 7.255 is what the router uses to determine the block size. The network and wildcard tell the router to start at 172.16.8.0 and go up a block size of eight addresses to network 172.16.15.0.

Seriously—it really is easier than it looks—really! I could certainly go through the binary math for you, but no one needs that. Actually, all you have to do is remember that the wildcard is always one number less than the block size. So, in our example, the wildcard would be 7 since our block size is 8. If you used a block size of 16, the wildcard would be 15. Easy, huh? But just in case, we'll go through some examples to help you nail it. The following example tells the router to match the first three octets exactly but that the fourth octet can be anything:

```
Corp(config)#
access-list 10 deny 172.16.10.0 0.0.0.255
The next example tells the router to match the first two octets and that the
last two octets
can be any value:
Corp(config)#
access-list 10 deny 172.16.0.0
0.0.255.255
Try to figure out this next line:
Corp(config)#
access-list 10 deny 172.16.16.0 0.0.3.255
```

This configuration tells the router to start at network 172.16.16.0 and use a block size of 4. The range would then be 172.16.16.0 through 172.16.19.0. The following example shows an access list starting at 172.16.16.0 and going up a block size of 8 to 172.16.23.0:

```
Corp(config)#
access-list 10 deny 172.16.16.0 0.0.7.255
```

The next example starts at network 172.16.32.0 and goes up a block size of 16 to 172.16.47.0:

```
Corp(config)#
access-list 10 deny 172.16.32.0 0.0.15.255
```

The next example starts at network 172.16.64.0 and goes up a block size of 64 to 172.16.127.0:

```
Corp(config)#
access-list 10 deny 172.16.64.0 0.0.63.255
```

The last example starts at network 192.168.160.0 and goes up a block size of 32 to 192.168.191.255:

```
Corp(config)#
access-list 10 deny 192.168.160.0 0.0.31.255
```

Here are two more things to keep in mind when working with block sizes and wildcards:

- Each block size must start at 0 or a multiple of the block size. For example, you can't say that you want a block size of 8 and then start at 12. You must use 0–7, 8–15, 16–23, and so on. For a block size of 32, the ranges are 0–31, 32–63, 64–95, and so on.

- The command any is the same thing as writing out the wildcard 0.0.0.0255.255.255.255.

Standard Access List Example
In Figure 5.1, a router has three LAN connections and one WAN connection to the Internet. Users on the Sales LAN should not have access to the Finance LAN, but they should be able to access the Internet and the marketing department. The Marketing LAN needs to access the Finance LAN for application services.

On the router in the figure, the following standard IP access list is configured:

```
Lab_A#config t
Lab_A(config)#access-list 10 deny 172.16.40.0 0.0.0.255
Lab_A(config)#access-list 10 permit any
```

It's very important to know that the any command is the same thing as saying the following using wildcard masking:

```
Lab_A(config)#access-list 10 permit 0.0.0.0 255.255.255.255
```

Since the wildcard mask says that none of the octets is to be evaluated, every address matches the test condition. So, this is functionally the same as using the any keyword.

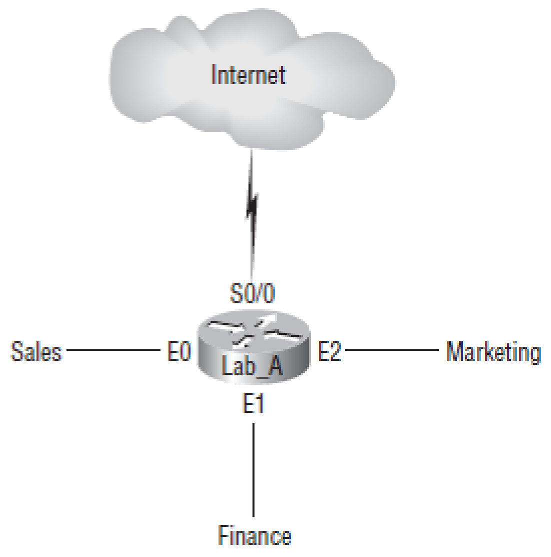


FIGURE 5 . 1 IP access list example with three LANs and a WAN connection

At this point, the access list is configured to deny source addresses from the Sales LAN access to the Finance LAN and allow everyone else. But remember, no action will be taken until the access list is applied on an interface in a specific direction. But where should this access list be placed? If you place it as an incoming access list on E0, you might as well shut down the Ethernet interface because all of the Sales LAN devices will be denied access to all networks attached to the router. The best place to apply this access list is on the E1 interface as an outbound list:

```
Lab_A(config)#int e1
Lab_A(config-if)#ip access-group 10 out
```

This completely stops traffic from 172.16.40.0 from getting out Ethernet 1. It has no effect on the hosts from the Sales LAN accessing the Marketing LAN and the Internet since traffic to those destinations doesn't go through interface E1. Any packet trying to exit out E1 will have to go through the access list first. If there were an inbound list placed on E0, then any packet trying to enter interface E0 would have to go through the access list before being routed to an exit interface.

Let's take a look at another example of a standard access list. Figure 7.2 shows an internetwork of two routers with three LANs and one serial WAN connection.

You want to stop the Accounting users from accessing the Human Resources server attached to the Lab_B router but allow all other users' access to that LAN. What standard access list would you create and where would you place it?

The real answer is that you should use an extended access list and place it closest to the source, but the question specifies that you should use a standard access list. Standard access lists, by rule of thumb, are placed closest to the destination—in this example, Ethernet 0 outbound on the Lab_B router. Here is the access list that should be placed on the Lab_B router:

```
Lab_B#config t
Lab_B(config)#access-list 10 deny 192.168.10.128 0.0.0.31
Lab_B(config)#access-list 10 permit any
Lab_B(config)#interface Ethernet 0
Lab_B(config-if)#ip access-group 10 out
```

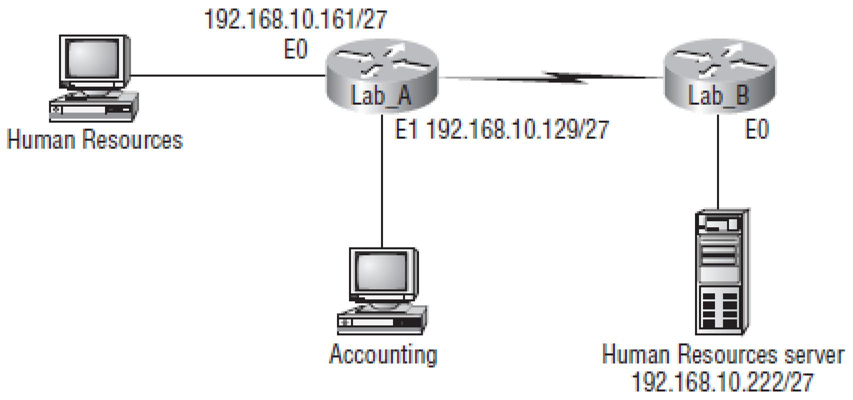


FIGURE 5.2 IP standard access list example 2

Before we move on to restricting Telnet access on a router, let’s take a look at one more standard access list example, but it will require some thought. In Figure 5.3 you have a router with four LAN connections and one WAN connection to the Internet.

You need to write an access list that will stop access from each of the four LANs shown in the diagram to the Internet. Each of the LANs shows a single host’s IP address, and from that you need to determine the subnet and use wildcards to configure the access list.

Here is an example of what your answer should look like (starting with the network on E0 and working through to E3):

```
Router(config)#access-list 1 deny 172.16.128.0 0.0.31.255
Router(config)#access-list 1 deny 172.16.48.0 0.0.15.255
Router(config)#access-list 1 deny 172.16.192.0 0.0.63.255
Router(config)#access-list 1 deny 172.16.88.0 0.0.7.255
Router(config)#access-list 1 permit any
Router(config)#interface serial 0
Router(config-if)#ip access-group 1 out
```

Okay, what would be the purpose of creating this list? If you actually applied this access list on the router, you'd effectively shut down access to the Internet, so what's the purpose of even having an Internet connection? I wrote this exercise so you can practice how to use block sizes with access lists—which is critical for your success when studying the CCNA objectives.

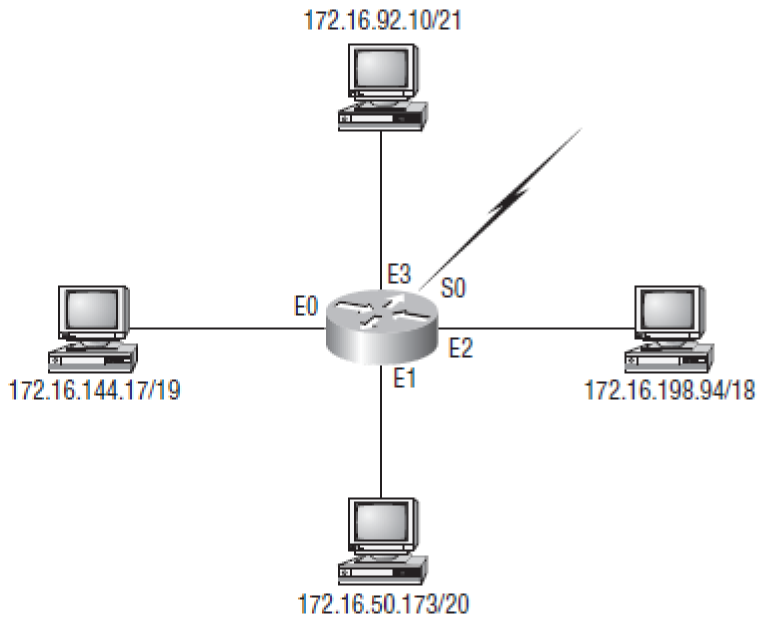


FIGURE 5.3 IP standard access list example 3

Controlling VTY (Telnet) Access

You'll probably have a difficult time trying to stop users from telnetting to a large router because any active interface on a router is fair game for VTY access. You could try to create an extended IP access list that limits Telnet access to every IP address on the router. But if you did that, you'd have to apply it inbound on every interface, and that really wouldn't scale well to a large router with dozens, even hundreds, of interfaces, would it? Here's a much better solution: Use a standard IP access list to control access to the VTY lines themselves.

Why does this work? Because when you apply an access list to the VTY lines, you don't need to specify the Telnet protocol since access to the VTY implies terminal access. You also don't need to specify a destination address, since it really doesn't matter which interface address the user used as a target for the Telnet session. You really only need to control where the user is coming from—their source IP address.

To perform this function, follow these steps:

1. Create a standard IP access list that permits only the host or hosts you want to be able to telnet into the routers.
2. Apply the access list to the VTY line with the access-class command.

Here is an example of allowing only host 172.16.10.3 to telnet into a router:

```
Lab_A(config)#access-list 50 permit 172.16.10.3
Lab_A(config)#line vty 0 4
Lab_A(config-line)#access-class 50 in
```

Extended Access Lists

In the standard IP access list example earlier, notice how you had to block all access from the Sales LAN to the finance department. What if you needed Sales to gain access to a certain server on the Finance LAN but not to other network services, for security reasons? With a standard IP access list, you can't allow users to get to one network service and not another.

Said another way, when you need to make decisions based on both source and destination addresses, a standard access list won't allow you to do that since it only makes decisions based on source address.

But an extended access list will hook you up. That's because extended access lists allow you to specify source and destination address as well as the protocol and port number that identify the upper-layer protocol or application. By using extended access lists, you can effectively allow users access to a physical LAN and stop them from accessing specific hosts—or even specific services on those hosts.

Here's an example of an extended IP access list:

```
Corp(config)#access-list ?
<1-99> IP standard access list
<100-199> IP extended access list
<1100-1199> Extended 48-bit MAC address access list
<1300-1999> IP standard access list (expanded range)
<200-299> Protocol type-code access list
<2000-2699> IP extended access list (expanded range)
<700-799> 48-bit MAC address access list
compiled Enable IP access-list compilation
dynamic-extended Extend the dynamic ACL absolute timer
rate-limit Simple rate-limit specific access list
```

The first command shows the access-list numbers available. You'll use the extended accesslist range from 100 to 199. Be sure to notice that the range 2000–2699 is also available for extended IP access lists.

At this point, you need to decide what type of list entry you are making. For this example, you'll choose a deny list entry.

```
Corp(config)#access-list 110 ?
deny Specify packets to reject
dynamic Specify a DYNAMIC list of PERMITs or DENYs
permit Specify packets to forward
remark Access list entry comment
```

Once you choose the access-list type, you then need to select a protocol field entry.

```
Corp(config)#access-list 110 deny ?
<0-255> An IP protocol number
ahp Authentication Header Protocol
eigrp Cisco's EIGRP routing protocol
esp Encapsulation Security Payload
gre Cisco's GRE tunneling
icmp Internet Control Message Protocol
igmp Internet Gateway Message Protocol
ip Any Internet Protocol
ipinip IP in IP tunneling
nos KA9Q NOS compatible IP over IP tunneling
ospf OSPF routing protocol
pcp Payload Compression Protocol
pim Protocol Independent Multicast
tcp Transmission Control Protocol
udp User Datagram Protocol
```

Identify the basic operation of NAT

Explain the basic operation of NAT

Similar to Classless Inter-Domain Routing (CIDR), the original intention for NAT was to slow the depletion of available IP address space by allowing many private IP addresses to be represented by some smaller number of public IP addresses.

Since then, it’s been discovered that NAT is also a useful tool for network migrations and mergers, server load sharing, and creating “virtual servers.”

At times, NAT really decreases the overwhelming amount of public IP addresses required in your networking environment. And NAT comes in really handy when two companies that have duplicate internal addressing schemes merge. NAT is also great to have around when an organization changes its Internet service provider (ISP) and the networking manager doesn’t want the hassle of changing the internal address scheme.

Here’s a list of situations when it’s best to have NAT on your side:

- You need to connect to the Internet and your hosts don’t have globally unique IP addresses.

- You change to a new ISP that requires you to renumber your network.
- You need to merge two intranets with duplicate addresses.

You typically use NAT on a border router. For an illustration of this, see Figure 5.4. Now you may be thinking, "NAT's totally cool. It's the grooviest greatest network gadget, and I just gotta have it." Well, hang on a minute. There are truly some serious snags related to NAT use. Oh—don't get me wrong: It really can save you sometimes, but there's a dark side you need to know about, too. For a visual of the pros and cons linked to using NAT, check out Table 5.1.

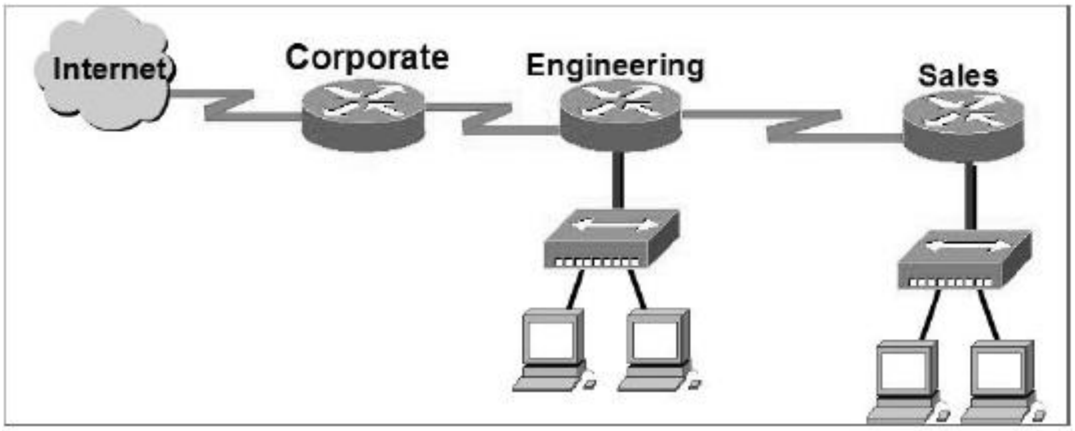


FIGURE 5.4 Where to configure NAT

TABLE 5.1 Advantages and Disadvantages of Implementing NAT

Advantages	Disadvantages
Conserves legally registered addresses.	Translation introduces switching path delays
Reduces address overlap occurrence.	Loss of end-to-end IP traceability.
Increases flexibility when connecting to Internet.	Certain applications will not function with NAT enabled.
Eliminates address renumbering as network changes.	

Types of Network Address Translation

Static NAT This type of NAT is designed to allow one-to-one mapping between local and global addresses. Keep in mind that the static version requires you to have one real Internet IP address for every host on your network.

Dynamic NAT This version gives you the ability to map an unregistered IP address to a registered IP address from out of a pool of registered IP addresses. You don't have to statically configure your router to map an inside to an outside address as you would using static NAT, but you do have to have enough real, bona fide IP addresses for everyone who's going to be sending packets to and receiving them from the Internet.

Configure and verify NAT for given network requirements

Configure NAT for given network requirements using (including CLI/SDM)

Then I'll show you how to configure NAT on a Cisco router using the Secure Device Manager (SDM).

Static NAT Configuration

Let's take a look at a simple basic static NAT configuration:

```
ip nat inside source static 10.1.1.1 170.46.2.2
!
interface Ethernet0
ip address 10.1.1.10 255.255.255.0
ip nat inside
!
interface Serial0
ip address 170.46.2.1 255.255.255.0
ip nat outside
!
```

Dynamic NAT Configuration

Dynamic NAT means that we have a pool of addresses that we will use to provide real IP addresses to a group of users on the inside. We do not use port numbers, so we have to have real IP addresses for every user trying to get outside the local network.

Here is a sample output of a dynamic NAT configuration:

```
ip nat pool todd 170.168.2.2 170.168.2.254
netmask 255.255.255.0
ip nat inside source list 1 pool todd
!
interface Ethernet0
ip address 10.1.1.10 255.255.255.0
ip nat inside
!
interface Serial0
ip address 170.168.2.1 255.255.255.0
ip nat outside
!
access-list 1 permit 10.1.1.0 0.0.0.255
!
```

PAT (Overloading) Configuration

This last example shows how to configure inside global address overloading. This is the typical NAT that we would use today. It is rare that we would use static or dynamic NAT unless we were statically mapping a server, for example.

Here is a sample output of a PAT configuration:

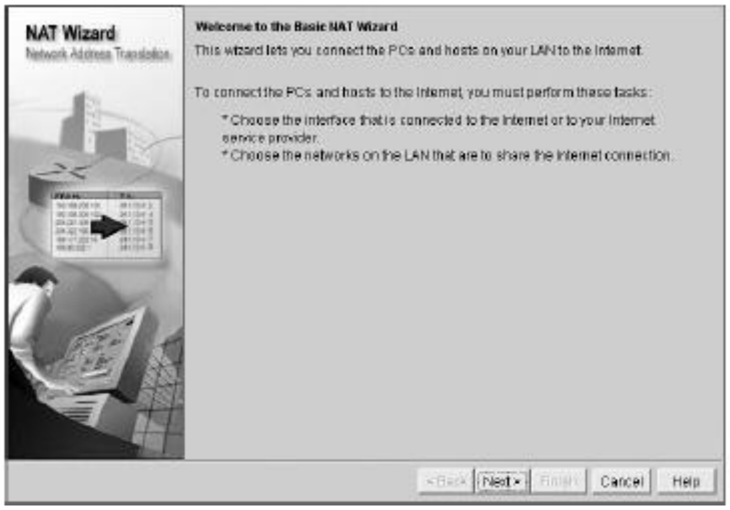
```
ip nat pool globalnet 170.168.2.1 170.168.2.1
netmask 255.255.255.0
ip nat inside source list 1 pool globalnet overload
!
interface Ethernet0/0
ip address 10.1.1.10 255.255.255.0
ip nat inside
!
interface Serial0/0
ip address 170.168.2.1 255.255.255.0
ip nat outside
!
access-list 1 permit 10.1.1.0 0.0.0.255
```

Configuring NAT using the SDM

Anyway, all you have to do is click Configure ——————————————— NAT and you get a handy wizard that does a lot more than just hold your hand to create a NAT rule. You get to pick between a basic and advanced wizard:

Basic NAT Use this wizard if you have some basic PCs/hosts on your trusted network that need access to the Internet. This wizard will guide you through the process of creating a basic NAT configuration.

Advanced NAT If you have a DMZ, or servers on your inside network that user from the outside need to access, you definitely want to opt for the Advanced NAT configuration. The first screen is the Create NAT Configuration screen (see Figure 5.5).



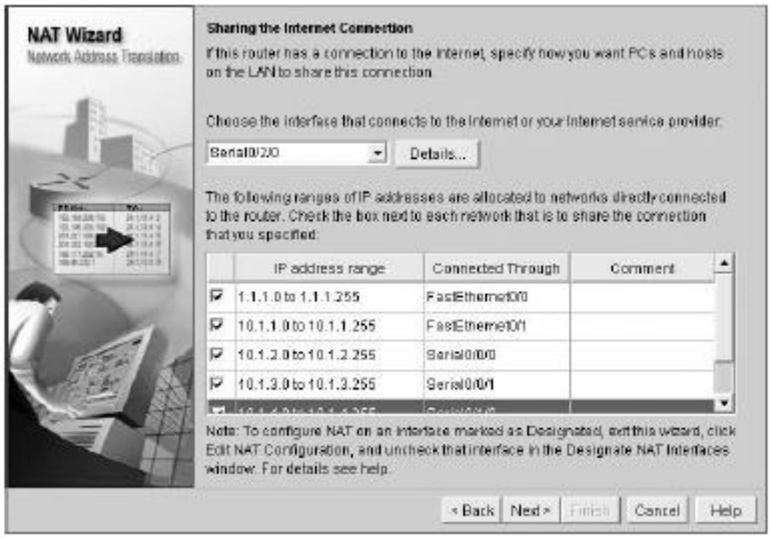
Welcome to the Basic NAT Wizard

This wizard lets you connect the PCs and hosts on your LAN to the Internet.

To connect the PCs and hosts to the Internet, you must perform these tasks:

* Choose the interface that is connected to the Internet or to your Internet service provider.
* Choose the networks on the LAN that are to share the Internet connection.

From here, I'm just going to simply connect up and create a basic NAT. After that, I click Launch the Selected Task, and get the next screen, which tells me what the Basic NAT Wizard is going to do (see Figure 5.6).



Sharing the Internet Connection

If this router has a connection to the Internet, specify how you want PCs and hosts on the LAN to share this connection.

Choose the interface that connects to the Internet or your Internet service provider:

Serial0/2/0

The following ranges of IP addresses are allocated to networks directly connected to the router. Check the box next to each network that is to share the connection that you specified:

	IP address range	Connected Through	Comment
✓	1.1.1.0 to 1.1.1.255	FastEthernet0/0	
✓	10.1.1.0 to 10.1.1.255	FastEthernet0/1	
✓	10.1.2.0 to 10.1.2.255	Serial0/0/0	
✓	10.1.3.0 to 10.1.3.255	Serial0/0/1	

Note: To configure NAT on an interface marked as Designated, exit this wizard, click Edit NAT Configuration, and uncheck that interface in the Designate NAT Interfaces window. For details see help.

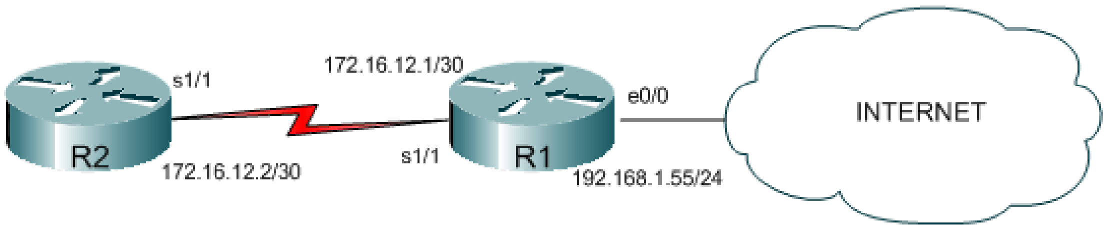
Configure and verify NTP as a client

Configure and verify NTP as a client

"Time is inherently important to the function of routers and networks. It provides the only frame of reference between all devices on the network. This makes synchronized time extremely important. Without synchronized time, accurately correlating information between devices becomes difficult, if not impossible. When it comes to security, if you cannot successfully compare logs between each of your routers and all your network servers, you will find it very hard to develop a reliable picture of an incident. Finally, even if you are able to put the pieces together, unsynchronized times, especially between log files, may give an attacker with a good attorney enough wiggle room to escape prosecution." –Thomas Akin, in Hardening Cisco Routers.

NTP provides us (the network, systems, and security guys) with an easy way to ensure that all of our network devices have the same time. In smaller networks, this is desirable; in large networks, it is a must. In this lab, we'll configure one Cisco router as both a client (synchronizing time from hosts on the Internet) and server (providing time synchronization to other internal devices).

The physical topology looks like this:



For our demonstration, R1 will be both an NTP client and NTP server while R2 will only be an NTP client. Because we will be synchronizing R1′s time with a host on the Internet, R1 will need connectivity to the Internet for this demonstration; R2 will not. R1′s actual connectivity to the Internet is irrelevant and out of the scope of this demonstration.

- R1 will synchronize time against nist.netservicesgroup.com (64.113.32.5).
- R2 will synchronize time against R1 (172.16.12.1).
- NTP synchronization between R1 and R2 will be authenticated, using a key of "ThereIsTimeForEverything".

Configuring R1 as an NTP client

Before we get started, I first want to show that we currently have an unsynchronized clock:

```
R1# show clock detail *00:01:19.099 UTC Fri Mar 1 2002 No time source R1#
```

Configuring a Cisco router to act as an NTP client is extremely simple. We'll use the "ntp server" command, followed by the IP address of the NTP server:

```
R1# configure terminal Enter configuration commands, one per line. End with
CNTL/Z. R1(config)# ntp server nist.netservicesgroup.com Translating
"nist.netservicesgroup.com"...domain server (192.168.1.12) [OK] R1(config)#
```

That's about all there is to it, from the client side. Note that if you do not have DNS configured on your devices, you'll need to use IP addresses instead of fully qualified domain names (above). If we take a look at our clock, we'll see that the time has been updated from NTP:

```
R1# show clock detail 03:15:52.593 UTC Sun Nov 23 2008 Time source is NTP
```

Let's verify our NTP associations and status:

```
R1# show ntp associations address ref clock st when poll reach delay offset
disp *~64.113.32.5 .ACTS. 1 45 64 17 36.2 -2.90 1901.2 * master (synced), #
master (unsynced), + selected, - candidate, ~ configured
R1# show ntp status Clock is synchronized, stratum 2, reference is
64.113.32.5 nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz,
precision is 2**18 reference time is CCD34DAA.B4D97B06 (03:34:02.706 UTC Sun
Nov 23 2008) clock offset is -2.8967 msec, root delay is 36.16 msec root
dispersion is 909.03 msec, peer dispersion is 906.13 msec
```

Note that the synchronization process could take up to five minutes. If you see "Clock is unsynchronized" in the output of "show ntp status", wait a moment and run the command again. If you so desired, you could use "debug ntp events" to watch the communications and see what is happening. Make sure that R1 becomes synchronized before beginning to configure R2 — a device providing NTP services will not do so until the device itself is fully synchronized.

Configuring R1 as an NTP server

Configuring R1 to act as an NTP server is easy enough — it's already done! As soon as R1 becomes synchronized, it will automatically begin answering NTP requests from other hosts. This is why access lists are important; we'll cover those in a later lab.

Configuring R2 as an NTP client

Now we'll configure R2 as an NTP client, just as we did with R1. Let's first show that our clock is unsynchronized, then configure R2 to synchronize with R1:

```
R2# show clock detail *02:16:24.771 UTC Fri Mar 1 2002 No time source
R2# configure terminal Enter configuration commands, one per line. End with
CNTL/Z. R2(config)# ntp server 172.16.12.1
```

After a moment, we should see that R2 is now successfully synchronizing time with R1:

```
R2# show clock detail 04:17:06.271 UTC Sun Nov 23 2008 Time source is NTP R2#
show ntp associations address ref clock st when poll reach delay offset disp
*~172.16.12.1 64.113.32.5 2 17 64 377 35.9 2.41 7.8 * master (synced), #
master (unsynced), + selected, - candidate, ~ configured R2# show ntp status
Clock is synchronized, stratum 3, reference is 172.16.12.1 nominal freq is
250.0000 Hz, actual freq is 250.0001 Hz, precision is 2**18 reference time is
```

```
CCD357B6.5807B0D4 (04:16:54.343 UTC Sun Nov 23 2008) clock offset is 2.4103
msec, root delay is 55.85 msec root dispersion is 261.81 msec, peer
dispersion is 7.80 msec
```

Configuring NTP Authentication

As of now, we have time synchronization working between R1 and an external NTP server and between R1 and R2. Our security policy, however, dictates (hypothetically, of course) that NTP traffic between hosts inside our network must be authenticated. (For additional security, we would use authentication for requests between R1 and the external NTP server as well.)

It is important to note that configuring authentication doesn't require clients to use authentication — it merely enables them to do so. Our NTP servers will still answer requests that are not authenticated, so we'll want to use access lists to control access as well (coming in the next lab). Our first step in turning on authentication is to enable it on R1. Recall from above that we're going to use the key "ThereIsTimeForEverything" (bonus points for the first person to leave a comment below telling me who that quote is from!):

```
R1# configure terminal R1(config)# ntp authenticate R1(config)# ntp
authentication-key 10 md5 ThereIsTimeForEverything R1(config)# ntp trusted-
key 10
Likewise, we must configure the same authentication key on R2 and instruct it
to use that key when "speaking" NTP to R1:
R2# configure terminal Enter configuration commands, one per line. End with
CNTL/Z. R2(config)# ntp authenticate R2(config)# ntp authentication-key 10
md5 ThereIsTimeForEverything R2(config)# ntp trusted-key 10 R2(config)# ntp
server 172.16.12.1 key 10
```

Shortly thereafter, we can check our NTP associations on R2 and we should see that the NTP exchanges between it and R1 are authenticated:

```
R2# show ntp associations detail 172.16.12.1 configured, authenticated,
our_master, sane, valid, stratum 2 ref ID 64.113.32.5, time CCD37909.8B261140
(06:39:05.543 UTC Sun Nov 23 2008) our mode client, peer mode server, our
poll intvl 128, peer poll intvl 128 root delay 20.14 msec, root disp 65.69,
reach 377, sync dist 116.699 delay 7.81 msec, offset 116.2200 msec,
dispersion 37.03 precision 2**18, version 3 org time CCD37929.2A38A6CA
(06:39:37.164 UTC Sun Nov 23 2008) rcv time CCD37929.17DC107A (06:39:37.093
UTC Sun Nov 23 2008) xmt time CCD37929.07835CB1 (06:39:37.029 UTC Sun Nov 23
2008) filtdelay = 63.72 7.81 55.88 48.10 48.02 48.02 36.01 55.95 filtoffset =
103.59 116.22 118.39 92.51 79.64 51.89 70.94 43.87 filterror = 0.02 0.99 1.60
3.56 5.51 7.46 8.44 8.97
```

Did you see it? Let's filter out some of the output and look only at the line we're concerned with:

```
R2# show ntp associations detail | include 172.16.12.1 172.16.12.1
configured, authenticated, our_master, sane, valid, stratum 2
```

Network Device Security

Configure and verify network device security features such as

Device Password security/enable secret vs enable
Five passwords are used to secure your Cisco routers: console, auxiliary, Telnet (VTY), enable password, and enable secret. The enable secret and enable password are used to set the password that's used to secure privileged mode. This will prompt a user for a password when the enable command is used. The other three are used to configure a password when user mode is accessed through the console port, through the auxiliary port, or via Telnet. Let's take a look at each of these now.

Enable Passwords

You set the enable passwords from global configuration mode like this:

```
John(config)#enable ?

last-resort Define enable action if no TACACS servers

respond

password    Assign the privileged level password

secret            Assign the privileged level secret

use-tacacs  Use TACACS to check enable passwords
```

The following points describe the enable password parameters:

- **last-resort** Allows you to still enter the router if you set up authentication through a TACACS server and it's not available. But it isn't used if the TACACS server is working.
- **password** Sets the enable password on older, pre-10.3 systems, and isn't ever used if an enable secret is set.
- **secret** This is the newer, encrypted password that overrides the enable password if it's set.
- **use-tacacs** This tells the router to authenticate through a TACACS server. It's convenient if you have anywhere from a dozen to multitudes of routers because, well, would you like to face the fun task of changing the password on all those routers? If you're sane, no, you

wouldn't. So instead, just go through the TACACS server, and you only have to change the password once

If you try to set the enable secret and enable passwords the same, the router will give you a nice, polite warning to change the second password. If you don't have older legacy routers, don't even bother to use the enable password. User-mode passwords are assigned by using the line command:

```
John(config)#line ?
<0-337> First Line number
aux         Auxiliary line
console     Primary terminal line
tty         Terminal controller
vty         Virtual terminal
x/y         Slot/Port for Modems
x/y/z             Slot/Subslot/Port for Modems
```

Here are the lines to be concerned with:

- **aux** Sets the user-mode password for the auxiliary port. It's usually used for attaching a modem to the router, but it can be used as a console as well. console Sets a console user-mode password.
- **vty** Sets a Telnet password on the router. If this password isn't set, then Telnet can't be used by default.

SSH

Instead of telnet, you can use secure shell, which creates a more secure session than the Telnet application that uses an unencrypted data stream. Secure Shell (SSH) uses encrypted keys to send data so that your username and password are not sent in the clear. Here are the steps to setting up SSH:

1. Set your hostname:

```
Router(config)#hostname john
```

2. Set the domain-name. (Both the hostname and domain-name are required for the encryption keys to be generated.)

```
Todd(config)#ip domain-name johndoe.com
```

3. Generate the encryption keys for securing the session:

```
John(config)#crypto key generate rsa general-keys modulus ?
  <360-2048> size of the key modulus [360-2048]
John(config)#crypto key generate rsa general-keys modulus 1024
The name for the keys will be: john.johndoe.com
```

```
% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-
exportable...[OK]
*June 24 19:25:30.035: %SSH-5-ENABLED: SSH 1.99 has been enabled
```

4. Set the max idle timer for a SSH session:

```
John(config)#ip ssh time-out ?
  <1-120> SSH time-out interval (secs)
John(config)#ip ssh time-out 60
```

5. **Set the max failed attempts for a SSH connection:**

```
John(config)#ip ssh authentication-retries ?
<0-5> Number of authentication retries
John(config)#ip ssh authentication-retries 2
```

6. **Connect to the vty lines of the router:**

```
Todd(config)#line vty 0 1180
```

7. **Last, configure SSH and then Telnet as access protocols.**

```
Todd(config-line)#transport input ssh telnet
```

If you do not use the keyword “telnet” at the end of the command string, then only SSH will work on the router. It will look like this:

```
Todd(config-line)#transport input ssh
```

I am not suggesting you use either way, but just understand that SSH is more secure than Telnet.

Service Password

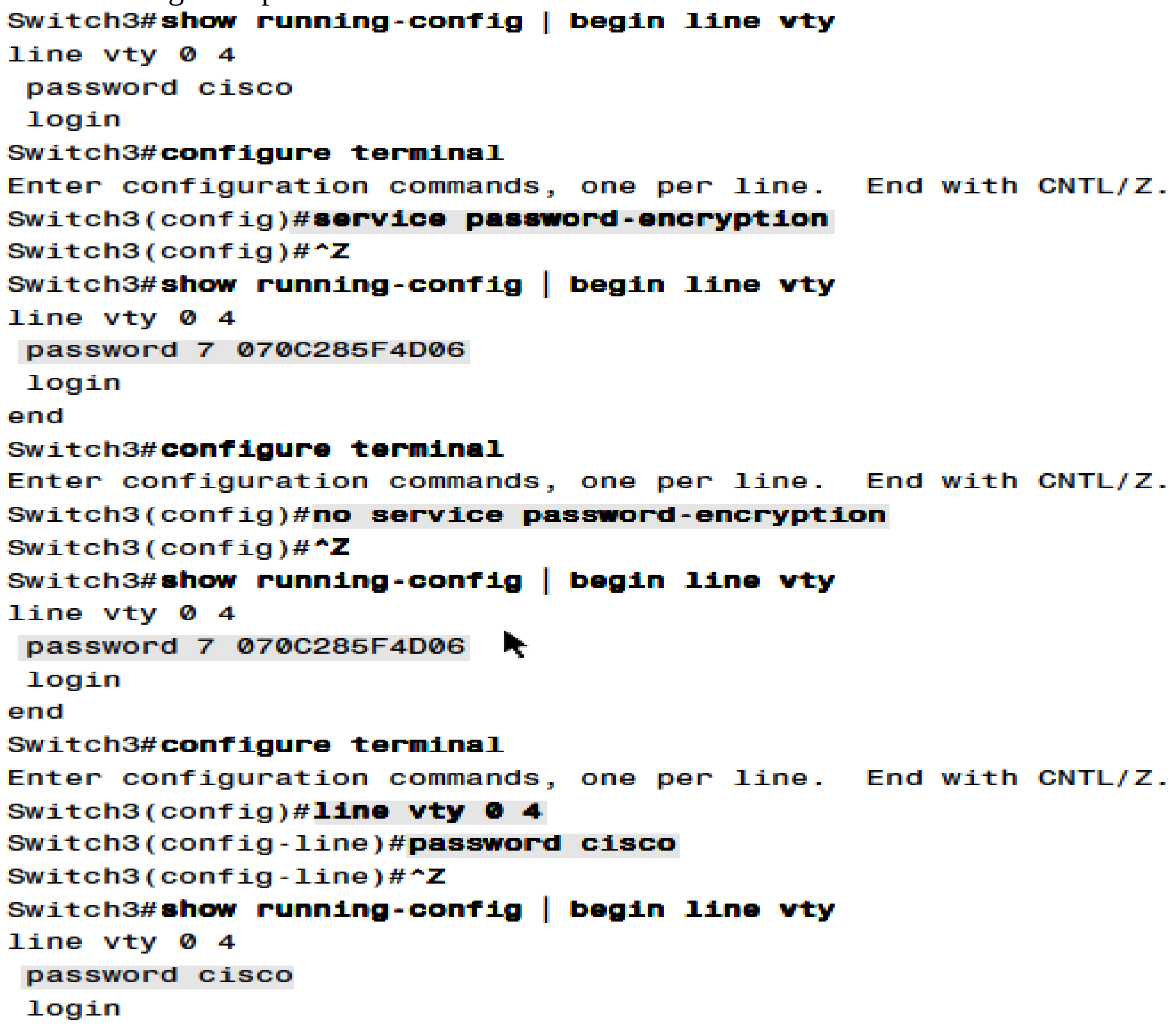
Several of the configuration commands used to configure passwords store the passwords in clear text in the running-config file, at least by default. In particular, the simple passwords configured on the console and vty lines, with the password command, plus the password in the username command, are all stored in clear text by default. (The enable secret command automatically hides the password value.)

To prevent password vulnerability in a printed version of the configuration file, or in a backup copy of the configuration file stored on a server, you can encrypt or encode the passwords using the service password-encryption global configuration command. The presence or absence of the service password-encryption global configuration command dictates whether the passwords are encrypted as follows:

- When the **service password-encryption** command is configured, all existing console, vty, and username command passwords are immediately encrypted.

- If the **service password-encryption** command has already been configured, any future changes to these passwords are encrypted.
- If the **no service password-encryption** command is used later, the passwords remain encrypted, until they are changed—at which point they show up in clear text.

The following example shows this in detail:



```
Switch3#show running-config | begin line vty
line vty 0 4
 password cisco
 login
Switch3#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)#service password-encryption
Switch3(config)#^Z
Switch3#show running-config | begin line vty
line vty 0 4
 password 7 070C285F4D06
 login
end
Switch3#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)#no service password-encryption
Switch3(config)#^Z
Switch3#show running-config | begin line vty
line vty 0 4
 password 7 070C285F4D06
 login
end
Switch3#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)#line vty 0 4
Switch3(config-line)#password cisco
Switch3(config-line)#^Z
Switch3#show running-config | begin line vty
line vty 0 4
 password cisco
 login
```

Configure and Verify Switch Port Security features such as

Port Security

If the network engineer knows what devices should be cabled and connected to particular interfaces on a switch, the engineer can use port security to restrict that interface so that only the expected devices can use it. This reduces exposure to some types of attacks in which the attacker connects a laptop to the wall socket that connects to a switch port that has been configured to use port security. When that inappropriate device attempts to send frames to the switch interface, the switch can issue informational messages, discard frames from that device, or even discard frames from all devices by effectively shutting down the interface.

Port security configuration involves several steps. Basically, you need to make the port an access port, which means that the port is not doing any VLAN trunking. You then need to enable port security and then configure the actual MAC addresses of the devices allowed to use that port. The following list outlines the steps, including the configuration commands used:

- Step 1 Make the switch interface an access interface using the switchport mode access interface subcommand.
- Step 2 Enable port security using the switchport port-security interface subcommand.
- Step 3 (Optional) Specify the maximum number of allowed MAC addresses associated with the interface using the switchport port-security maximum number interface subcommand. (Defaults to one MAC address.)
- Step 4 (Optional) Define the action to take when a frame is received from a MAC address other than the defined addresses using the switchport port-security violation {protect | restrict | shutdown} interface subcommand. (The default action is to shut down the port.)
- Step 5A Specify the MAC address(es) allowed to send frames into this interface using the switchport port-security mac-address mac-address command. Use the command multiple times to define more than one MAC address.
- Step 5B Alternatively, instead of Step 5A, use the “sticky learning” process to dynamically learn and configure the MAC addresses of currently connected hosts by configuring the switchport port-security macaddress sticky interface subcommand.

For example, in Figure 6.1, Server 1 and Server 2 are the only devices that should ever be connected to interfaces FastEthernet 0/1 and 0/2, respectively. When you configure port security on those interfaces, the switch examines the source MAC address of all frames received on those ports, allowing only frames sourced from the configured MAC addresses.

Example 6.1 shows a sample port security configuration matching Figure 9-2, with interface Fa0/1 being configured with a static MAC address, and with interface Fa0/2 using sticky learning.

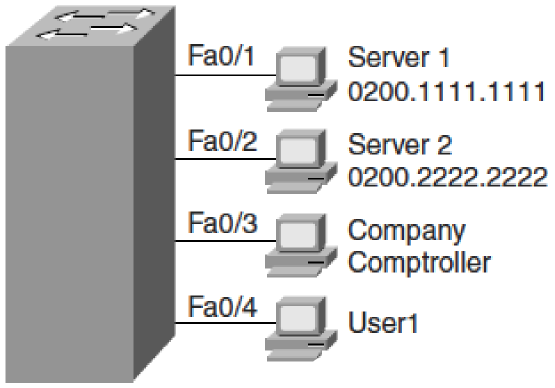


Figure 6.1 Port Security Configuration Example

Example 6.1 Using Port Security to Define Correct MAC Addresses of Particular Interfaces

```
fred#show running-config
(Lines omitted for brevity)
interface FastEthernet0/1
switchport mode access
switchport port-security
switchport port-security mac-address 0200.1111.1111
!
interface FastEthernet0/2
switchport mode access
switchport port-security
switchport port-security mac-address sticky
fred#show port-security interface fastEthernet 0/1
Port Security : Enabled
Port Status : Secure-shutdown
Violation Mode : Shutdown
Aging Time : 0 mins
Aging Type : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses : 1
Total MAC Addresses : 1
Configured MAC Addresses : 1
Sticky MAC Addresses : 0
Last Source Address:Vlan : 0013.197b.5004:1
Security Violation Count : 1
fred#show port-security interface fastEthernet 0/2
Port Security : Enabled
Port Status : Secure-up
Violation Mode : Shutdown
Aging Time : 0 mins
Aging Type : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses : 1
Total MAC Addresses : 1
Configured MAC Addresses : 1
Sticky MAC Addresses : 1
Last Source Address:Vlan : 0200.2222.2222:1
Security Violation Count : 0
fred#show running-config
(Lines omitted for brevity)
interface FastEthernet0/2
switchport mode access
switchport port-security
switchport port-security mac-address sticky
switchport port-security mac-address sticky 0200.2222.2222
```

For FastEthernet 0/1, Server 1’s MAC address is configured with the switchport portsecurity mac-address 0200.1111.1111 command. For port security to work, the 2960 must think that the interface is an access interface, so the switchport mode access command is required. Furthermore, the switchport port-security command is required to enable port security on the interface. Together, these three interface subcommands enable port security, and only MAC address 0200.1111.1111 is allowed to use the interface. This interface uses defaults for the other settings, allowing only one MAC address on the interface, and causing the switch to disable the interface if the switch receives a frame whose source MAC address is not 0200.1111.111.

Interface FastEthernet 0/2 uses a feature called sticky secure MAC addresses. The configuration still includes the switchport mode access and switchport port-security commands for the same reasons as on FastEthernet 0/1. However, the switchport portsecurity mac-address sticky command tells the switch to learn the MAC address from the first frame sent to the switch and then add the MAC address as a secure MAC to the running configuration. In other words, the first MAC address heard “sticks” to the configuration, so the engineer does not have to know the MAC address of the device connected to the interface ahead of time.

The show running-config output at the beginning of Example 9-10 shows the configuration for Fa0/2, before any sticky learning occurred. The end of the example shows the configuration after an address was sticky-learned, including the switchport port-security mac-address sticky 0200.2222.2222 interface subcommand, which the switch added to the configuration. If you wanted to save the configuration so that only 0200.2222.2222 is used on that interface from now on, you would simply need to use the copy running-config startup-config command to save the configuration.

As it turns out, a security violation has occurred on FastEthernet 0/1 in Example 9-10, but no violations have occurred on FastEthernet 0/2. The show port-security interface

fastethernet 0/1 command shows that the interface is in a secure-shutdown state, which means that the interface has been disabled due to port security. The device connected to interface FastEthernet 0/1 did not use MAC address 0200.1111.1111, so the switch received a frame in Fa0/1 with a different source MAC, causing a violation.

The switch can be configured to use one of three actions when a violation occurs. All three configuration options cause the switch to discard the offending frame, but some of the configuration options include additional actions. The actions include the sending of syslog messages to the console and SNMP trap message to the network management station, as well as whether the switch should shut down (err-disable) the interface. The shutdown option actually puts the interface in an error disabled (err-disabled) state, making it unusable. An interface in err-disabled state requires that someone manually shutdown the interface and then use the no shutdown command to recover the interface.

Configure and verify ACLs to filter network traffic

Configure and apply ACLs based on network filtering requirements (including CLI/SDM)
For the CCNA exam prep, we will look at two types of access lists; standard IP access lists, extended IP access lists, and named access lists, which is another way of configuring standard and extended access lists. We will also look at a technique for specifying ranges of addressing called wildcard masking that can be used with all three types of access list. For now, let's get started on standard access lists!

Standard IP Access Lists
Standard IP access lists filter network traffic by examining the source IP address in a packet. You create a standard IP access list by using the access-list numbers 1–99 or 1300–1999 (expanded range). Access-list types are generally differentiated using a number. Based on the number used when the access list is created, the router knows which type of syntax to expect as the list is entered.
By using numbers 1–99 or 1300–1999, you're telling the router that you want to create a standard IP access list, so the router will expect syntax specifying only the source IP address in the test lines.

The following is an example of the many access-list number ranges that you can use to filter traffic on your network (the protocols for which you can specify access lists depend on your IOS version):

```
Corp(config)#
access-list ?
<1-99>             IP standard access list
<100-199>          IP extended access list
<1100-1199>        Extended 48-bit MAC address access list
<1300-1999>        IP standard access list (expanded range)
<200-299>          Protocol type-code access list
<2000-2699>        IP extended access list (expanded range)
<700-799>          48-bit MAC address access list
compiled           Enable IP access-list compilation
dynamic-extended   Extend the dynamic ACL absolute timer
rate-limit         Simple rate-limit specific access list
```

Let's take a look at the syntax used when creating a standard access list:

```
Corp(config)# access-list 10 ?
deny      Specify packets to reject
permit    Specify packets to forward
remark    Access list entry comment
```

As I said, by using the access-list numbers 1–99 or 1300–1999, you're telling the router that you want to create a standard IP access list. After you choose the access-list number, you need to decide whether you're creating a Permit or deny statement.
For this example, you will create a deny statement:

```
Corp(config)#
```

```
access-list 10 deny ?
Hostname or A.B.C.D Address to match
Any       Any source host
host      A single host address
```

The next step requires a more detailed explanation. There are three options available. You can use the any parameter to permit or deny any host or network, you can use an IP address to specify either a single host or a range of them, or you can use the host command to specify a specific host only. The any command is pretty obvious—any source address matches the statement, so every packet compared against this line will match. The host command is relatively simple. Here's an example using it:

```
Corp(config)#
access-list 10 deny host ?
Hostname or A.B.C.D Host address
Corp(config)#
access-list 10 deny host 172.16.30.2
```

This tells the list to deny any packets from host 172.16.30.2. The default parameter is
Host. In other words, if you type access-list 10 deny 172.16.30.2 , the router assumes you that mean host 172.16.30.2.

But there's another way to specify either a particular host or a range of hosts—you can use wildcard masking. In fact, to specify any range of hosts, you have to use wildcard masking in the access list.

Wildcard Masking

Wildcards are used with access lists to specify an individual host, a network, or a certain range of a network or networks. To understand a Wildcard, you need to understand what a Block size is; it's used to specify a range of addresses. Some of the different block sizes available are 64, 32, 16, 8, and 4.

When you need to specify a range of addresses, you choose the next-largest block size for your needs. For example, if you need to specify 34 networks, you need a block size of 64. If you want to specify 18 hosts, you need a block size of 32. If you only specify 2 networks, then a block size of 4 would work.

Wildcards are used with the host or network address to tell the router a range of available addresses to filter. To specify a host, the address would look like this:
172.16.30.5 0.0.0.0

The four zeros represent each octet of the address. Whenever a zero is present, it means that octet in the address must match exactly. To specify that an octet can be any value, the value of 255 is used. As an example, here's how a /24 subnet is specified with a wildcard:

172.16.30.0 0.0.0.255

This tells the router to match up the first three octets exactly, but the fourth octet can be any value.

Now, that was the easy part. What if you want to specify only a small range of subnets? This is where the block sizes come in. You have to specify the range of values in a block size. In other words, you can't choose to specify 20 networks. You can only specify the exact amount as the block size value. For example, the range would have to be either 16 or 32, but not 20.

Let's say that you want to block access to part of the network that is in the range from 172.16.8.0 through 172.16.15.0. That is a block size of 8. Your network number would be 172.16.8.0, and the wildcard would be 0.0.7.255. Whoa! What is that? The 7.255 is what the router uses to determine the block size. The network and wildcard tell the router to start at 172.16.8.0 and go up a block size of eight addresses to network 172.16.15.0.

Seriously—it really is easier than it looks—really! I could certainly go through the binary math for you, but no one needs that. Actually, all you have to do is remember that the wildcard is always one number less than the block size. So, in our example, the wildcard would be 7 since our block size is 8. If you used a block size of 16, the wildcard would be 15. Easy, huh?

But just in case, we'll go through some examples to help you nail it. The following example tells the router to match the first three octets exactly but that the fourth octet can be anything:

```
Corp(config)#
access-list 10 deny 172.16.10.0 0.0.0.255
```

The next example tells the router to match the first two octets and that the last two octets can be any value:

```
Corp(config)#
access-list 10 deny 172.16.0.0
0.0.255.255
```

Try to figure out this next line:

```
Corp(config)#
access-list 10 deny 172.16.16.0 0.0.3.255
```

This configuration tells the router to start at network 172.16.16.0 and use a block size of 4. The range would then be 172.16.16.0 through 172.16.19.0.

The following example shows an access list starting at 172.16.16.0 and going up a block size of 8 to 172.16.23.0:

```
Corp(config)#
access-list 10 deny 172.16.16.0 0.0.7.255
```

The next example starts at network 172.16.32.0 and goes up a block size of 16 to 172.16.47.0:

```
Corp(config)#
access-list 10 deny 172.16.32.0 0.0.15.255
```

The next example starts at network 172.16.64.0 and goes up a block size of 64 to 172.16.127.0:

```
Corp(config)#
access-list 10 deny 172.16.64.0 0.0.63.255
```

The last example starts at network 192.168.160.0 and goes up a block size of 32 to 192.168.191.255:

```
Corp(config)#
access-list 10 deny 192.168.160.0 0.0.31.255
```

Here are two more things to keep in mind when working with block sizes and wildcards:

- Each block size must start at 0 or a multiple of the block size. For example, you can't say that you want a block size of 8 and then start at 12. You must use 0–7, 8–15, 16–23, and so on. For a block size of 32, the ranges are 0–31, 32–63, 64–95, and so on.
- The command any is the same thing as writing out the wildcard 0.0.0.0255.255.255.255.

Configure and verify an ACLs to limit telnet and SSH access to the router

Configure and apply ACLs to limit telnet and SSH access to the router using (including: SDM/CLI)
You'll probably have a difficult time trying to stop users from telnetting to a large router because any active interface on a router is fair game for VTY access. You could try to create an extended IP access list that limits Telnet access to every IP address on the router. But if you did that, you'd have to apply it inbound on every interface, and that really wouldn't scale well to a large router with dozens, even hundreds, of interfaces, would it? Here's a much better solution: Use a standard IP access list to control access to the VTY lines themselves.

Why does this work? Because when you apply an access list to the VTY lines, you don't need to specify the Telnet protocol since access to the VTY implies terminal access. You also don't need to specify a destination address, since it really doesn't matter which interface address the user used as a target for the Telnet session. You really only need to control where the user is coming from—their source IP address. To perform this function, follow these steps:

1. Create a standard IP access list that permits only the host or hosts you want to be able to telnet into the routers.
2. Apply the access list to the VTY line with the access-class command.

Here is an example of allowing only host 172.16.10.3 to telnet into a router:

```
Lab_A(config)#access-list 50 permit 172.16.10.3
Lab_A(config)#line vty 0 4
Lab_A(config-line)#access-class 50 in
```

Because of the implied deny any at the end of the list, the access list stops any host from telnetting into the router except the host 172.16.10.3, regardless of which individual IP address on the router is used as a target.

Troubleshooting

Troubleshoot and correct common problems associated with IP addressing and host configurations

Troubleshooting host routing problems should begin with the same two-step routing logic used by a host. The first question to ask is whether the host can ping other hosts inside the same subnet. If a ping of a same-subnet host fails, the root cause typically falls into one of two categories:

- The two hosts have incorrect IP address and mask configurations, typically so that at least one of the two hosts thinks it is in a different subnet.
- The two hosts have correct IP address and mask configurations, but the underlying Ethernet has a problem.

For the exam, start by looking at the host's addresses and masks, and determine the subnet number and range of addresses for each. If the subnets are the same, then move on to Layer 1 and 2 Ethernet troubleshooting. If the host can ping other hosts in the same subnet, the next step is to confirm if the host can ping IP addresses in other subnets, thereby testing the second branch of a host's routing logic. Two different pings can be helpful at this step:

- Ping the default gateway IP address to confirm that the host can send packets over the LAN to and from the default gateway.
- Ping a different IP address on the default gateway/router, but not the IP address on the same LAN.

You could first issue the ping 172.16.1.253 command to confirm whether PC1 can send packets to and from its presumed default gateway. If the ping was successful, PC1 could use a ping 172.16.2.253 command, which forces PC1 to use its default gateway setting, because PC1 thinks that 172.16.2.253 is in a different subnet.

So, when a host can ping other hosts in the same subnet, but not hosts in other subnets, the root cause typically ends up being one of a few items, as follows:

- There is some mismatch between the host's default gateway configuration and the router acting as default gateway. The problems include mismatched masks between the host and the router, which impacts the perceived range of addresses in the subnet, or the host simply referring to the wrong router IP address.
- If the default gateway settings are all correct, but the ping of the default gateway IP address fails, there is probably some Layer 1 or Layer 2 problem on the LAN.
- If the default gateway settings are all correct and the ping of the default gateway works, but the ping of one of the other router interface IP addresses fails (like the ping 172.16.2.253 command based on Figure 21-1), then the router's other interface may have failed.

For the exam, be ready to find the IP address and masks, and apply the math from Part III to quickly determine where these types of problems exist.

Troubleshoot and Resolve vLAN problems

It may come as a surprise to you, but configuring VLANs is actually pretty easy. Figuring out which users you want in each VLAN is not; it's extremely time-consuming. But once you've decided on the number of VLANs you want to create and established which users you want to belong to each one, it's time to bring your first VLAN into the world.

To configure VLANs on a Cisco Catalyst switch, use the global config vlan command. In the following example, I'm going to demonstrate how to configure VLANs on the S1 switch by creating three VLANs for three different departments—again, remember that VLAN 1 is the native and administrative VLAN by default:

```
S1#config t
S1(config)#vlan ?
WORD ISL VLAN IDs 1-4094
internal internal VLAN
S1(config)#vlan 2
S1(config-vlan)#name Sales
S1(config-vlan)#vlan 3
S1(config-vlan)#name Marketing
S1(config-vlan)#vlan 4
S1(config-vlan)#name Accounting
S1(config-vlan)#^Z
S1#
```

From the preceding, you can see that you can create VLANs from 2 to 4094. This is only mostly true. As I said, VLANs can really only be created up to 1005, and you can't use, change, rename, or delete VLANs 1 and 1002 through 1005 because they're reserved. The VLAN numbers above that are called extended VLANs and won't be saved in the database unless your switch is set to VTP transparent mode.

You won't see these VLAN numbers used too often in production. Here's an example of setting my S1 switch to VLAN 4000 when my switch is set to VTP server mode (the default VTP mode):

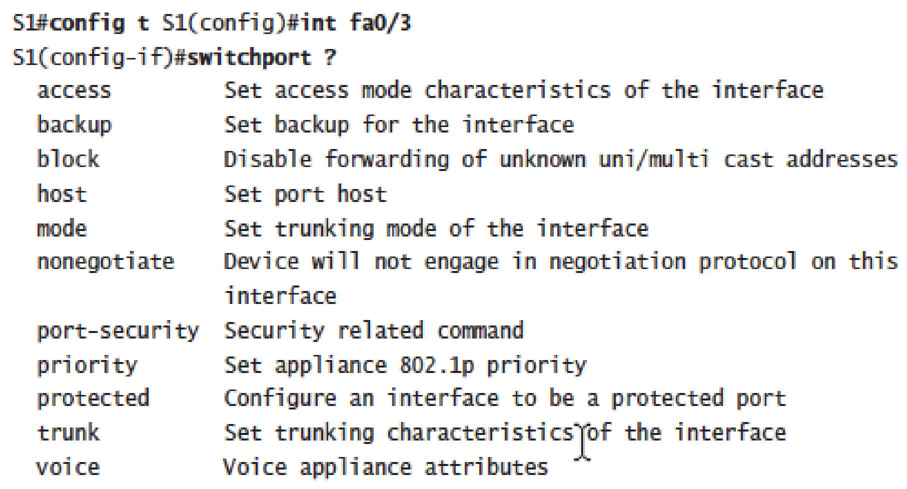
```
S1#config t
S1(config)#vlan 4000
S1(config-vlan)#^Z
% Failed to create VLANs 4000
Extended VLAN(s) not allowed in current VTP mode.
%Failed to commit extended VLAN(s) changes.
```

After you create the VLANs that you want, you can use the show vlan command to check them out. But notice that, by default, all ports on the switch are in VLAN 1. To change the VLAN associated with a port, you need to go to each interface and tell it which VLAN to be a part of.

Identify that vLANs are configured/port membership correct/IP address configured

You configure a port to belong to a VLAN by assigning a membership mode that specifies the kind of traffic the port carries, plus the number of VLANs to which it can belong. You can configure each port on a switch to be in a specific VLAN (access port) by using the interface switchport command. You can also configure multiple ports at the same time with the interface range command.

I'm only going to cover the static flavor. In the following example, I'll configure interface fa0/3 to VLAN 3. This is the connection from the S1 switch to the HostA device:



```
S1#config t S1(config)#int fa0/3
S1(config-if)#switchport ?
  access         Set access mode characteristics of the interface
  backup         Set backup for the interface
  block          Disable forwarding of unknown uni/multi cast addresses
  host           Set port host
  mode           Set trunking mode of the interface
  nonegotiate    Device will not engage in negotiation protocol on this
                 interface
  port-security  Security related command
  priority       Set appliance 802.1p priority
  protected      Configure an interface to be a protected port
  trunk          Set trunking characteristics of the interface
  voice          Voice appliance attributes
```

Well now, what do we have here? There's some new stuff showing up in the preceding output. Let's start with setting an access port on S1, which is probably the most widely used type of port on production switches that has VLANs configured:

```
S1(config-if)#switchport mode ?
 access     Set trunking mode to ACCESS unconditionally
 dynamic    Set trunking mode to dynamically negotiate access or
trunk mode
 trunk      Set trunking mode to TRUNK unconditionally
```

```
S1(config-if)#switchport mode access
S1(config-if)#switchport access vlan 3
```

By starting with the switchport mode access command, you're telling the switch that this is a layer 2 port. You can then assign a VLAN to the port with the switchport access command. Remember, you can choose many ports to configure at the same time if you use the interface range command. The dynamic and trunk commands are used for trunk ports exclusively.

Troubleshoot and Resolve trunking problems on Cisco switches

Configure, verify, and troubleshoot trunking on Cisco switches
The 2960 switch only runs the IEEE 802.1Q encapsulation method. To configure trunking on a Fast Ethernet port, use the interface command trunk [parameter]. It's a tad different on the 3560 switch.

The following switch output shows the trunk configuration on interface fa0/8 as set to trunk on:

```
S1#config t
S1(config)#int fa0/8
S1(config-if)#switchport mode trunk
The following list describes the different options available when configuring
a
switch interface:
```

switch-port mode access This puts the interface (access port) into permanent non-trunking mode and negotiates to convert the link into a non-trunk link. The interface becomes a non-trunk interface regardless of whether the neighboring interface is a trunk interface. The port would be a dedicated layer 2 port.

Switch-port mode dynamic auto This mode makes the interface able to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to trunk or desirable mode. This is now the default switch-port mode for all Ethernet interfaces on all new Cisco switches.
Switch-port mode dynamic desirable This one makes the interface actively attempt to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to trunk, desirable, or auto mode. I used to see this mode as the default on some older switches, but not any longer. The default is dynamic auto now.
switchport mode trunk Puts the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link. The interface becomes a trunk interface even if the neighboring interface isn't a trunk interface.
switchport non-egotiate Prevents the interface from generating DTP frames. You can use this command only when the interface switchport mode is access or trunk. You must manually configure the neighboring interface as a trunk interface to establish a trunk link.

Defining the Allowed VLANs on a Trunk

As I've mentioned, trunk ports send and receive information from all VLANs by default, and if a frame is untagged, it's sent to the management VLAN. This applies to the extended range VLANs as well.

But we can remove VLANs from the allowed list to prevent traffic from certain VLANs from traversing a trunked link. Here's how you'd do that:

```
S1#config t
S1(config)#int f0/1
S1(config-if)#switchport trunk allowed vlan ?

WORD     VLAN IDs of the allowed VLANs when this port is in
Trunking   mode add add VLANs to the current list
all              all VLANs
except        all VLANs except the following
none            no VLANs
remove        remove VLANs from the current list

S1(config-if)#switchport trunk allowed vlan remove ?

WORD VLAN IDs of disallowed VLANS when this port is in trunking mode

S1(config-if)#switchport trunk allowed vlan remove 4
```

The preceding command stopped the trunk link configured on S1 port f0/1, causing it to drop all traffic sent and received for VLAN 4. You can try to remove VLAN 1 on a trunk link, but it will still send and receive management like CDP, PAgP, LACP, DTP, and VTP, so what's the point?

To remove a range of VLANs, just use a hyphen:

S1(config-if)#switchport trunk allowed vlan remove 4-8
If by chance someone has removed some VLANs from a trunk link and you want to set the trunk back to default, just use this command:

```
S1(config-if)#switchport trunk allowed vlan all
```

Or this command to accomplish the same thing:

```
S1(config-if)#no switchport trunk allowed vlan
```

Next, I want to show you how to configure pruning for VLANs before we start routing between VLANs.

Changing or Modifying the Trunk Native VLAN

You really don't want to change the trunk port native VLAN from VLAN 1, but you can, and some people do it for security reasons. To change the native VLAN, use the following command:

```
S1#config t
S1(config)#int f0/1
S1(config-if)#switchport trunk ?
allowed Set allowed VLAN characteristics when interface is
in trunking mode
native Set trunking native characteristics when interface
is in trunking mode
pruning Set pruning VLAN characteristics when interface is
in trunking mode
S1(config-if)#switchport trunk native ?
vlan Set native VLAN when interface is in trunking mode
S1(config-if)#switchport trunk native vlan ?
<1-4094> VLAN ID of the native VLAN when this port is in
trunking mode
S1(config-if)#switchport trunk native vlan 40
S1(config-if)#^Z
So, we've changed our native VLAN on our trunk link to 40, and by using the
show
running-config command, we can see the configuration under the trunk link:
!
interface FastEthernet0/1
switchport trunk native vlan 40
switchport trunk allowed vlan 1-3,9-4094
switchport trunk pruning vlan 3,4
!
```

Hold on there partner! You didn't think it would be this easy and would just start working, did you? Sure you didn't. Here's the rub: If all switches don't have the same native VLAN configured on the trunk links, then we'll start to receive this error:

```
19:23:29: %CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch
discovered on FastEthernet0/1 (40), with Core FastEthernet0/7 (1).
19:24:29: %CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch
discovered on FastEthernet0/1 (40), with Core FastEthernet0/7 (1).
```

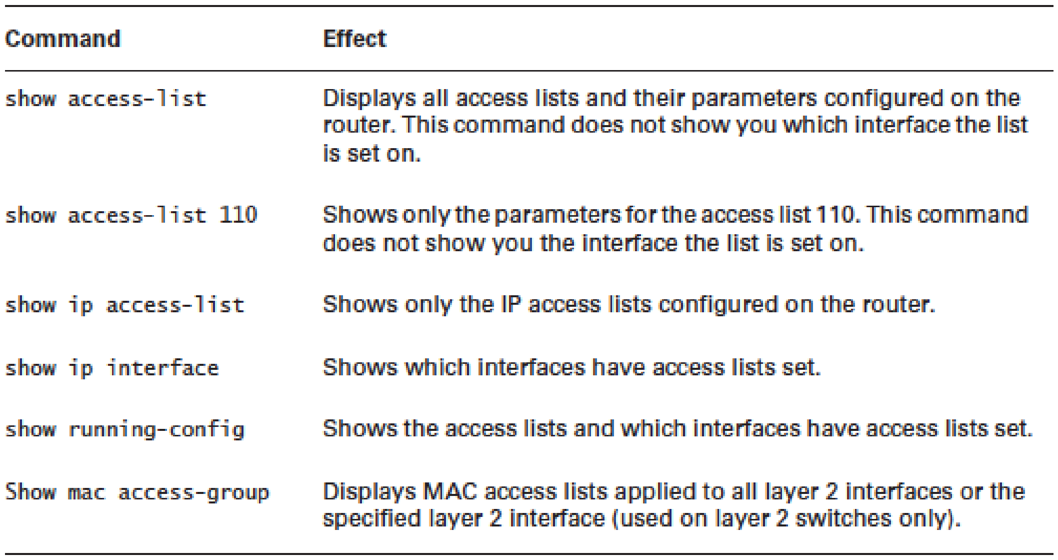
Actually, this is a good, noncryptic error, so either we go to the other end of our trunk link(s) and change the native VLAN or we set the native VLAN back to the default.
Here's how we'd do that:

```
S1(config-if)#no switchport trunk native vlan
```

Troubleshoot and Resolve ACL issues

When working on a problem, one item to eliminate is the possibility of an access list blocking traffic. It is a crucial troubleshooting skill to be able to quickly view both the contents of access lists, and where they are applied.

Again, it's always good to be able to verify a router's configuration. The following table lists the commands that can be used to verify the configuration.



Command	Effect
`show access-list`	Displays all access lists and their parameters configured on the router. This command does not show you which interface the list is set on.
`show access-list 110`	Shows only the parameters for the access list 110. This command does not show you the interface the list is set on.
`show ip access-list`	Shows only the IP access lists configured on the router.
`show ip interface`	Shows which interfaces have access lists set.
`show running-config`	Shows the access lists and which interfaces have access lists set.
`Show mac access-group`	Displays MAC access lists applied to all layer 2 interfaces or the specified layer 2 interface (used on layer 2 switches only).

We've already used the show running-config command to verify that a named access list was in the router as well as a MAC access list on a layer 2 switch. So, now let's take a look at the output from some of the other commands.

The show access-list command will list all access lists on the router, whether they're applied to an interface or not:

```
Lab_A#show access-list
Standard IP access list 10
      deny 172.16.40.0, wildcard bits 0.0.0.255
      permit any
Standard IP access list BlockSales
      deny 172.16.40.0, wildcard bits 0.0.0.255
      permit any
Extended IP access list 110
      deny tcp any host 172.16.30.5 eq ftp
      deny tcp any host 172.16.30.5 eq telnet
      permit ip any any
Lab_A#
```

First, notice that both access list 10 and our named access list appear on this list. Second, notice that even though I entered actual numbers for TCP ports in access list 110, the show command gives us the protocol names rather than TCP ports for readability. (Hey, not everyone has them all memorized!)

Here's the output of the show ip interface command:

```
Lab_A#show ip interface e1
Ethernet1 is up, line protocol is up
Internet address is 172.16.30.1/24
Broadcast address is 255.255.255.255
Address determined by non-volatile memory
MTU is 1500 bytes
Helper address is not set
Directed broadcast forwarding is disabled
Outgoing access list is BlockSales
Inbound access list is not set
Proxy ARP is enabled
Security level is default
Split horizon is enabled
ICMP redirects are always sent
ICMP unreachables are always sent
ICMP mask replies are never sent
IP fast switching is disabled
IP fast switching on the same interface is disabled
IP Null turbo vector
IP multicast fast switching is disabled
IP multicast distributed fast switching is disabled
Router Discovery is disabled
IP output packet accounting is disabled
IP access violation accounting is disabled
TCP/IP header compression is disabled
RTP/IP header compression is disabled
Probe proxy name replies are disabled
Policy routing is disabled
Network address translation is disabled
Web Cache Redirect is disabled
BGP Policy Mapping is disabled
Lab_A#
```

Be sure to notice the bold line indicating that the outgoing list on this interface is BlockSales but the inbound access list isn't set. One more verification command and then we'll move on to using the SDM to configure firewall security.

Troubleshoot and Resolve Layer 1 problems

Troubleshooting with the show controller t1 Command

The show controller t1 command displays the controller status specific to the controller hardware. This information is useful for diagnostic tasks performed by technical support personnel. The Network

Processor Module (NPM) or Multi-Channel Interface Processor (MIP) can query the port adapters to determine their current status.

The show controller t1 EXEC command also provides this information:

- Statistics about the T1 link. If you specify a slot and a port number, statistics for each 15-minute period are displayed.
- Information to troubleshoot physical layer and data link layer problems.
- Local or remote alarm information, if any, on the T1 line.

Most T1 errors are caused by incorrectly configured lines. Ensure that line coding, framing, and clock source are configured in accordance to the recommendations of your Service Provider.

The T1 controller can be in three states:

- Administratively down
- Down
- Up

Administratively Down T1 Controller

The controller is administratively down when it has been manually shut down. Complete these steps in order to restart the controller to correct this error:

Enter enable mode.

For example:

```
maui-nas-03>enable
Password:
maui-nas-03#
Enter global configuration mode.
```

Enter global configuration mode.

For example:

```
maui-nas-03#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
maui-nas-03(config)#
Enter controller configuration mode.
```

Enter controller configuration mode.

For example:

```
maui-nas-03(config)#controller t1 0
maui-nas-03(config-controlle)#
```

Restart the controller.

```
maui-nas-03(config-controlle)#no shutdown
```

Ensure the Line is up

If the T1 controller and line are not up, ensure one of these messages appears in the **show controller t1** EXEC command output:
Receiver has loss of frame. or Receiver has loss of signal.

Loss of Frame

Complete these steps if the receiver has loss of frame:

1. Ensure the framing format configured on the port matches the framing format of the line. Check the framing format of the controller from the running configuration or the **show controller t1** command output.

   Issue the **framing {SF | ESF}** command in controller configuration mode in order to change the framing format. For example:

   ```
   maui-nas-03#configure terminal Enter configuration commands, one
   per line. End with CNTL/Z. maui-nas-03(config)#controller t1 0
   maui-nas-03(config-controlle)#framing esf
   ```

2. Try the other framing format to see if the alarm clears.
3. Issue the **cablelength long** or **cablelength short** command in order to change the line build-out (LBO) setting.

   LBO compensates for the loss in decibels based on the distance from the device to the first repeater in the circuit. A longer distance from the device to the repeater requires that the signal strength on the circuit be boosted to compensate for loss over that distance.

   Issue the **cablelength long** controller configuration command in order to configure transmit and receive levels for a cable length (line build-out) longer than 655 feet for a T1 trunk with a channel service unit (CSU) interface. Issue the**cablelength short** controller configuration command in order to configure transmit attenuation for a cable length (line build-out) of 655 feet or shorter for a T1 trunk with a DSX-1 interface.

Loss of Signal

Complete these steps:

1. Ensure the cable between the interface port and the T1 Service Provider equipment or T1 terminal equipment is connected correctly.

Ensure the cable is hooked up to the correct ports. Correct the cable connections if necessary.

2. Check the cable integrity by looking for breaks or other physical abnormalities in the cable.

   Ensure the pinouts are set correctly. Replace the cable if necessary.

3. Check the cable connectors. A reversal of the transmit and receive pairs or an open receive pair can cause errors.

   The receive pair should be on lines 1 and 2, and the transmit pair should be on lines 4 and 5.

   The pins on a RJ-45/48 jack plug are numbered from 1 through 8. With the metal pins facing toward you, pin 1 is the left-most pin. This figure shows the pin numbering on an RJ-45 jack:

4. If you have completed all of these steps and you still experience problems, use a rollover cable.

Issue the **show controller t1** EXEC command after each step in order to see if the controller exhibits any errors.

Loopback Mode

Ensure the line is in loopback mode from the show controller t1 command output. The line should be in loopback mode only for testing purposes.

Issue the **no loopback** command in controller configuration mode in order to turn off loopback. For example:

```
maui-nas-03(config-controlle)#no loopback
```

LAN Switching Technologies

Identify enhanced switching technologies

Describe enhanced switching technologies (including: VTP, RSTP, VLAN, PVSTP, 802.1q)
The basic goals of VLAN Trunking Protocol (VTP) are to manage all configured VLANs across a switched internetwork and to maintain consistency throughout that network. VTP allows you to add, delete, and rename VLANs—information that is then propagated to all other switches in the VTP domain.

Here’s a list of some of the cool features VTP has to offer:

- Consistent VLAN configuration across all switches in the network
- VLAN trunking over mixed networks, such as Ethernet to ATM LANE or even FDDI
- Accurate tracking and monitoring of VLANs
- Dynamic reporting of added VLANs to all switches in the VTP domain
- Plug and Play VLAN adding

Very nice, but before you can get VTP to manage your VLANs across the network, you have to create a VTP server. All servers that need to share VLAN information must use the same domain name, and a switch can be in only one domain at a time. So, basically, this means that a switch can only share VTP domain information with other switches if they’re configured into the same VTP domain. You can use a VTP domain if you have more than one switch connected in a network, but if you’ve got all your switches in only one VLAN, you just don’t need to use VTP. Do keep in mind that VTP information is sent between switches only via a trunk port.

Switches advertise VTP management domain information as well as a configuration revision number and all known VLANs with any specific parameters. But there’s also something called VTP transparent mode. In it, you can configure switches to forward VTP information through trunk ports but not to accept information updates or update their VTP databases.

If you’ve got sneaky users adding switches to your VTP domain behind your back, you can include passwords, but don’t forget—every switch must be set up with the same password.

And as you can imagine, this little snag can be a real hassle administratively!

Switches detect any added VLANs within a VTP advertisement, and then prepare to send information on their trunk ports with the newly defined VLAN in tow. Updates are sent out as revision numbers that consist of the notification plus 1. Anytime a switch sees a higher revision number, it knows the information it’s getting is more current, so it will overwrite the existing database with the latest information.

You should know these three requirements for VTP to communicate VLAN information between switches:

- The VTP management domain name of both switches must be set the same.
- One of the switches has to be configured as a VTP server.
- No router is necessary.

Now that you’ve got that down, we’re going to delve deeper in the world of VTP with VTP modes and VTP pruning.

RSTP

Rapid Spanning-Tree Protocol (RSTP) 802.1w
How would you like to have a good STP configuration running on your switched network (regardless of the brand of switches) and have all the features built in and enabled on every switch? Absolutely—yes! Well then, welcome to the world of Rapid Spanning- Tree Protocol (RSTP).

Cisco created PortFast, UplinkFast, and BackboneFast to "fix" the holes and liabilities the IEEE 802.1d standard presented. The drawbacks to these enhancements are only that they are Cisco proprietary and need additional configuration. But the new 802.1w standard (RSTP) addresses all these "issues" in one tight package—just turn on RSTP and you're good to go.

Importantly, you must make sure that all the switches in your network are running the 802.1w protocol for 802.1w to work properly! It might come as a surprise, but RSTP actually can interoperate with legacy STP protocols. Just know that the inherently fast convergence ability of 802.1w is lost when it interacts with legacy bridges.

PVSTP

PVST
Understand that Cisco switches run what is called Per-VLAN Spanning-Tree (PVST), which basically means that each VLAN runs its own instance of the STP protocol. If we typed show spanning-tree, we'd receive information for each VLAN, starting with VLAN 1. So, say we've got multiple VLANs, and we want to see what's up with VLAN 2—we'd use the command show spanning-tree vlan 2.

Etherchannels

IEEE 802.1Q
Created by the IEEE as a standard method of frame tagging, IEEE 802.1Q actually inserts a field into the frame to identify the VLAN. If you're trunking between a Cisco switched link and a different brand of switch, you've got to use 802.1Q for the trunk to work.
It works like this: You first designate each port that is going to be a trunk with 802.1Q encapsulation. The ports must be assigned a specific VLAN ID, which makes them the native VLAN, in order for them to communicate. The ports that populate the same trunk create a group with this native VLAN, and each port gets tagged with an identification number reflecting that, again, the default is VLAN 1. The native VLAN allows the trunks to carry information that was received without any VLAN identification or frame tag.

The 2960s support only the IEEE 802.1Q trunking protocol, but the 3560s will support both the ISL and IEEE methods.

The basic purpose of ISL and 802.1Q frame-tagging methods is to provide inter-switch
VLAN communication. Also, remember that any ISL or 802.1Q frame tagging is removed if a frame is forwarded out an access link—tagging is used across trunk links only!

Configure and verify PVSTP operation

PVST+ Operation

In a Cisco PVST+ environment, you can tune the spanning-tree parameters so that half of the VLANs forward on each uplink trunk. To easily achieve this, you configure one switch to be elected the root bridge for half of the total number of VLANs in the network and a second switch to be elected the root bridge for the other half of the VLANs. Providing different STP root switches per VLAN creates a more redundant network.

Spanning-tree operation requires that each switch has a unique BID. In the original 802.1D standard, the BID was composed of the bridge priority and the MAC address of the switch, and all VLANs were represented by a CST. Because PVST+ requires that a separate instance of spanning tree runs for each VLAN, the BID field is required to carry VID information. This is accomplished by reusing a portion of the Priority field as the extended system ID to carry a VID. Figure 8.1 shows how modifying the bridge priority offers this support.

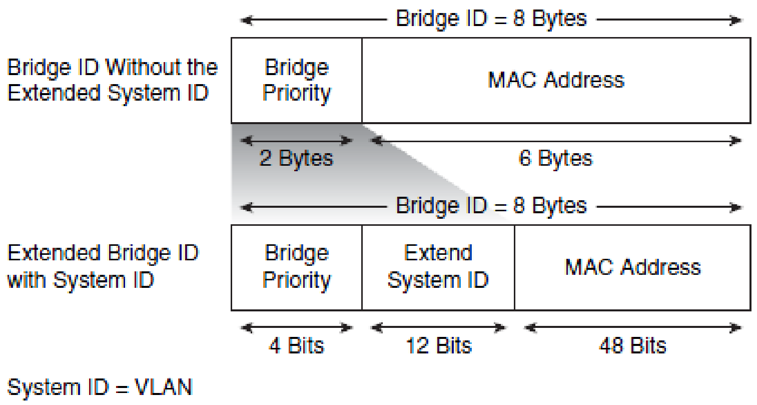


Figure 8.1 PVST+ VLAN ID

To accommodate the extended system ID, the original 802.1D 16-bit bridge priority field is split into two fields, resulting in these components in the BID:

- **Bridge priority**: A 4-bit field still used to carry bridge priority. Because of the limited bit count, the priority is conveyed in discreet values in increments of 4096 rather than discreet values in increments of 1, as they would be if the full 16-bit field was available. The default priority, in accordance with IEEE 802.1D, is 32,768, which is the midrange value.
- **Extended system ID**: A 12-bit field carrying, in this case, the VID for PVST+.
- **MAC address**: A 6-byte field with the MAC address of a single switch.

By virtue of the MAC address, a BID is always unique. When the priority and extended system ID are prepended to the switch MAC address, each VLAN on the switch can be represented by a unique BID.

If no priority has been configured, every switch will have the same default priority, and the election of the root for each VLAN will be based on the MAC address. This method is a random means of selecting the ideal root bridge; for this reason, it is advisable to assign a lower priority to the switch that should serve as the root bridge. The root bridge should be located in the center of your network traffic flow.

Describe root bridge election

STP performs three steps to provide a loop-free logical network topology:

1. Elects one root bridge: STP has a process to elect a root bridge. Only one bridge can act as the root bridge in a given network. On the root bridge, all ports are designated ports. Designated ports are in the forwarding state and are designated to forward traffic for a given segment. When in the forwarding state, a port can send and receive traffic.

In Figure 8.2, switch X is elected as the root bridge.

2. Selects the root port on the non-root bridge: STP establishes one root port on each non-root bridge. The root port is the lowest-cost path from the non-root bridge to the root bridge. Root ports are in the forwarding state. Spanning-tree path cost is an accumulated cost calculated on the bandwidth. In Figure 8.2, the lowest-cost path to the root bridge from switch Y is through the 100BASE-T Fast-Ethernet link.
3. Selects the designated port on each segment: On each segment, STP establishes one designated port. The designated port is selected on the bridge that has the lowest-cost path to the root bridge. Designated ports are in the forwarding state, forwarding traffic for the segment. In Figure 8.2, the designated port for both segments is on the root bridge because the root bridge is directly connected to both segments. The 10BASE-T Ethernet port on switch Y is a non-designated port because there is only one designated port per segment. Non-designated ports are normally in the blocking state to logically break the loop topology. When a port is in the blocking state, it is not forwarding data traffic but can still receive traffic.

Switches and bridges running the Spanning Tree Algorithm exchange configuration messages with other switches and bridges at regular intervals (every 2 seconds by default).

Switches and bridges exchange these messages using a multicast frame called the BPDU.

One of the pieces of information included in the BPDU is the bridge ID (BID). STP calls for each switch or bridge to be assigned a unique BID. Typically, the BID is composed of a priority value (2 bytes) and the bridge MAC address (6 bytes). The default priority, in accordance with IEEE 802.1D, is 32,768 (1000 0000 0000 0000 in binary, or 0x8000 in hex format), which is the midrange value. The root bridge is the bridge with the lowest BID.

Example: Selecting the Root Bridge
In Figure 8.1, both switches use the same default priority. The switch with the lowest

MAC address is the root bridge. In the example, switch X is the root bridge, with a BID of 0x8000 (0c00.1111.1111).

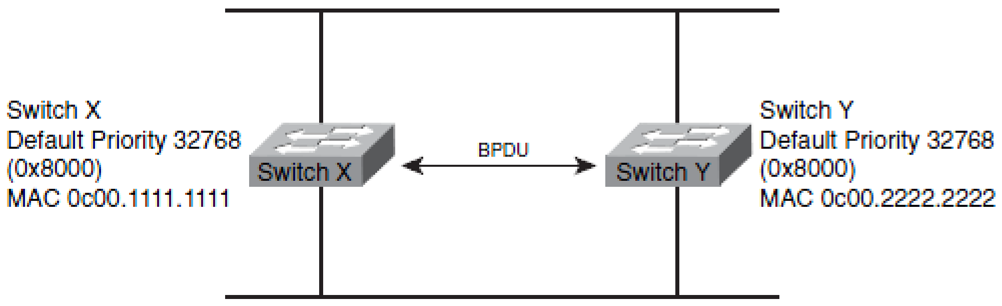


Figure 8.2 Root Bridge Selections

There are five STP port states:

- Blocking
- Listening
- Learning
- Forwarding
- Disabled

When STP is enabled, every bridge in the network goes through the blocking state and the transitory states of listening and learning at power-up. If properly configured, the ports then stabilize to the forwarding or blocking state. Forwarding ports provide the lowest-cost path to the root bridge. During a topology change, a port temporarily implements the listening and learning states.

The disabled state is not strictly part of STP; a network administrator can manually disable a port, or a security or an error condition may disable it. An example of a port that is disabled would be a port that is shut down. Figure 8.3 shows the flow of spanning-tree port states.

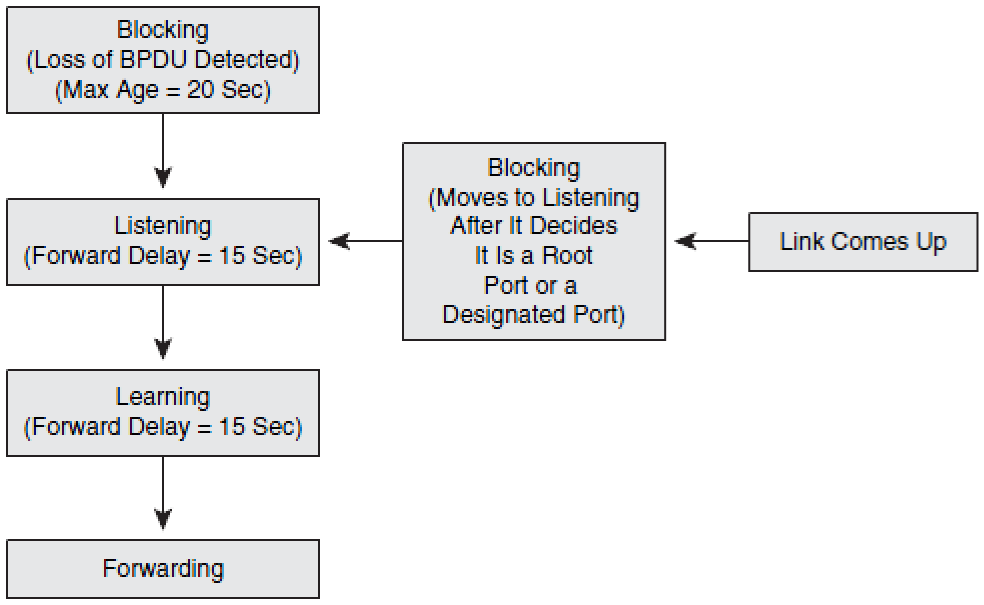


Figure 8.3 Spanning-Tree Port States

All bridge ports initially start in the blocking state, from which they listen for BPDUs.

When the bridge first boots, it functions as if it were the Root Bridge and transitions to the listening state. An absence of BPDUs for a certain period is called the maximum age (max_age), which has a default of 20 seconds. If a port is in the blocking state and does not receive a new BPDU within the max_age, the bridge transitions from the blocking state to the listening state. When a port is in the transitional listening state, it can send and receive BPDUs to determine the active topology. At this point, the switch is not passing user data.

During the listening state, the bridge performs these three steps:

1. Selects the root bridge
2. Selects the root ports on the non-root bridges
3. Selects the designated ports on each segment

The time that it takes for a port to transition from the listening state to the learning state or from the learning state to the forwarding state is called the forward delay. The forward delay has a default value of 15 seconds.

The learning state reduces the amount of flooding required when data forwarding begins. If a port is still a designated or root port at the end of the learning state, the port transitions to the forwarding state. In the forwarding state, a port is capable of sending and receiving user data. Ports that are not the designated or root ports transition back to the blocking state.

A port normally transitions from the blocking state to the forwarding state in 30 to 50 seconds. You can tune the spanning-tree timers to adjust the timing, but these timers are meant to be set to the default value. The default values are put in place to give the network enough time to gather all the correct information about the network topology.

Spanning-tree PortFast causes an interface that is configured as a Layer 2 access port to transition immediately from the blocking state to the forwarding state, bypassing the listening and learning states. You can use PortFast on Layer 2 access ports that are connected to a single workstation or server to allow those devices to connect to the network immediately rather than wait for spanning tree to converge. Figure 8.4 shows access ports connected with PortFast enabled.

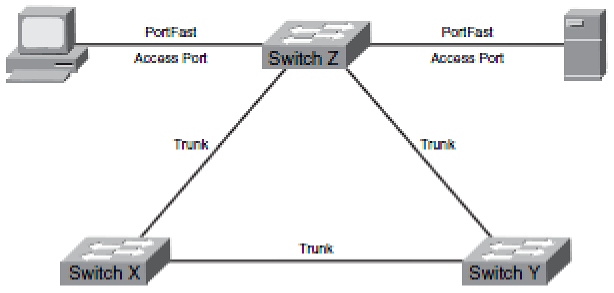


Figure 8.4 PortFast

If an interface that is configured with PortFast receives a BPDU, then spanning tree can transition the port to the blocking state. Using a feature called BPDU guard, the port can be disabled completely when it receives a BPDU to prevent any potential loops caused by PortFast.

Table 8.1 lists the commands used to implement and verify PortFast on an interface.

Table 8.2 PortFast Commands

Command	Description
Switch(config-if)#**spanning-tree portfast**	Enables PortFast on a Layer 2 access port and forces it to enter the forwarding state immediately.
Switch(config-if)#**spanning-tree portfast bpdu-guard**	Enables PortFast with BPDU guard. This disables the switch port if a BPDU is ever received, preventing any possibility of a loop.
Switch(config-if)#**no spanning-tree portfast**	Disables PortFast on a Layer 2 access port. PortFast is disabled by default.
Switch(config)#**spanning-tree portfast default**	Globally enables the PortFast feature on all nontrunking ports. When the PortFast feature is enabled, the port changes from a blocking state to a forwarding state without making the intermediate spanning-tree state changes.
Switch#**show running-config interface** ***type*** *slot/port*	Indicates whether PortFast has been configured on a port. It can also be used to show if configuration has occurred on an EtherChannel link by specifying *port-channel channel_number* in place of *type slot/port*.

Example: Spanning-Tree Operation
The best way to understand how spanning tree operates is to look at an operation example.

Figure 8.5 shows a sample network spanning tree topology and the relevant information used by spanning tree.

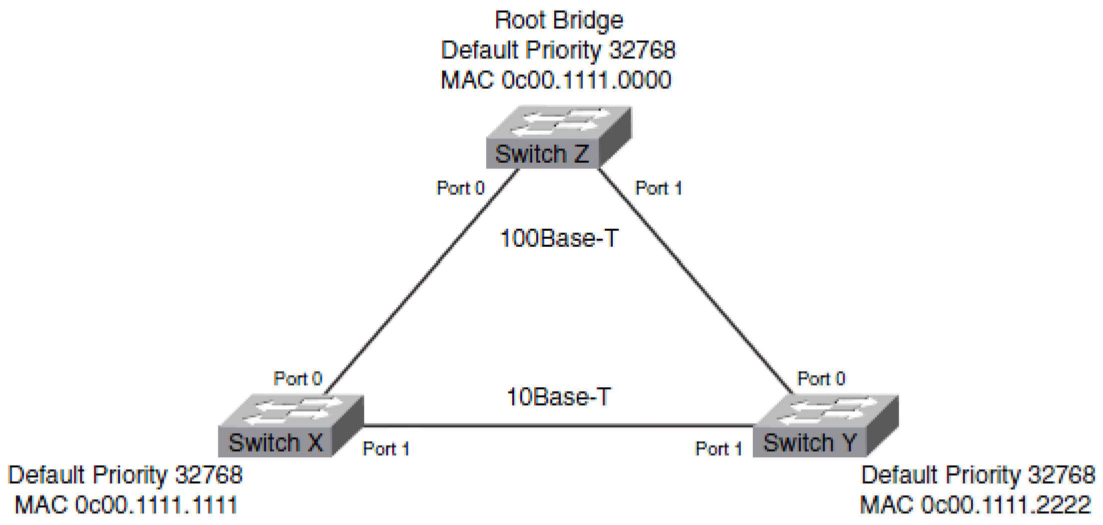


Figure 8.5 Spanning Tree Topology

The following describes the STP port states illustrated in Figure 8.5:

- The root bridge is switching Z, which has the lowest BID.
- The root port is port 0 on switches X and Y. Port 0 is the lowest-cost path to the root on both switches.
- The designated ports on switch Z are port 0 and port 1. All ports on the root are designated ports. Port 1 of switch X is a designated port for the segment between switch X and switch Y. Because switch X and switch Y have the same path cost to the root bridge, the designated port is selected to be on switch X because it has a lower BID than switch Y.
- Port 1 on switch Y is the non-designated port on the segment and is in the blocking state.
- All designated and root ports are in the forwarding state.

Example: Spanning-Tree Path Cost

The spanning-tree path cost is an accumulated total path cost based on the bandwidth of all the links in the path. In the figure, some of the path costs specified in the 802.1D specification is shown. The 802.1D specification has been revised; in the older specification, the cost was calculated based on a bandwidth of 1000 Mbps. The calculation of the new specification uses a nonlinear scale to accommodate higher-speed interfaces.

Table 8.3 describes the spanning-tree path cost calculations based on bandwidth of a link.

Table 8.3 Spanning-Tree Path Costs

Link Speed	Cost (Revised IEEE Specification)	Cost (Previous IEEE Specification)
10 Gbps	2	1
1 Gbps	4	1
100 Mbps	19	10
10 Mbps	100	100

When there is a topology change because of a bridge or link failure, spanning tree adjusts the network topology to ensure connectivity by placing blocked ports in the forwarding state.

NOTE Most Cisco Catalyst switches incorporate the revised cost calculations. A key point to remember about STP cost is that lower costs are better.

Example: Spanning-Tree Recalculation

In Figure 1.6, if switch Z (the root bridge) fails and does not send a BPDU to switch Y within the max_age time (default is 20 seconds, which equals 10 missed BPDUs), switch Y detects the missing BPDU from the root bridge. When the max_age timer on switch Y expires before a new BPDU has been received from switch Z, a spanning-tree recalculation is initiated. Switch Y transitions its blocking port (port 1) from the blocking state to the listening state to the learning state, and then finally to the forwarding state.

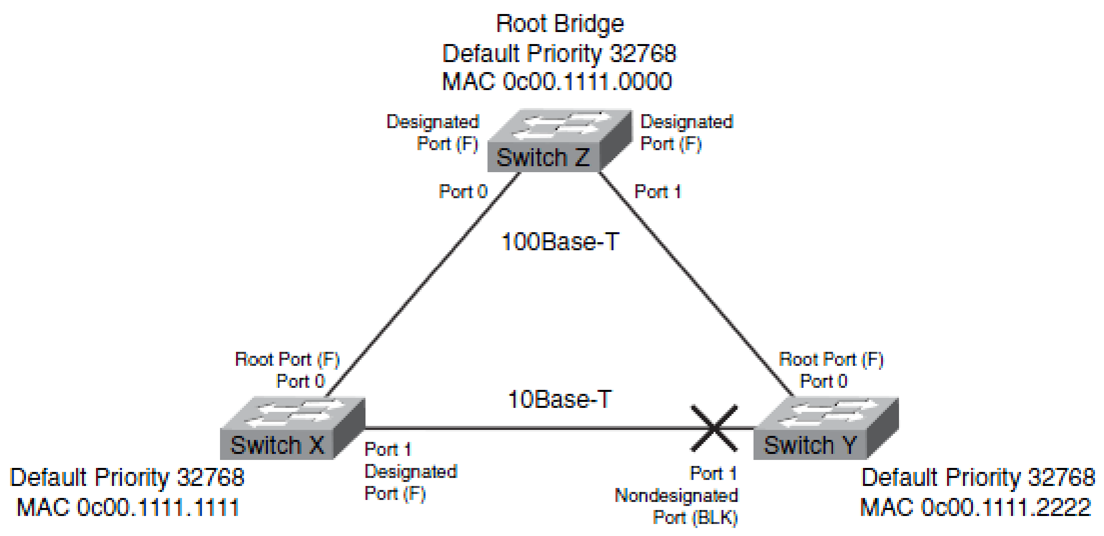


Figure 8.6 Spanning-Tree Recalculation

After all the switch and bridge ports have transitioned to either a forwarding state or a blocking state, switch X becomes the root bridge and forwards traffic between the segments.

Spanning tree mode

Understand the Rapid Spanning-Tree Protocol. The 802.1w STP standard (RSTP) addresses all the problems found in the 802.1d STP protocol and is not Cisco proprietary. This is not enabled on any Cisco switch by default, and if you enable this protocol, you should enable it on all your switches for the fastest convergence times.
Understand the purpose and configuration of VTP. VTP provides propagation of the VLAN database throughout your switched network. All switches must be in the same VTP domain.

Be able to define PVST. Per-VLAN Spanning-Tree; each VLAN runs its own instance of the STP protocol.

IP Routing Technologies

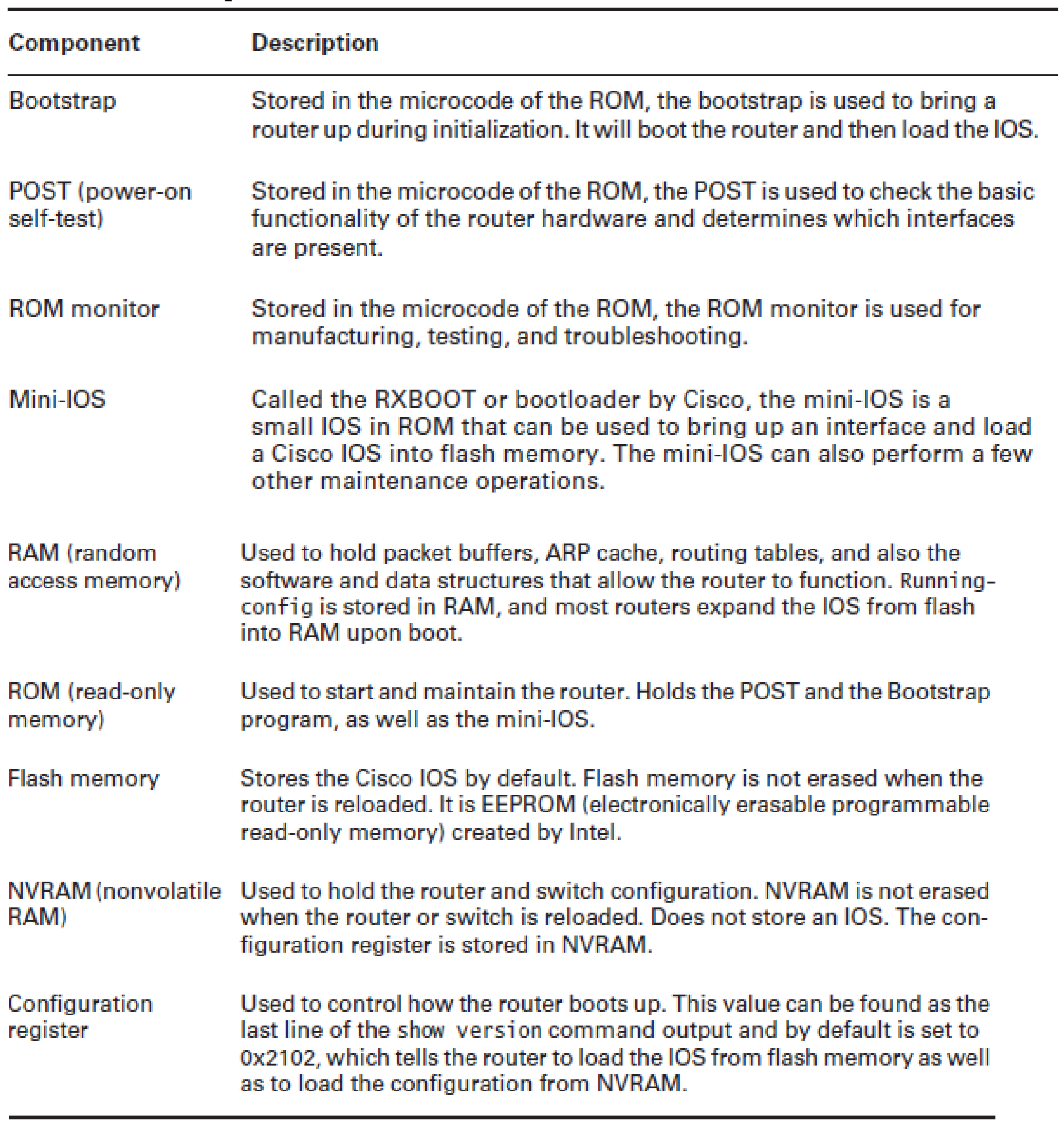
Describe the boot process of Cisco IOS routers

Describe the operation of Cisco routers (including router boot up process, POST, router components)

To configure and troubleshoot a Cisco internetwork, you need to know the major components of Cisco routers and understand what each one does.

POST

Cisco Router Components



Component	Description
Bootstrap	Stored in the microcode of the ROM, the bootstrap is used to bring a router up during initialization. It will boot the router and then load the IOS.
POST (power-on self-test)	Stored in the microcode of the ROM, the POST is used to check the basic functionality of the router hardware and determines which interfaces are present.
ROM monitor	Stored in the microcode of the ROM, the ROM monitor is used for manufacturing, testing, and troubleshooting.
Mini-IOS	Called the RXBOOT or bootloader by Cisco, the mini-IOS is a small IOS in ROM that can be used to bring up an interface and load a Cisco IOS into flash memory. The mini-IOS can also perform a few other maintenance operations.
RAM (random access memory)	Used to hold packet buffers, ARP cache, routing tables, and also the software and data structures that allow the router to function. `Running-config` is stored in RAM, and most routers expand the IOS from flash into RAM upon boot.
ROM (read-only memory)	Used to start and maintain the router. Holds the POST and the Bootstrap program, as well as the mini-IOS.
Flash memory	Stores the Cisco IOS by default. Flash memory is not erased when the router is reloaded. It is EEPROM (electronically erasable programmable read-only memory) created by Intel.
NVRAM (nonvolatile RAM)	Used to hold the router and switch configuration. NVRAM is not erased when the router or switch is reloaded. Does not store an IOS. The configuration register is stored in NVRAM.
Configuration register	Used to control how the router boots up. This value can be found as the last line of the `show version` command output and by default is set to 0x2102, which tells the router to load the IOS from flash memory as well as to load the configuration from NVRAM.

Router bootup process

The Router Boot Sequence
When a router boots up, it performs a series of steps, called the boot sequence, to test the hardware and load the necessary software. The boot sequence consists of the following steps:

1. The router performs a POST. The POST tests the hardware to verify that all components of the device are operational and present. For example, the POST checks for the different interfaces on the router. The POST is stored in and run from ROM (read-only memory).
2. The bootstrap then looks for and loads the Cisco IOS software. The bootstrap is a program in ROM that is used to execute programs. The bootstrap program is responsible for finding where each IOS program is located and then loading the file. By default, the IOS software is loaded from flash memory in all Cisco routers.
3. The IOS software looks for a valid configuration file stored in NVRAM. This file is called startup-config and is only there if an administrator copies the running-config file into NVRAM. (Cisco's new Integrated Services Router (ISR) have a small start upconfig file preloaded.)
4. If a startup-config file is in NVRAM, the router will copy this file and place it in RAM and call the file running-config. The router will use this file to run the router. The router should now be operational. If a startup-config file is not in NVRAM, the router will broadcast out any interface that detects carrier detect (CD) for a TFTP host looking for a configuration, and when that fails (typically it will fail—most people won't even realize the router has attempted this process), it will start the setup mode configuration process.

Managing Configuration Register

All Cisco routers have a 16-bit software register that's written into NVRAM. By default, the configuration register is set to load the Cisco IOS from flash memory and to look for and load the startup-config file from NVRAM. I am going to discuss the configuration register settings and how to use these settings to provide password recovery on your routers.

Configure and verify operation status of a Serial interface

Select the appropriate media, cables, ports, and connectors to connect routers to other network devices and hosts
Ethernet cabling is an important discussion, especially if you are planning on taking the Cisco exams. Three types of Ethernet cables are available:

- Straight-through cable
- Crossover cable
- Rolled cable

Straight-Through Cable
The straight-through cable is used to connect

- Host to switch or hub
- Router to switch or hub

Four wires are used in straight-through cable to connect Ethernet devices. It is relatively simple to create this type; Figure 9.1 shows the four wires used in a straight-through Ethernet cable.

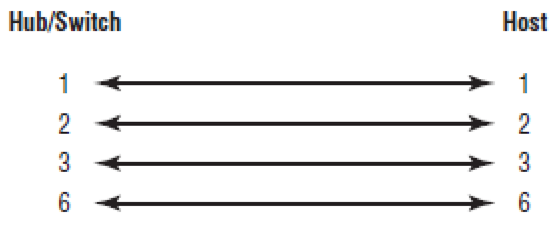


FIGURE 9.1 Straight-through Ethernet cable

Notice that only pins 1, 2, 3, and 6 are used. Just connect 1 to 1, 2 to 2, 3 to 3, and 6 to
6, and you'll be up and networking in no time. However, remember that this would be an Ethernet-only cable and wouldn't work with voice, Token Ring, ISDN, and so on.

Crossover Cable
The crossover cable can be used to connect
- Switch to switch
- Hub to hub
- Host to host
- Hub to switch
- Router direct to host

The same four wires are used in this cable as in the straight-through cable; we just connect different pins together. Figure 9.2 shows how the four wires are used in a crossover Ethernet cable.

Notice that instead of connecting 1 to 1, 2 to 2, and so on, here we connect pins 1 to 3 and 2 to 6 on each side of the cable.

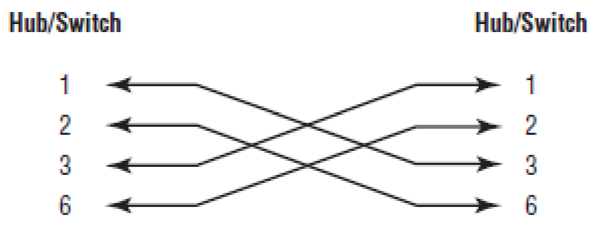


FIGURE 9.2 Crossover Ethernet cable

Manage Cisco IOS Files

Manage IOS configuration files (including save, edit, upgrade, restore)

Any changes that you make to the router configuration are stored in the running-config file. And if you don't enter a copy run start command after you make a change to running config, that change will go poof if the router reboots or gets powered down. So, you probably want to make another backup of the configuration information just in case the router or switch completely dies on you. Even if your machine is healthy and happy, it's good to have for reference and documentation reasons.

I'll describe how to copy the configuration of a router to a TFTP server and how to restore that configuration.

Backing Up the Cisco Router Configuration

To copy the router's configuration from a router to a TFTP server, you can use either the copy running-config tftp or the copy startup-config tftp command. Either one will back up the router configuration that's currently running in DRAM or that's stored in NVRAM.

Verifying the Current Configuration

To verify the configuration in DRAM, use the show running-config command (sh run for short) like this:

```
Router#show running-config
Building configuration...
Current configuration : 776 bytes
!
version 12.4
The current configuration information indicates that the router is running
version 12.4 of
the IOS.
Verifying the Stored Configuration
Next, you should check the configuration stored in NVRAM. To see this, use
the show
startup-config command (sh start for short) like this:
Router#show startup-config
Using 776 out of 245752 bytes
!
```

```
version 12.4
```

The second line shows you how much room your backup configuration is using. Here, you can see that NVRAM is 239KB (again, memory is easier to see with the show version command when using an ISR router) and that only 776 bytes of it are used.

If you're not sure that the files are the same and the running-config file is what you want to use, and then use the copy running-config startup-config. This will help you verify that both files are in fact the same.

Copying the Current Configuration to NVRAM

By copying running-config to NVRAM as a backup, as shown in the following output, you're assured that your running-config will always be reloaded if the router gets rebooted.

In the new IOS version 12.0, you're prompted for the filename you want to use.

```
Router#copy running-config startup-config
Destination filename [startup-config]?[enter]
Building configuration...
[OK]
Router#
```

The reason that the filename prompt appears is that there are now so many options you can use when using the copy command:

```
Router#copy running-config ?
archive: Copy to archive: file system
flash: Copy to flash: file system
ftp: Copy to ftp: file system
http: Copy to http: file system
https: Copy to https: file system
ips-sdf Update (merge with) IPS signature configuration
null: Copy to null: file system
nvram: Copy to nvram: file system
rcp: Copy to rcp: file system
running-config Update (merge with) current system configuration
scp: Copy to scp: file system
startup-config Copy to startup configuration
syslog: Copy to syslog: file system
system: Copy to system: file system
tftp: Copy to tftp: file system
xmodem: Copy to xmodem: file system
ymodem: Copy to ymodem: file system
```

We'll go over the copy command again in a minute.

Copying the Configuration to a TFTP Server

Once the file is copied to NVRAM, you can make a second backup to a TFTP server by using the copy running-config tftp command (copy run tftp for short), like this:

```
Router#copy running-config tftp
Address or name of remote host []?1.1.1.2
Destination filename [router-confg]?todd-confg
!!
776 bytes copied in 0.800 secs (970 bytes/sec)
Router#
```

In the preceding example, I named the file todd-confg because I had not set a hostname for the router. If you have a hostname already configured, the command will automatically use the hostname plus the extension -confg as the name of the file.

Boot preferences

Verifying Flash Memory

Before you attempt to upgrade the Cisco IOS on your router with a new IOS file, it's a good idea to verify that your flash memory has enough room to hold the new image. You verify the amount of flash memory and the file or files being stored in flash memory by using the show flash command (sh flash for short):

```
Router#sh flash
-#- --length-- -----date/time------ path
1 21710744 Jan 2 2007 22:41:14 +00:00 c2800nm-advsecurityk9-mz.124-12.bin
[output cut]
32989184 bytes available (31027200 bytes used)
```

The ISR router above has 64MB of RAM, and roughly half of the memory is in use.

The amount of flash is actually easier to tally using the show version command on the ISR routers:

```
Router#show version
[output cut]
Cisco 2811 (revision 49.46) with 249856K/12288K bytes of memory.
Processor board ID FTX1049A1AB
2 FastEthernet interfaces
4 Serial(sync/async) interfaces
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity enabled.
239K bytes of non-volatile configuration memory.
62720K bytes of ATA CompactFlash (Read/Write)
```

You can see that the amount of flash shows up on the last line. By averaging up, we get the amount of flash to 64MB.

Notice that the filename in this example is c2800nm-advsecurityk9-mz.124-12.bin.
The main difference in the output of the show flash and show version commands is that the show flash command displays all files in flash, and the show version command shows the actual name of the file that the router is using to run the router.

Cisco IOS image(s)

Manage Cisco IOS

Before you upgrade or restore a Cisco IOS, you really should copy the existing file to a TFTP host as a backup just in case the new image crashes and burns. And you can use any TFTP host to accomplish this. By default, the flash memory in a router is used to store the Cisco IOS. I'll describe how to check the amount of flash memory, how to copy the Cisco IOS from flash memory to a TFTP host, and how to copy the IOS from a TFTP host to flash memory.

But before you back up an IOS image to a network server on your intranet, you've got to do these three things:

- Make sure that you can access the network server.
- Ensure that the network server has adequate space for the code image.
- Verify the file naming and path requirements.

And if you have a laptop or workstation's Ethernet port directly connected to a router's Ethernet interface, as shown in Figure 9.3, you need to verify the following before attempting to copy the image to or from the router:

- TFTP server software must be running on the administrator's workstation.
- The Ethernet connection between the router and the workstation must be made with a crossover cable.

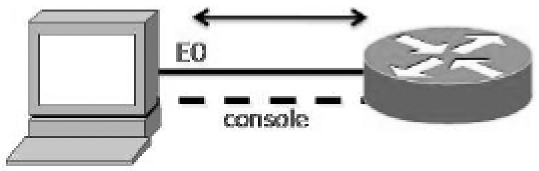


FIGURE 9.3 Copying an IOS from a workstation to a router

- The workstation must be on the same subnet as the router's Ethernet interface.
- The copy flash tftp command must be supplied the IP address of the workstation if you are copying from the router flash.
- And if you're copying "into" flash, you need to verify that there's enough room in flash memory to accommodate the file to be copied.

Verifying Flash Memory

Before you attempt to upgrade the Cisco IOS on your router with a new IOS file, it's a good idea to verify that your flash memory has enough room to hold the new image. You verify the amount of flash memory and the file or files being stored in flash memory by using the show flash command (sh flash for short):

```
Router#sh flash

-#- --length-- -----date/time------ path

1 21710744 Jan 2 2007 22:41:14 +00:00 c2800nm-advsecurityk9-mz.124-12.bin

[output cut]

32989184 bytes available (31027200 bytes used)
```

The ISR router above has 64MB of RAM, and roughly half of the memory is in use.
The amount of flash is actually easier to tally using the show version command on the ISR routers:

```
Router#show version
[output cut]
Cisco 2811 (revision 49.46) with 249856K/12288K bytes of memory.
Processor board ID FTX1049A1AB
2 FastEthernet interfaces
4 Serial(sync/async) interfaces
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity enabled.
239K bytes of non-volatile configuration memory.
62720K bytes of ATA CompactFlash (Read/Write)
```

You can see that the amount of flash shows up on the last line. By averaging up, we get the amount of flash to 64MB.

Notice that the filename in this example is c2800nm-advsecurityk9-mz.124-12.bin.
The main difference in the output of the show flash and show version commands is that the show flash command displays all files in flash, and the show version command shows the actual name of the file that the router is using to run the router.

Backing Up the Cisco IOS
To back up the Cisco IOS to a TFTP server, you use the copy flash tftp command. It's a straightforward command that requires only the source filename and the IP address of the TFTP server.

The key to success in this backup routine is to make sure that you've got good, solid connectivity to the TFTP server. Check this by pinging the TFTP device from the router console prompt like this:

```
Router#ping 1.1.1.2
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 1.1.1.2, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max
= 4/4/8 ms
```

After you ping the TFTP server to make sure that IP is working, you can use the copy flash tftp command to copy the IOS to the TFTP server as shown next:

```
Router#copy flash tftp
Source filename []?c2800nm-advsecurityk9-mz.124-12.bin
Address or name of remote host []?1.1.1.2
Destination filename [c2800nm-advsecurityk9-mz.124-12.bin]?[enter]
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!
!!!!!!!
21710744 bytes copied in 60.724 secs (357532 bytes/sec)
Router#
```

Just copy the IOS filename from either the show flash or show version command and then paste it when prompted for the source filename. In the preceding example, the contents of flash memory were copied successfully to the TFTP server. The address of the remote host is the IP address of the TFTP host, and the source filename is the file in flash memory.

Restoring or Upgrading the Cisco Router IOS

What happens if you need to restore the Cisco IOS to flash memory to replace an original file that has been damaged or if you want to upgrade the IOS? You can download the file from a TFTP server to flash memory by using the copy tftp flash command. This command requires the IP address of the TFTP host and the name of the file you want to download.

But before you begin, make sure that the file you want to place in flash memory is in the default TFTP directory on your host. When you issue the command, TFTP won’t ask you where the file is, so if the file you want to use isn’t in the default directory of the TFTP host, this just won’t work.

```
Router#copy tftp flash
Address or name of remote host []?1.1.1.2
Source filename []?c2800nm-advsecurityk9-mz.124-12.bin
Destination filename [c2800nm-advsecurityk9-mz.124-12.bin]?[enter]
%Warning:There is a file already existing with this name
Do you want to over write? [confirm][enter]
Accessing tftp://1.1.1.2/c2800nm-advsecurityk9-mz.124-12.bin...
Loading c2800nm-advsecurityk9-mz.124-12.bin from 1.1.1.2 (via
FastEthernet0/0):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!
!!!!!!
[OK - 21710744 bytes]
```

```
21710744 bytes copied in 82.880 secs (261954 bytes/sec)
Router#
```

In the above example, I copied the same file into flash memory, so it asked me if I wanted to overwrite it. Remember that we are "playing" with files in flash memory. If I just corrupted my file by overwriting the file, I won't know this until I reboot the router. Be careful with this command! If the file is corrupted, then you'll need to do an IOS restore from ROMMON.

If you are loading a new file and you don't have enough room in flash memory to store both the new and existing copies, the router will ask to erase the contents of flash memory before writing the new file into flash memory.

Licensing

Customers license the right to use Cisco intellectual property (software) from Cisco. When you purchase hardware, you also purchase the software and the license to use it. Hardware arrives with licensed software. When you need a different feature set, you need to purchase the license for it. Upgrade features may be licensed separately, and special Cisco IOS features are licensed separately, either on a flat usage or per-seat basis, depending on the features.

When customers and partners download software from the Cisco.com Software Center, or e-mail downloaded software to someone, they are obligated to comply to the terms and conditions of the Cisco Software License Agreement.

Licensing Best Practices

- Pay an additional fee to upgrade to a new feature set—upgrades require new license. No additional fee is due when the feature set stays the same—updates require no new license.
- Resell software (and license to use) that runs on the Cisco hardware when they resell the hardware.
- Charge customers for upgrade or special feature licenses.
- Tell customers to purchase licenses for software running on their equipment.
- Make one archival copy of software at no charge.
- Do not operate unlicensed software in customer networks
- Do not transfer to customers of Cisco IOS software that is deployed on a Partner's own network
- Do not transfer Cisco IOS software licenses from one company to another except in special circumstances, such as company mergers

Show license

To display information about a Cisco IOS software license, use the show license command in either user EXEC or privileged EXEC mode.

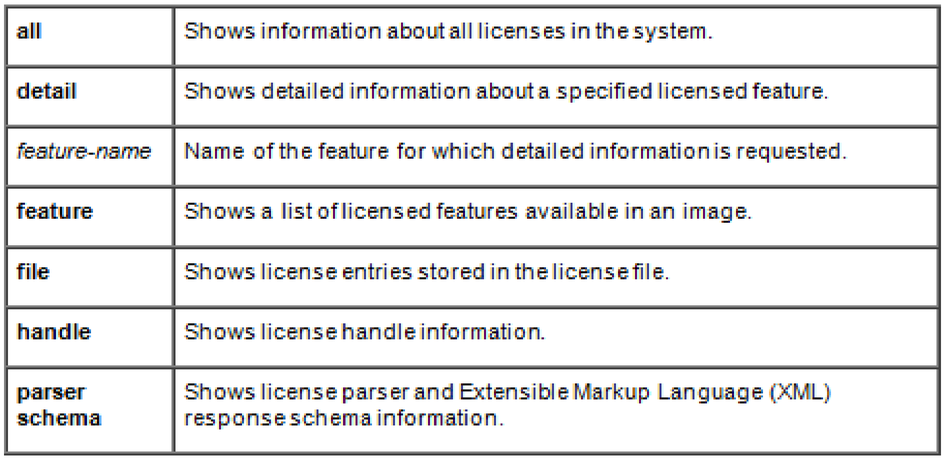
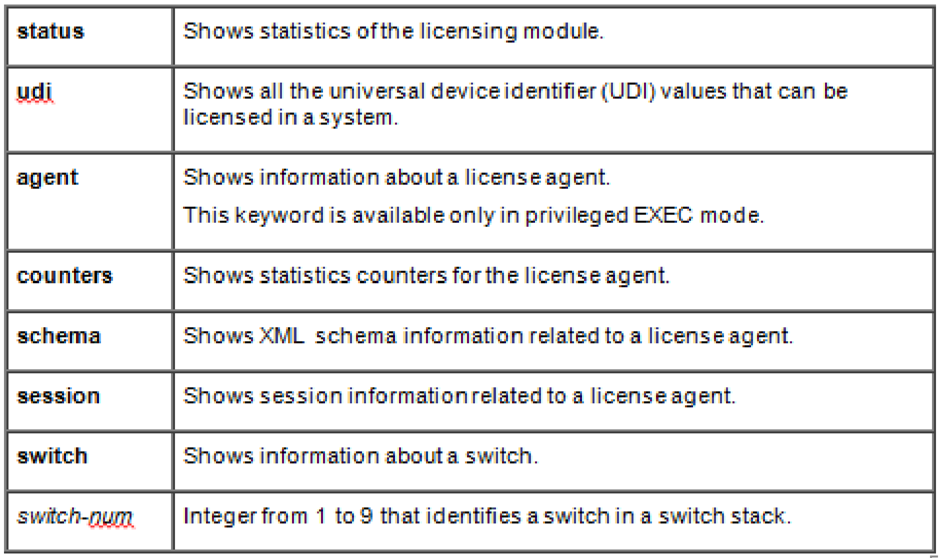
Cisco Router Platforms

show license {all | detail feature-name | feature | file | handle | parser schema | status | udi}

Cisco Catalyst 3560-E and 3750-E Switch Platforms
show license {agent {counters | schema | session} | {all | detail feature-name | feature | file | handle | status | udi}switch switch-num | parser schema}

Cisco Catalyst 3750-E Mixed Switch Stacks
show license {agent {counters | schema | session} | detail {feature-name switch switch-num | switch switch-num} | {all | feature | file | handle | status | udi} switch switch-num | parser schema}

Syntax Description

all	Shows information about all licenses in the system.
detail	Shows detailed information about a specified licensed feature.
feature-name	Name of the feature for which detailed information is requested.
feature	Shows a list of licensed features available in an image.
file	Shows license entries stored in the license file.
handle	Shows license handle information.
parser schema	Shows license parser and Extensible Markup Language (XML) response schema information.
status	Shows statistics of the licensing module.
udi	Shows all the universal device identifier (UDI) values that can be licensed in a system.
agent	Shows information about a license agent. This keyword is available only in privileged EXEC mode.
counters	Shows statistics counters for the license agent.
schema	Shows XML schema information related to a license agent.
session	Shows session information related to a license agent.
switch	Shows information about a switch.
switch-num	Integer from 1 to 9 that identifies a switch in a switch stack.

Usage Guidelines
Use this command to display license information and to help with troubleshooting issues related to Cisco IOS software licenses. This command displays all the licenses in the system.

This command also displays the features that are available but not licensed to execute. Output is grouped according to how the features are stored in license storage.

If a switch ID is specified, information from that switch is displayed. If a switch ID is not specified, details of the local switch are displayed.

Examples

Following is sample output from the show license all command:

```
Router# show license all
 License Store: Primary License Storage
StoreIndex: 0 Feature: TEST_FEATURE_1 Version: 1.0
License Type: Trial
License State: Active, Not in Use, EULA accepted
Trial total period: 1 day 0 hour
Trial period left: 0 second
Lock type: Node locked
Vendor info <PID>AS54XM-AC-RPS</PID><SN>JAE0948QXJW</SN>
License Addition: Additive
License Generation version 135266304
License Precedence: 1
```

Differentiate methods of routing and routing protocols

Compare and contrast methods of routing and routing protocols

A routing protocol is used by routers to dynamically find all the networks in the internetwork and to ensure that all routers have the same routing table. Basically, a routing protocol determines the path of a packet through an internetwork. Examples of routing protocols are RIP, RIPv2, EIGRP, and OSPF.

Once all routers know about all networks, a routed protocol can be used to send user data (packets) through the established enterprise. Routed protocols are assigned to an interface and determine the method of packet delivery. Examples of routed protocols are IP and IPv6.

The administrative distance (AD) is used to rate the trustworthiness of routing information received on a router from a neighbor router. An administrative distance is an integer from 0 to 255, where 0 is the most trusted and 255 means no traffic will be passed via this route.

If a router receives two updates listing the same remote network, the first thing the router checks is the AD. If one of the advertised routes has a lower AD than the other, then the route with the lowest AD will be placed in the routing table.

If both advertised routes to the same network have the same AD, then routing protocol metrics (such as hop count or bandwidth of the lines) will be used to find the best path to the remote network. The advertised route with the lowest metric will be placed in the routing table. But if both advertised routes have the same AD as well as the same metrics, then the routing protocol will load-balance to the remote network (which means that it sends packets down each link).

Table 9.1 shows the default administrative distances that a Cisco router uses to decide which route to take to a remote network.

TABLE 9.1 Default Administrative Distances

Route Source	Default AD
Connected interface	0
Static route	1
EIGRP	90
IGRP	100
OSPF	110
RIP	120
External EIGRP	170
Unknown	255 (this route will never be used)

If a network is directly connected, the router will always use the interface connected to the network. If you configure a static route, the router will then believe that route over any other learned routes. You can change the administrative distance of static routes, but, by default, they have an AD of 1. In our static route configuration, the AD of each route is set at 150 or 151. This lets us configure routing protocols without having to remove the static routes. They'll be used as backup routes in case the routing protocol experiences a failure of some type.

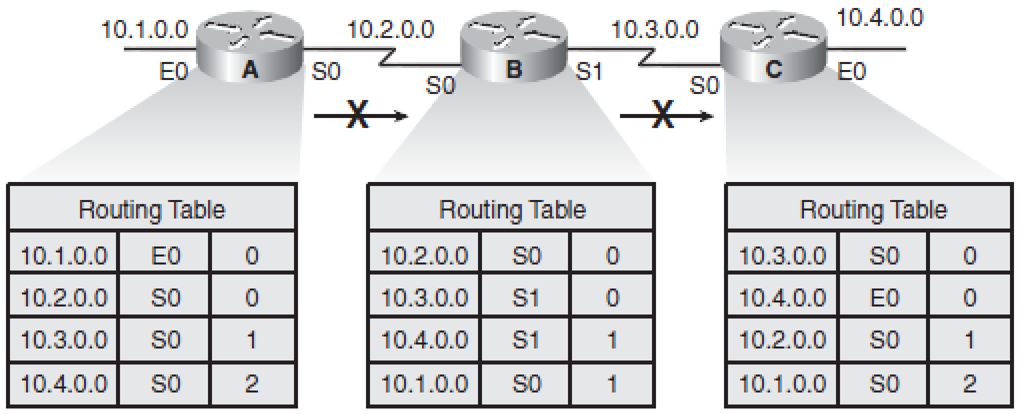
For example, if you have a static route, a RIP-advertised route, and an IGRP-advertised route listing the same network, then, by default, the router will always use the static route unless you change the AD of the static route—which we did.

Split horizon

Preventing Routing Loops with Split Horizon

One way to eliminate routing loops and speed up convergence is through the technique called split horizon. The split horizon rule is that sending information about a route back in the direction from which the original update came is never useful. For example, Figure 9.4 illustrates the following:

- Router B has access to network 10.4.0.0 through Router C. It makes no sense for Router B to announce to Router C that Router B has access to network 10.4.0.0 through Router C.
- Given that Router B passed the announcement of its route to network 10.4.0.0 to Router A, it makes no sense for Router A to announce its distance from network 10.4.0.0 to Router B.
- Having no alternative path to network 10.4.0.0, Router B concludes that network 10.4.0.0 is inaccessible.



Routing Table		
10.1.0.0	E0	0
10.2.0.0	S0	0
10.3.0.0	S0	1
10.4.0.0	S0	2

Routing Table		
10.2.0.0	S0	0
10.3.0.0	S1	0
10.4.0.0	S1	1
10.1.0.0	S0	1

Routing Table		
10.3.0.0	S0	0
10.4.0.0	E0	0
10.2.0.0	S0	1
10.1.0.0	S0	2

Figure 9.4 Split Horizon

Configure and verify OSPF (single area)

Configure, verify, and troubleshoot OSPF
Open Shortest Path First (OSPF) is an open standard routing protocol that's been implemented by a wide variety of network vendors, including Cisco. If you have multiple routers and not all of them are Cisco (what!), then you can't use EIGRP, can you? So, your remaining CCNA objective options are basically RIP, RIPv2, and OSPF.

OSPF works by using the Dijkstra algorithm. First, a shortest path tree is constructed, and then the routing table is populated with the resulting best paths. OSPF converges quickly, although perhaps not as quickly as EIGRP, and it supports multiple, equal-cost routes to the same destination. Like EIGRP, it does support both IP and IPv6 routed protocols.

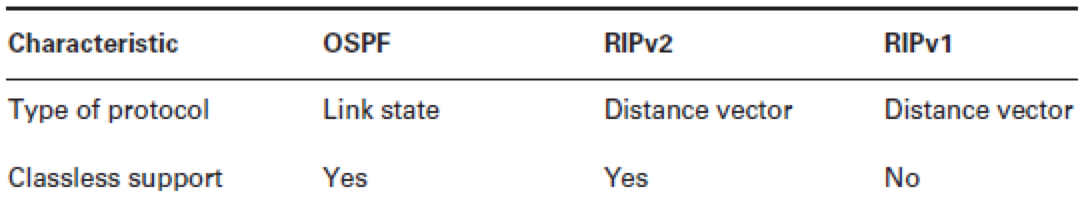
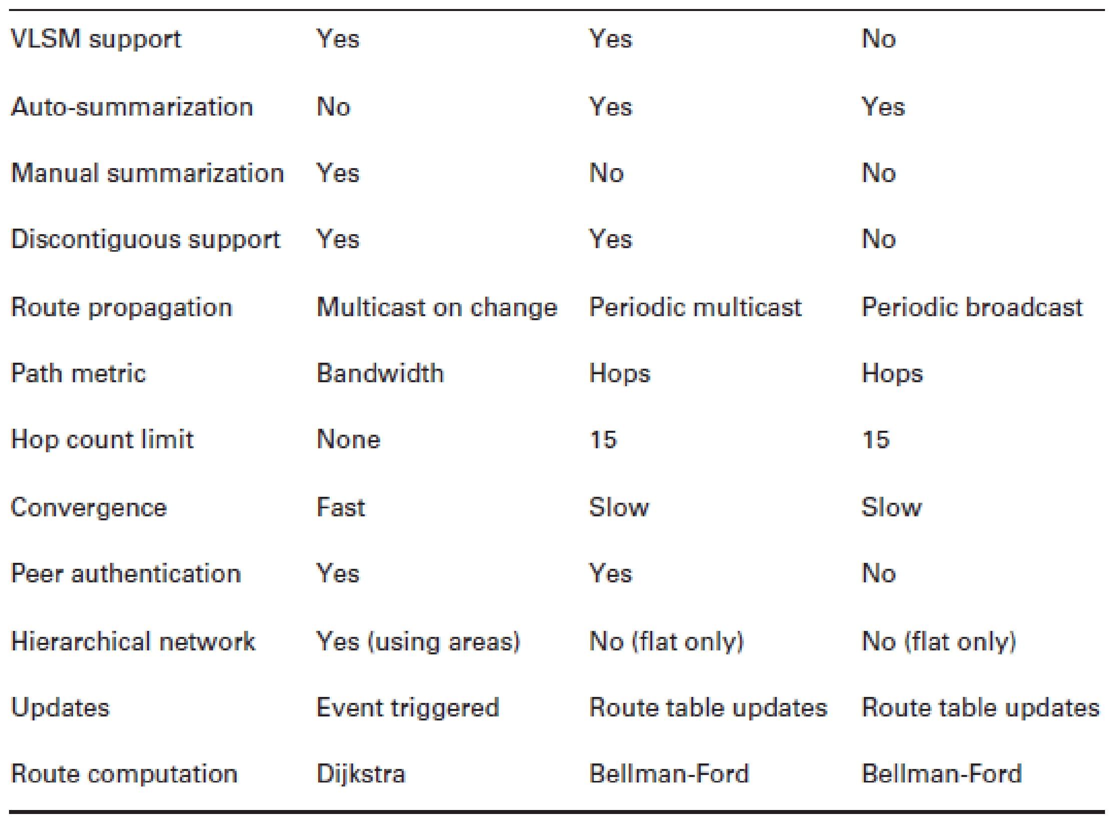
OSPF provides the following features:

- Consists of areas and autonomous systems
- Minimizes routing update traffic
- Allows scalability
- Supports VLSM/CIDR
- Has unlimited hop count
- Allows multi-vendor deployment (open standard)

OSPF is the first link-state routing protocol that most people are introduced to, so it's useful to see how it compares to more traditional distance-vector protocols such as RIPv2 and RIPv1.

Table 9.2 gives you a comparison of these three protocols.

TABLE 9.2 OSPF and RIP comparison

Characteristic	OSPF	RIPv2	RIPv1
Type of protocol	Link state	Distance vector	Distance vector
Classless support	Yes	Yes	No
VLSM support	Yes	Yes	No
Auto-summarization	No	Yes	Yes
Manual summarization	Yes	No	No
Discontiguous support	Yes	Yes	No
Route propagation	Multicast on change	Periodic multicast	Periodic broadcast
Path metric	Bandwidth	Hops	Hops
Hop count limit	None	15	15
Convergence	Fast	Slow	Slow
Peer authentication	Yes	Yes	No
Hierarchical network	Yes (using areas)	No (flat only)	No (flat only)
Updates	Event triggered	Route table updates	Route table updates
Route computation	Dijkstra	Bellman-Ford	Bellman-Ford

OSPF has many features beyond the few I've listed in Table 9.8, and all of them contribute to a fast, scalable, and robust protocol that can be actively deployed in thousands of production networks.

OSPF is supposed to be designed in a hierarchical fashion, which basically means that you can separate the larger internetwork into smaller internetworks called areas. This is the best design for OSPF.

The following are reasons for creating OSPF in a hierarchical design:

- To decrease routing overhead
- To speed up convergence
- To confine network instability to single areas of the network

This does not make configuring OSPF easier, but more elaborate and difficult.

Figure 9.5 shows a typical OSPF simple design. Notice how each router connects to the backbone—called area 0, or the backbone area. OSPF must have an area 0, and all other areas should connect to this area. Routers that connect other areas to the backbone area within an AS are called Area Border Routers (ABRs). Still, at least one interface of the ABR must be in area 0.

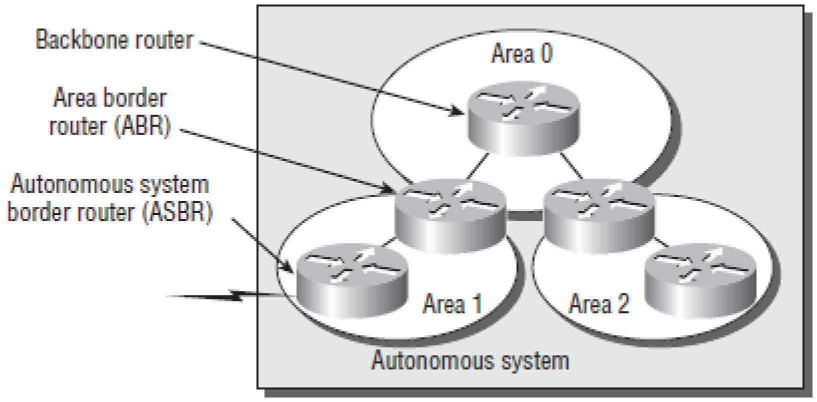


FIGURE 9.5 OSPF design example

OSPF runs inside an autonomous system, but it can also connect multiple autonomous systems. The router that connects this ASs is called an Autonomous System Boundary Router (ASBR). Ideally, you would create other areas of networks to help keep route updates to a minimum and to keep problems from propagating throughout the network.

Configuring OSPF Areas

After identifying the OSPF process, you need to identify the interfaces that you want to activate OSPF communications on as well as the area in which each resides. This will also configure the networks you're going to advertise to others. OSPF uses wildcards in the configuration—which are also used in access-list configurations.

Here’s an OSPF basic configuration example for you:

```
Lab_A#config t
Lab_A(config)#router ospf 1
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area ?
<0-4294967295> OSPF area ID as a decimal value
A.B.C.D OSPF area ID in IP address format
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area 0
```

Remember, the OSPF Process ID number is irrelevant. It can be the same on every router on the network, or it can be different—doesn’t matter. It’s locally significant and just enables the OSPF routing on the router.

I’ll show you the OSPF show commands you need to know in order to do this. We’re going to start by taking a quick look at the routing table of the Corp router:

So, let’s issue a show ip route command on the Corp router:

```
10.0.0.0/24 is subnetted, 12 subnets
O 10.1.11.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
O 10.1.10.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
O 10.1.9.0 [110/74] via 10.1.4.2, 00:01:31, Serial0/1/0
O 10.1.8.0 [110/65] via 10.1.4.2, 00:01:31, Serial0/1/0
O 10.1.12.0 [110/66] via 10.1.5.2, 00:01:31, Serial0/2/0
C 10.1.3.0 is directly connected, Serial0/0/1
C 10.1.2.0 is directly connected, Serial0/0/0
C 10.1.1.0 is directly connected, FastEthernet0/1
O 10.1.7.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
O 10.1.6.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
C 10.1.5.0 is directly connected, Serial0/2/0
C 10.1.4.0 is directly connected, Serial0/1/0
```

The Corp router shows the found routes for all 12 of our networks, with the O representing OSPF internal routes (the Cs are obviously our directly connected networks). It also found the dual routes to networks 10.1.6.0 and 10.1.7.0. I removed the bandwidth and delay commands from under the interface, so the defaults are being used to determine the metric. But remember, OSPF only uses bandwidth to determine the best path to a network.

The show ip ospf Command

The show ip ospf command is used to display OSPF information for one or all OSPF processes running on the router. Information contained therein includes the Router ID, area information, SPF statistics, and LSA timer information. Let’s check out the output from the Corp router:

```
Corp#sh ip ospf
Routing Process "ospf 132" with ID 10.1.5.1
Start time: 04:32:04.116, Time elapsed: 01:27:10.156
Supports only single TOS(TOS0) routes
Supports opaque LSA
Supports Link-local Signaling (LLS)
Supports area transit capability
Router is not originating router-LSAs with maximum metric
Initial SPF schedule delay 5000 msecs
Minimum hold time between two consecutive SPFs 10000 msecs
Maximum wait time between two consecutive SPFs 10000 msecs
Incremental-SPF disabled
Minimum LSA interval 5 secs
Minimum LSA arrival 1000 msecs
LSA group pacing timer 240 secs
Interface flood pacing timer 33 msecs
Retransmission pacing timer 66 msecs
Number of external LSA 0. Checksum Sum 0x000000
Number of opaque AS LSA 0. Checksum Sum 0x000000
Number of DCbitless external and opaque AS LSA 0
Number of DoNotAge external and opaque AS LSA 0
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
Number of areas transit capable is 0
External flood list length 0
Area BACKBONE(0)
Number of interfaces in this area is 5
Area has no authentication
SPF algorithm last executed 00:14:52.220 ago
SPF algorithm executed 14 times
Area ranges are
Number of LSA 6. Checksum Sum 0x03C06F
Number of opaque link LSA 0. Checksum Sum 0x000000
Number of DCbitless LSA 0
Number of indication LSA 0
Number of DoNotAge LSA 0
Flood list length 0
```

Notice the Router ID (RID) of 10.1.5.1, which is the highest IP address configured on the router.

The show ip ospf database Command

Using the show ip ospf database command will give you information about the number of routers in the internetwork (AS) plus the neighboring router's ID (this is the topology database I mentioned earlier). Unlike the show ip eigrp topology command, this command shows the "OSPF routers," not each and every link in the AS as EIGRP does.

The show ip ospf interface Command

The show ip ospf interface command displays all interface-related OSPF information.

Data is displayed about OSPF information for all interfaces or for specified interfaces. (I'll bold some of the important things.)

```
Corp#sh ip ospf interface f0/1
FastEthernet0/1 is up, line protocol is up
Internet Address 10.1.1.1/24, Area 0
Process ID 132, Router ID 10.1.5.1, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DR, Priority 1
Designated Router (ID) 10.1.5.1, Interface address 10.1.1.1
No backup designated router on this network
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
oob-resync timeout 40
Hello due in 00:00:01
Supports Link-local Signaling (LLS)
Index 1/1, flood queue length 0
Next 0x0(0)/0x0(0)
Last flood scan length is 0, maximum is 0
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 0, Adjacent neighbor count is 0
Suppress hello for 0 neighbor(s)
```

The following information is displayed by this command:

- Interface IP address
- Area assignment
- Process ID
- Router ID
- Network type
- Cost
- Priority
- DR/BDR election information (if applicable)
- Hello and Dead timer intervals
- Adjacent neighbor information

The reason I used the show ip ospf interface f0/1 command is that I knew that there would be a designated router elected on the FastEthernet broadcast multi-access network.

We'll get into DR and DBR elections in detail in a minute.

The show ip ospf neighbor Command
The show ip ospf neighbor command is super-useful because it summarizes the pertinent OSPF information regarding neighbors and the adjacency state. If a DR or BDR exists, that information will also be displayed. Here's a sample:

```
Corp#sh ip ospf neighbor
Neighbor ID Pri State Dead Time Address Interface
10.1.11.1 0 FULL/ - 00:00:37 10.1.5.2 Serial0/2/0
```

```
10.1.9.1 0 FULL/ - 00:00:34 10.1.4.2 Serial0/1/0
10.1.7.1 0 FULL/ - 00:00:38 10.1.3.2 Serial0/0/1
10.1.7.1 0 FULL/ - 00:00:34 10.1.2.2 Serial0/0/0
```

Debugging OSPF

Debugging is a great tool for any protocol, so let's take a look in Table 9.4 at a few debugging commands for troubleshooting OSPF.

TABLE 9.4 Debugging Commands for Troubleshooting OSPF

Command	Description/Function
`debug ip ospf packet`	Shows Hello packets being sent and received on your router.
`debug ip ospf hello`	Shows Hello packets being sent and received on your router. Shows more detail than the debug `ip ospf packet` output.
`debug ip ospf adj`	Shows DR and DBR elections on a broadcast and nonbroadcast multi-access network.

I'll start by showing you the output from the Corp router I got using the debug ip ospf packet command:

```
Corp#debug ip ospf packet
OSPF packet debugging is on
*Mar 23 01:20:42.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
Corp#
*Mar 23 01:20:45.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/1
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.4
aid:0.0.0.0 chk:8074 aut:0 auk: from Serial0/2/0
*Mar 23 01:20:52.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
*Mar 23 01:20:55.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
*Mar 23 01:20:55.527: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/1
*Mar 23 01:20:55.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.4
aid:0.0.0.0 chk:8074 aut:0 auk: from Serial0/2/0
```

In the preceding output, you can see that our router is both sending and receiving Hello packets every 10 seconds from neighbor (adjacent) routers. The next command will provide us with the same information, but with more detail. For example, you can see the multicast address used (224.0.0.5) and the area:

```
Corp#debug ip ospf hello
```

```
*Mar 23 01:18:41.103: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/1/0 from 10.1.4.1
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
FastEthernet0/1 from 10.1.1.1
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/0 from 10.1.2.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/2/0 from 10.1.5.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/1 from 10.1.3.1
*Mar 23 01:18:42.199: OSPF: Rcv hello from 172.16.10.3 area 0 from
Serial0/1/0 10.1.4.2
*Mar 23 01:18:42.199: OSPF: End of hello processing
*Mar 23 01:18:45.519: OSPF: Rcv hello from 172.16.10.2 area 0 from
Serial0/0/0 10.1.2.2
*Mar 23 01:18:45.519: OSPF: End of hello processing
*Mar 23 01:18:45.543: OSPF: Rcv hello from 172.16.10.2 area 0 from
Serial0/0/1 10.1.3.2
*Mar 23 01:18:45.543: OSPF: End of hello processing
*Mar 23 01:18:45.543: OSPF: Rcv hello from 172.16.10.4 area 0 from
Serial0/2/0 10.1.5.2
*Mar 23 01:18:45.543: OSPF: End of hello processing
```

The last debug command I'm going show you is the debug ip ospf adj command, which will show you elections as they occur on broadcast and nonbroadcast multi-access networks:

```
Corp#debug ip ospf adj
OSPF adjacency events debugging is on
*Mar 23 01:24:34.823: OSPF: Interface FastEthernet0/1 going Down
*Mar 23 01:24:34.823: OSPF: 172.16.10.1 address 10.1.1.1 on
FastEthernet0/1 is dead, state DOWN
*Mar 23 01:24:34.823: OSPF: Neighbor change Event on interface
FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: DR/BDR election on FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: DR: none BDR: none
*Mar 23 01:24:34.823: OSPF: Flush network LSA immediately
*Mar 23 01:24:34.823: OSPF: Remember old DR 172.16.10.1 (id)
*Mar 23 01:24:35.323: OSPF: We are not DR to build Net Lsa for
interface FastEthernet0/1
*Mar 23 01:24:35.323: OSPF: Build router LSA for area 0, router ID
172.16.10.1, seq 0x80000006
*Mar 23 01:24:35.347: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 148 LSA count 1
*Mar 23 01:24:40.703: OSPF: Interface FastEthernet0/1 going Up
*Mar 23 01:24:41.203: OSPF: Build router LSA for area 0, router ID
```

```
172.16.10.1, seq 0x80000007
*Mar 23 01:24:41.231: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 160 LSA count 1
```

Configure and verify EIGRP (single AS)

Configure, verify, and troubleshoot EIGRP
There are two modes from which EIGRP commands are entered: router configuration mode and interface configuration mode. Router configuration mode enables the protocol, determines which networks will run EIGRP, and sets global characteristics. Interface configuration mode allows customization of summaries, metrics, timers, and bandwidth.

To start an EIGRP session on a router, use the router eigrp command followed by the autonomous system number of your network. You then enter the network numbers connected to the router using the network command followed by the network number.

Let's look at an example of enabling EIGRP for autonomous system 20 on a router connected to two networks, with the network numbers being 10.3.1.0/24 and 172.16.10.0/24:

```
Router#config t
Router(config)#router eigrp 20
Router(config-router)#network 172.16.0.0
Router(config-router)#network 10.0.0.0
```

Remember—as with RIP, you use the classful network address, which is all subnet and host bits turned off.

Understand that the AS number is irrelevant—that is, as long as all routers use the same number! You can use any number from 1 to 65,535

Feasible Distance / Feasible Successors /Administrative distance

EIGRP will load-balance across both links automatically when they are of equal variance (equal cost), but EIGRP can also load-balance across unequal-cost links as well if we use the variance command. The variance metric is set to 1 by default, meaning that only equal-cost links will load balance.

You can change the metric anywhere up to 128. Changing a variance value enables EIGRP to install multiple, loop-free routes with unequal cost in a local routing table. So basically, if the variance is set to 1, only routes with the same metric as the successor will be installed in the local routing table. And, for example, if the variance is set to 2, any EIGRP learned route with a metric less than two times the successor metric will be installed in the local routing table (if it is already a feasible successor).

Now's a great time for us to check out some debugging outputs. First, let's use the debug eigrp packet command that will show our Hello packets being sent between neighbor routers:

```
Corp#debug eigrp packet
```

```
EIGRP Packets debugging is on
(UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB,
SIAQUERY, SIAREPLY)
Corp#
*Mar 21 23:17:35.050: EIGRP: Sending HELLO on FastEthernet0/1
*Mar 21 23:17:35.050: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
*Mar 21 23:17:35.270: EIGRP: Received HELLO on Serial0/1/0 nbr 10.1.4.2
*Mar 21 23:17:35.270: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:35.294: EIGRP: Received HELLO on Serial0/0/0 nbr 10.1.2.2
*Mar 21 23:17:35.294: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:38.014: EIGRP: Received HELLO on Serial0/2/0 nbr 10.1.5.2
*Mar 21 23:17:38.014: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
```

Feasibility condition

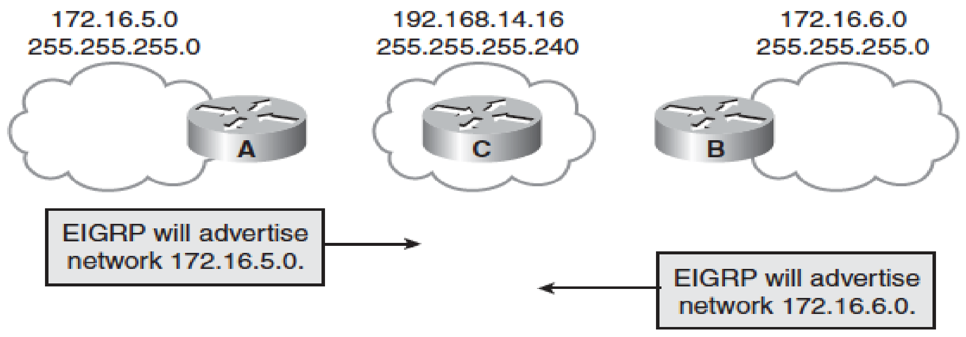
The feasible distance is the best metric to reach the destination or the best metric that was known when the route went active. This value is used in the feasibility condition check. If the reported distance of the router (the metric after the slash) is less than the feasible distance, the feasibility condition is met and that path is a feasible successor. After the software determines it has a feasible successor, it does not need to send a query for that destination.

Router ID

The show ip eigrp topology command displays the EIGRP router ID. The EIGRP router ID comes from the highest IP address assigned to a loopback interface. If no loopback interfaces are configured, the highest IP address assigned to any other active interface is chosen as the router ID. No two EIGRP routers can have the same EIGRP router ID. If they do, you will experience problems exchanging routes between the two routers with equal router IDs.

Auto summary

To disable automatic summarization, use the no auto-summary command in the EIGRP router configuration mode. When this command is used, both Router A and Router B will advertise the route specific to the subnet of a given interface, as shown in the following figure:


172.16.5.0
255.255.255.0
192.168.14.16
255.255.255.240
172.16.6.0
255.255.255.0
A
C
B
EIGRP will advertise network 172.16.5.0.
EIGRP will advertise network 172.16.6.0.

- Path selection
- Load balancing

Typically, networks are configured with multiple paths to a remote network. When these paths are equal or nearly equal, it makes sense to utilize all the available paths. Unlike Layer 2 forwarding, Layer 3 forwarding has the capability to load-balance between multiple paths. That is, the router can send frames out multiple interfaces to reduce the amount of traffic sent to a single network connection. The key to this feature is that the network paths must be of equal cost (or nearly equal for some protocols like EIGRP). EIGRP uses a metric to compute the costs to a given network.

Equal

Load Balancing Across Equal Paths
Equal-cost load balancing is the capability of a router to distribute traffic over all its network ports that are the same metric from the destination address. Load balancing increases the use of network segments and increases effective network bandwidth.

For IP, Cisco IOS Software applies load balancing across up to four equal-cost paths by default. With the maximum-paths maximum-path router configuration command, up to 16 equal-cost routes can be kept in the routing table. If you set the maximum-path to 1, you disable load balancing. When a packet is process switched, load balancing over equal-cost paths occurs on a per-packet basis. When packets are fast switched, load balancing over equal-cost paths occurs on a per-destination basis.

Unequal

Configuring Load Balancing Across Unequal-Cost Paths

EIGRP can also balance traffic across multiple routes that have different metrics, which is called unequal-cost load balancing. The degree to which EIGRP performs load balancing is controlled with the variance command.

The multiplier parameter for the variance command is a value from 1 to 128, used for load balancing. The default is 1, which indicates that only equal-cost load balancing is being performed. The multiplier defines the range of metric values that are accepted for load balancing by the EIGRP process.

IP Services

Recognize High availability (FHRP)

VRRP

There are several ways a LAN client can determine which router should be the first hop to a particular remote destination. The client can use a dynamic process or static configuration. Examples of dynamic router discovery are as follows:

- **Proxy ARP—**The client uses Address Resolution Protocol (ARP) to get the destination it wants to reach, and a router will respond to the ARP request with its own MAC address.
- **Routing protocol**—The client listens to dynamic routing protocol updates (for example, from Routing Information Protocol [RIP]) and forms its own routing table.
- **ICMP Router Discovery Protocol (IRDP) client**—The client runs an Internet Control Message Protocol (ICMP) router discovery client.

The drawback to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, in the event of a router failure, the process of switching to another router can be slow.

An alternative to dynamic discovery protocols is to statically configure a default router on the client. This approach simplifies client configuration and processing, but creates a single point of failure. If the default gateway fails, the LAN client is limited to communicating only on the local IP network segment and is cut off from the rest of the network.

VRRP can solve the static configuration problem. VRRP enables a group of routers to form a single virtual router. The LAN clients can then be configured with the virtual router as their default gateway. The virtual router, representing a group of routers, is also known as a VRRP group.
VRRP is supported on Ethernet, Fast Ethernet, BVI, and Gigabit Ethernet interfaces, and on MPLS VPNs, VRF-aware MPLS VPNs, and VLANs.

VRRP Benefits

Redundancy
VRRP enables you to configure multiple routers as the default gateway router, which reduces the possibility of a single point of failure in a network.

Load Sharing
You can configure VRRP in such a way that traffic to and from LAN clients can be shared by multiple routers, thereby sharing the traffic load more equitably among available routers.

Multiple Virtual Routers
VRRP supports up to 255 virtual routers (VRRP groups) on a router physical interface, subject to the platform supporting multiple MAC addresses. Multiple virtual router support enables you to implement redundancy and load sharing in your LAN topology.

Multiple IP Addresses
The virtual router can manage multiple IP addresses, including secondary IP addresses. Therefore, if you have multiple subnets configured on an Ethernet interface, you can configure VRRP on each subnet.

Preemption
The redundancy scheme of VRRP enables you to preempt a virtual router backup that has taken over for a failing virtual router master with a higher priority virtual router backup that has become available.

Authentication
VRRP message digest 5 (MD5) algorithm authentications protects against VRRP-spoofing software and uses the industry-standard MD5 algorithm for improved reliability and security.

Advertisement Protocol
VRRP uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address (224.0.0.18) for VRRP advertisements. This addressing scheme minimizes the number of routers that must service the multicasts and allows test equipment to accurately identify VRRP packets on a segment. The IANA assigned VRRP the IP protocol number 112.

VRRP Object Tracking
VRRP object tracking provides a way to ensure the best VRRP router is the virtual router master for the group by altering VRRP priorities to the status of tracked objects such as the interface or IP route states.

HSRP

Most IP hosts have an IP address of a single router configured as the default gateway. When HSRP is used, the HSRP virtual IP address is configured as the host’s default gateway instead of the IP address of the router.

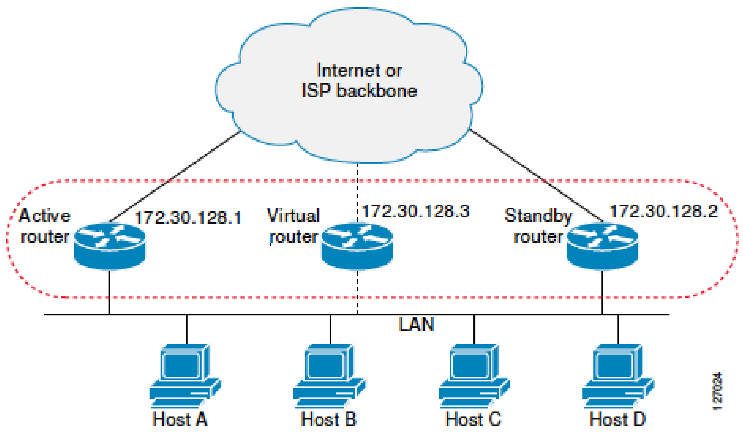
HSRP is useful for hosts that do not support a router discovery protocol (such as ICMP Router Discovery Protocol [IRDP]) and cannot switch to a new router when their selected router reloads or loses power. Because existing TCP sessions can survive the failover, this protocol also provides a more transparent recovery for hosts that dynamically choose a next hop for routing IP traffic.
When HSRP is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among a group of routers running HSRP. The address of this HSRP group is referred to as the virtual IP address. One of these devices is selected by the protocol to be the active router. The active router receives and routes packets destined for the MAC address of the group. For n routers running HSRP, n + 1 IP and MAC addresses are assigned.

HSRP detects when the designated active router fails, at which point a selected standby router assumes control of the MAC and IP addresses of the Hot Standby group. A new standby router is also selected at that time.

HSRP uses a priority mechanism to determine which HSRP configured router is to be the default active router. To configure a router as the active router, you assign it a priority that is higher than the priority of all the other HSRP-configured routers. The default priority is 100, so if you configure just one router to have a higher priority, that router will be the default active router. Devices that are running HSRP send and receive multicast UDP-based hello messages to detect router failure and to designate active and standby routers. When the active router fails to send a hello message within a configurable period of time, the standby router with the highest priority becomes the active router. The transition of packet forwarding functions between routers is completely transparent to all hosts on the network.

You can configure multiple Hot Standby groups on an interface, thereby making fuller use of redundant routers and load sharing.

The figure below shows a network configured for HSRP. By sharing a virtual MAC address and IP address, two or more routers can act as a single virtual router. The virtual router does not physically exist but represents the common default gateway for routers that are configured to provide backup to each other. You do not need to configure the hosts on the LAN with the IP address of the active router. Instead, you configure them with the IP address (virtual IP address) of the virtual router as their default gateway. If the active router fails to send a hello message within the configurable period of time, the standby router takes over and responds to the virtual addresses and becomes the active router, assuming the active router duties.



GLBP

GLBP provides automatic router backup for IP hosts configured with a single default gateway on an IEEE 802.3 LAN. Multiple first-hop routers on the LAN combine to offer a single virtual first-hop IP router while sharing the IP packet forwarding load. Other routers on the LAN may act as redundant GLBP routers that will become active if any of the existing forwarding routers fail.
GLBP performs a similar function for the user as HSRP and VRRP. HSRP and VRRP allow multiple routers to participate in a virtual router group configured with a virtual IP address. One member is elected to be the active router to forward packets sent to the virtual IP address for the group. The other routers in the group are redundant until the active router fails. These standby routers have unused bandwidth that the protocol is not using. Although multiple virtual router groups can be configured for the same set of routers, the hosts must be configured for different default gateways, which result in an extra administrative burden.

The advantage of GLBP is that it additionally provides load balancing over multiple routers (gateways) using a single virtual IP address and multiple virtual MAC addresses. The forwarding load is shared among all routers in a GLBP group rather than being handled by a single router while the other routers stand idle. Each host is configured with the same virtual IP address, and all routers in the virtual router group participate in forwarding packets. GLBP members communicate between each other through hello messages sent every 3 seconds to the multicast address 224.0.0.102, UDP port 3222 (source and destination).

Configure and verify Syslog

Utilize syslog output

Most Cisco devices use the syslog protocol to manage system logs and alerts. But unlike their PC and server counterparts, Cisco devices lack large internal storage space for storing these logs. To overcome this limitation, Cisco devices offer the following two options:

- **Internal buffer**— The device's operating system allocates a small part of memory buffers to log the most recent messages. The buffer size is limited to few kilobytes. This option is enabled by default. However, when the device reboots, these syslog messages are lost.
- **Syslog**— Use a UNIX-style SYSLOG protocol to send messages to an external device for storing. The storage size does not depend on the router's resources and is limited only by the available disk space on the external syslog server. This option is not enabled by default.

Before configuring a Cisco device to send syslog messages, make sure that it is configured with the right date, time, and time zone. Syslog data would be useless for troubleshooting if it shows the wrong date and time. You should configure all network devices to use NTP. Using NTP ensures a correct and synchronized system clock on all devices within the network. Setting the devices with the accurate time is helpful for event correlation.

To enable syslog functionality in a Cisco network, you must configure the built-in syslog client within the Cisco devices.

Cisco devices use a severity level of warnings through emergencies to generate error messages about software or hardware malfunctions. The debugging level displays the output of debug commands. The Notice level displays interface up or down transitions and system restart messages. The informational level reloads requests and low-process stack messages.

Describe SNMP v2 & v3

SNMP is an application-layer protocol that provides a message format for communication between managers and agents. The SNMP system consists of an SNMP manager, an SNMP agent, and a MIB. The SNMP manager can be part of a network management system (NMS) such as Cisco Works. The agent and MIB reside on the switch. To configure SNMP on the switch, you define the relationship between the manager and the agent.

The SNMP agent contains MIB variables whose values the SNMP manager can request or change. A manager can get a value from an agent or store a value into the agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to a manager's requests to get or set data.

An agent can send unsolicited traps to the manager. Traps are messages alerting the SNMP manager to a condition on the network. Traps can mean improper user authentication, restarts, link status (up or down), MAC address tracking, closing of a TCP connection, loss of connection to a neighbor, or other significant events.

SNMP Versions

This software release supports these SNMP versions:

- SNMPv1—The Simple Network Management Protocol, a Full Internet Standard, defined in RFC 1157.
- SNMPv2C replaces the Party-based Administrative and Security Framework of SNMPv2Classic with the community-string-based Administrative Framework of SNMPv2C while retaining the bulk retrieval and improved error handling of SNMPv2Classic. It has these features:
- SNMPv2—Version 2 of the Simple Network Management Protocol, a Draft Internet Standard, defined in RFCs 1902 through 1907.
- SNMPv2C—The community-string-based Administrative Framework for SNMPv2, an Experimental Internet Protocol defined in RFC 1901.
- SNMPv3—Version 3 of the SNMP is an interoperable standards-based protocol defined in RFCs 2273 to 2275. SNMPv3 provides secure access to devices by authenticating and encrypting packets over the network and includes these security features:
  - Message integrity—ensuring that a packet was not tampered with in transit
  - Authentication—determining that the message is from a valid source
  - Encryption—mixing the contents of a package to prevent it from being read by an unauthorized source.

SNMP v2
SNMP version 2 (SNMPv2) is an evolution of the SNMPv1. The Get, GetNext, and Set operations used in SNMPv1 are exactly the same as those used in SNMPv2. However, SNMPv2 adds and enhances some protocol operations. The SNMPv2 Trap operation, for example, serves the same function as that used in SNMPv1, but it uses a different message format and is designed to replace the SNMPv1 Trap.

SNMPv2 also defines two new operations: GetBulk and Inform. The GetBulk operation is used to efficiently retrieve large blocks of data. The Inform operation allows one NMS to send trap information to another NMS and to then receive a response. In SNMPv2, if the agent responding to GetBulk operations cannot provide values for all the variables in a list, it provides partial results.

SNMP v3
Security has been the biggest weakness of SNMP since the beginning. Authentication in SNMP Versions 1 and 2 amounts to nothing more than a password (community string) sent in clear text between a manager and agent. Any security-conscious network or system administrator knows that clear-text passwords provide no real security at all. It is trivial for someone to intercept the community string, and once he has it, he can use it to retrieve information from devices on your network, modify their configuration, and even shut them down.

The *Simple Network Management Protocol Version 3* (SNMPv3) addresses the security problems that have plagued both SNMPv1 and SNMPv2. For all practical purposes, security is the only issue SNMPv3 addresses; there are no other changes to the protocol. There are no new operations; SNMPv3 supports all the operations defined by Versions 1 and 2. There are several new textual conventions, but these are really just more precise ways of interpreting the data types that were defined in earlier versions.

Troubleshooting

Identify and correct common network problems

Determining IP Address Problems
It’s common for a host, router, or other network device to be configured with the wrong IP address, subnet mask, or default gateway. Because this happens way too often, I’m going to teach you how to both determine and fix IP address configuration errors.

Once you’ve worked through the four basic steps of troubleshooting and determined there’s a problem, you obviously then need to find and fix it. It really helps to draw out the network and IP addressing scheme. If it’s already done, consider yourself lucky and go buy a lottery ticket, because although it should be done, it rarely is. And if it is, it’s usually outdated or inaccurate anyway. Typically it is not done, and you’ll probably just have to bite the bullet and start from scratch.

Once you have your network accurately drawn out, including the IP addressing scheme, you need to verify each host’s IP address, mask, and default gateway address to determine the problem. (I’m assuming that you don’t have a physical problem or that if you did, you’ve already fixed it.)

Let’s check out the example illustrated in Figure 11.1. A user in the sales department calls and tells you that she can’t get to ServerA in the marketing department. You ask her if she can get to ServerB in the marketing department, but she doesn’t know because she doesn’t have rights to log on to that server. What do you do?

You ask the client to go through the four troubleshooting steps that you learned. Steps 1 through 3 work, but step 4 fails. By looking at the figure, can you determine the problem? Look for clues in the network drawing. First, the WAN link between the Lab_A router and the Lab_B router shows the mask as a /27. You should already know that this mask is 255.255.255.224 and then determine that all networks are using this mask. The network address is 192.168.1.0. What are our valid subnets and hosts? 256 – 224 = 32, so this makes our subnets 32, 64, 96, 128, and so on. So, by looking at the figure, you can see that subnet 32 is being used by the sales department, the WAN link is using subnet 96, and the marketing department is using subnet 64.

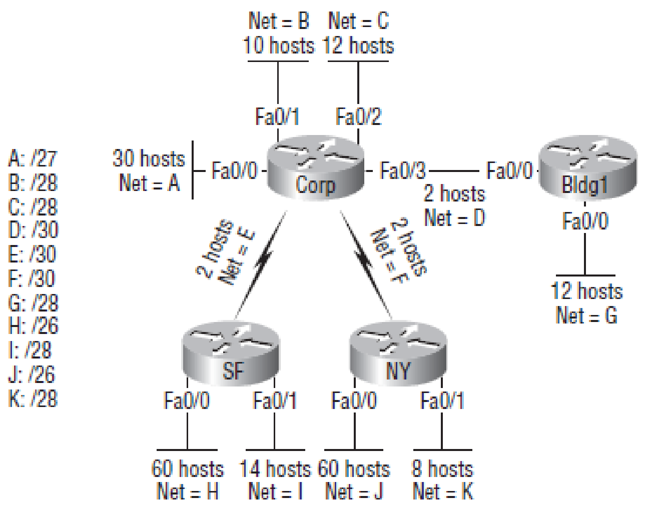


FIGURE 11.1 IP address problem 1

Now you've got to determine what the valid host ranges are for each subnet. You should now be able to easily determine the subnet address, broadcast addresses, and valid host ranges. The valid hosts for the Sales LAN are 33 through 62—the broadcast address is 63 because the next subnet is 64, right? For the Marketing LAN, the valid hosts are 65 through 94 (broadcast 95), and for the WAN link, 97 through 126 (broadcast 127). By looking at the figure, you can determine that the default gateway on the Lab_B router is incorrect. That address is the broadcast address of the 64 subnet, so there's no way it could be a valid host.

Did you get all that? Maybe we should try another one, just to make sure. Figure 11.2 shows a network problem. A user in the Sales LAN can't get to ServerB. You have the user run through the four basic troubleshooting steps and find that the host can communicate to the local network but not to the remote network. Find and define the IP addressing problem.

If you use the same steps used to solve the last problem, you can see first that the WAN link again provides the subnet mask to use— /29, or 255.255.255.248. You need to determine what the valid subnets, broadcast addresses, and valid host ranges are to solve this problem. The 248 mask is a block size of 8 (256 – 248 = 8), so the subnets both start and increment in multiples of 8. By looking at the figure, you see that the Sales LAN is in the 24 subnet, the WAN is in the 40 subnet, and the Marketing LAN is in the 80 subnet. Can you see the problem yet? The valid host range for the Sales LAN is 25–30, and the configuration appears correct.

The valid host range for the WAN link is 41–46, and this also appears correct. The valid host range for the 80 subnet is 81–86, with a broadcast address of 87 because the next subnet is 88.

ServerB has been configured with the broadcast address of the subnet. Okay, now that you can figure out misconfigured IP addresses on hosts, what do you do if a host doesn't have an IP address and you need to assign one? What you need to do is look at other hosts on the LAN and figure out the network, mask, and default gateway.

Let’s take a look at a couple of examples of how to find and apply valid IP addresses to hosts.

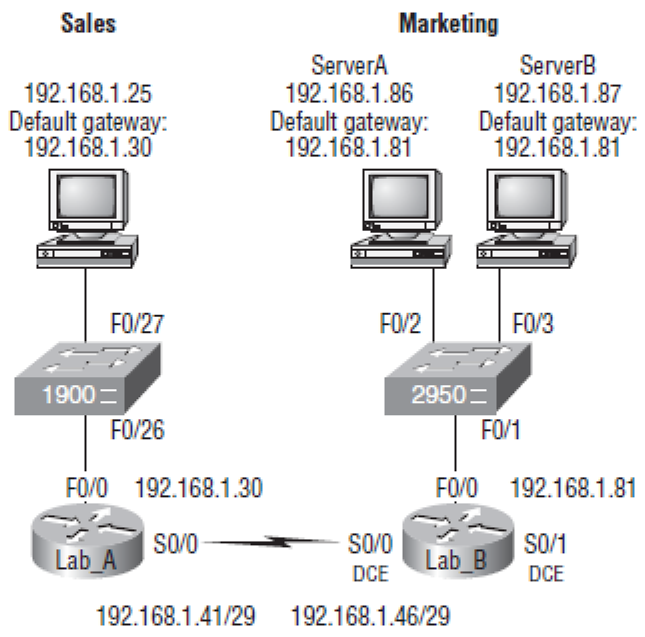


FIGURE 11.2 IP address problem 2

You need to assign a server and router IP addresses on a LAN. The subnet assigned on that segment is 192.168.20.24/29, and the router needs to be assigned the first usable address and the server the last valid host ID. What are the IP address, mask, and default gateway assigned to the server?

To answer this, you must know that a /29 is a 255.255.255.248 mask, which provides a block size of 8. The subnet is known as 24, the next subnet in a block of 8 is 32, so the broadcast address of the 24 subnet is 31, which makes the valid host range 25–30.

```
Server IP address: 192.168.20.30
Server mask: 255.255.255.248
Default gateway: 192.168.20.25 (router’s IP address)
```

As another example, let’s take a look at Figure 11.3 and solve this problem.

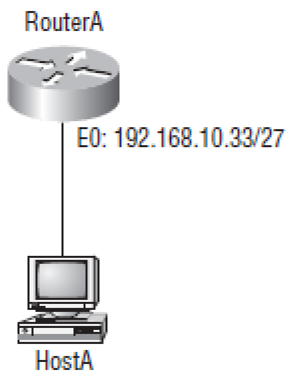


FIGURE 11.3 Find the valid host.

Look at the router's IP address on Ethernet0. What IP address, subnet mask, and valid host range could be assigned to the host? The IP address of the router's Ethernet0 is 192.168.10.33/27. As you already know, a /27 is a 224 mask with a block size of 32. The router's interface is in the 32 subnet. The next subnet is 64, so that makes the broadcast address of the 32 subnet 63 and the valid host range 33–62.

Host IP address: 192.168.10.34–62 (any address in the range except for 33, which is assigned to the router)

Mask: 255.255.255.224

Default gateway: 192.168.10.33

Figure 11.4 shows two routers with Ethernet configurations already assigned. What are the host addresses and subnet masks of hosts A and B?

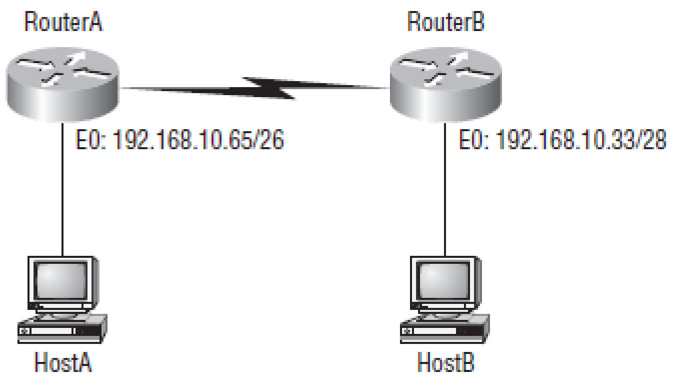


FIGURE 11.4 Find the valid host #2

RouterA has an IP address of 192.168.10.65/26, and RouterB has an IP address of

192.168.10.33/28. What are the host configurations? RouterA Ethernet0 is in the 192.168.10.64subnet, and RouterB Ethernet0 is in the 192.168.10.32 network.

Host A IP address: 192.168.10.66–126
Host A mask: 255.255.255.192
Host A default gateway: 192.168.10.65
Host B IP address: 192.168.10.34–46
Host B mask: 255.255.255.240
Host B default gateway: 192.168.10.33

Let’s try another example. Figure 11.5 shows two routers; you need to configure the S0/0 interface on RouterA. The network assigned to the serial link is 172.16.17.0/22.

What IP address can be assigned?

First, you must know that a /22 CIDR is 255.255.252.0, which makes a block size of 4 in the third octet. Since 17 is listed, the available range is 16.1 through 19.254; so, for example, the IP address S0/0 could be 172.16.18.255 since that’s within the range.

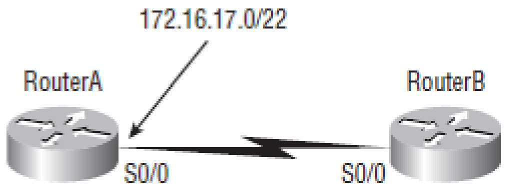


FIGURE 11.5 Find the valid host address #3

Here’s one final example. You have one Class C network ID and you need to provide one usable subnet per city while allowing enough usable host addresses for each city specified in Figure 11.6. What is your mask?

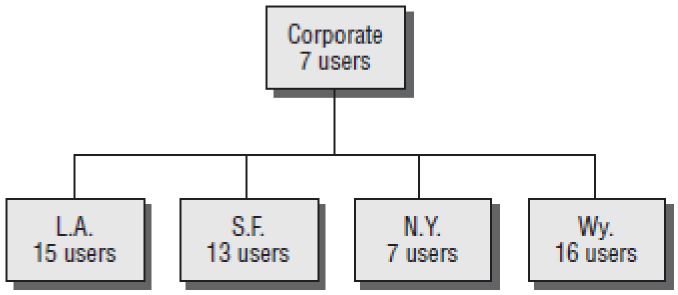


FIGURE 11.6 Find the valid subnet mask.

Actually, this is probably the easiest thing you've done all day! I count 5 subnets needed, and the Wyoming office needs 16 users (always look for the network that needs the most hosts). What block size is needed for the Wyoming office? 32. (Remember, you cannot use a block size of 16 because you always have to subtract 2!) What mask provides you with a block size of 32? 224. Bingo! This provides 8 subnets, each with 30 hosts.

Utilize netflow data

NetFlow is a technology that lets you evaluate IP traffic and understand how and where it flows. NetFlow gathers data that can be used in accounting, network monitoring, and network planning.

A flow is a one-directional stream of packets that arrives on a source interface (or sub-interface), matching a set of criteria. All packets with the same source/destination IP address, source/destination ports, protocol interface and class of service are grouped into a flow and then packets and bytes are tallied. This condenses a large amount of network information into a database called the NetFlow cache.

You create a flow by defining the criteria it gathers. Flow information tells you the following:

- Source address tells you who is originating the traffic.
- Destination address tells who is receiving the traffic.
- Ports characterize the application using the traffic.
- Class of service examines the priority of the traffic.
- The device interface tells how traffic is being used by the network device.
- Tallied packets and bytes show the amount of traffic.

A flow record defines the information that NetFlow gathers, such as packets in the flow and the types of counters gathered per flow. You can define new flow records or use the pre-defined Cisco Nexus 1000V flow record.

NetFlow Troubleshooting Commands

Use the commands listed in this section to troubleshoot NetFlow problems.

- debug logfile filename—Use this command to redirect the output of the following debug commands to a file stored in bootflash.

    - debug nfm all
    - debug sf_nf_srv all

- vemdebug netflow dump policy— Use this command to verify if the correct policy is installed on an interface on a VEM. The output of this command goes to the vemlog internal buffer. Make sure the output shows the cache type as normal, and shows the correct cache size and cache timer values.

```
Apr 14 12:20:51.504410       19   2    2  16   Debug Port 49 has 1
monitors for dir INPUT traffic IPV4
Apr 14 12:20:51.504412       20   2    2  16   Debug Flow Monitor fm1:
Apr 14 12:20:51.504413       21   2    2  16   Debug   Description:
fm1
Apr 14 12:20:51.504413       22   2    2  16   Debug   Monitor ID:
3
Apr 14 12:20:51.504413       23   2    2  16   Debug   Cache:
Apr 14 12:20:51.504414       24   2    2  16   Debug     Type:
normal
Apr 14 12:20:51.504414       25   2    2  16   Debug     Status:
allocated
Apr 14 12:20:51.504415       26   2    2  16   Debug     Size:
256 entries
Apr 14 12:20:51.504415       27   2    2  16   Debug     Inactive
Timeout:  15 secs
Apr 14 12:20:51.504416       28   2    2  16   Debug     Active
Timeout:    1800 secs
```

- vemdebug netflow dump pakstore—Use this command to dump pakstore usage for a policy on an interface. The output goes to a vemlog internal buffer. Make sure the output shows the correct monitor name and interface.

```
Apr 14 12:25:30. 29787      260   0    2  16   Debug Pak Store for
Client: fm1
Apr 14 12:25:30. 29793      266   0    2  16   Debug Pak Store for
Client: LTL49
```

- vemlog debug sfnetflow_cache all
- vemlog debug sfnetflow_config all
- vemlog debug sfnetflow_flowapi all

Use these command to enable NetFlow debugging for policy installation on the VEM. Debug messages are printed for every PDL session open, verify, and commit requests coming from the DPA.

- vemlog debug sfnetflow all

Use this command to enable packet path debugging for Netflow policies on the VEM. Debug messages are printed for every packet that hits a NetFlow policy. Use this command with caution. High traffic could result in lot of debug messages.

Use the following commands to collect information about NFM process run-time configuration errors:
- show flow internal event-history errors
- show flow internal event-history msgs
- show flow internal ddb b

- show flow internal mem-stats (to debug memory usage and leaks)

Use the following commands to collect sf_nf_srv process run-time information:

- show system internal sf_nf_srv event-history errors
- show system internal sf_nf_srv event-history msgs
- show system internal sf_nf_srv pdl detailed
- show system internal sf_nf_srv mem-stats (to debug memory usage and leaks)

Common NetFlow Problems
Common NetFlow configuration problems on the VSM can occur if you attempt to do the following:

- Use undefined records, exporters, samplers, or monitors
- Use invalid records, exporters, samplers, or monitors
- Modify records, exporters, samplers, or monitors after they are applied to an interface
- Configure a monitor on an interface which causes the VEM to run out of memory and results in a verification error

In addition, a configuration error can occur if there is a mismatch between the UDP port configured on the exporter and the port NetFlow Collector has listening turned on.

Troubleshoot and Resolve Spanning Tree operation issues

STP Troubleshooting
Although this section helps you prepare to troubleshoot STP problems in real networks, the main goal for this section is to prepare you to answer STP questions on the CCNA exams.

STP questions tend to intimidate many test takers. STP uses many rules, with tiebreakers in case one rule ends with a tie. Without much experience with STP, people tend to distrust their own answers. Also, even those of us with networking jobs already probably do not troubleshoot STP very often, because STP runs by default and works well using default configuration settings in medium to small networks, so engineers seldom need to troubleshoot STP problems. So, STP begs for a good troubleshooting strategy before examining a complex STP question.

It also makes some practical suggestions about how to go about answering exam questions such as "which switch is the root switch."

Determining the Root Switch
Determining the STP root switch is easy if you know all the switches' BIDs: Just pick the lowest value. If the question lists the priority and MAC address separately, as is common in some show command output, pick the switch with the lowest priority, or in the case of a tie, pick the lower MAC address value.
And just to be extra clear, STP does not have nor need a tiebreaker for electing the root switch. The BID uses a switch universal MAC address as the last 48 bits of the BID. These MAC addresses are unique in the universe, so there should never be identical BIDs, with no-need for a tiebreaker.

For the exam, a question that asks about the root switch might not be so simple as listing a bunch of BIDs and asking you which one is "best." A more likely question is a sim question in which you have to do any show commands you like or a multiple choice question that lists the output from only one or two commands. Then you have to apply the STP algorithm to figure out the rest.

When faced with an exam question using a simulator, or just the output in an exhibit, use a simple strategy of ruling out switches, as follows:

- **Step 1**. Begin with a list or diagram of switches, and consider all as possible root switches.
- **Step 2.** Rule out any switches that have an RP (show spanning-tree, show spanning-tree root), because root switches do not have a RP.
- **Step 3**. Always try show spanning-tree, because it identifies the local switch as root directly: "This switch is the root" on the fifth line of output.
- **Step 4**. Always try show spanning-tree root , because it identifies the local switch as root indirectly: The RP column is empty if the local switch is the root.
- **Step 5.** When using a sim, rather than try switches randomly, chase the RPs. For example, if starting with SW1, and SW1's G0/1 is an RP, next try the switch on the other end of SW1's G0/1 port.
- **Step 6**. When using a Sim, using show spanning-tree vlan x on a few switches, and recording the root switch, RP, and DP ports can quickly show you most STP facts. Use that strategy if available.

The one step in this list that most people ignore is the idea of ruling out switches that have an RP. Root switches do not have an RP, so any switch with an RP can be ruled out as not being the root switch for that VLAN. Example 11.1 shows two commands on switch SW2 in some LAN that confirms that SW2 has an RP and is therefore not the root switch.

Example 11.1 Ruling Out Switches as Root Based on Having a Root Port

```
SW2# show spanning-tree vlan 20 root
Root Hello Max Fwd
Vlan Root ID Cost Time Age Dly Root Port
---------------- ------------------- --------- ----- --- --- ------------
VLAN0020 32788 1833.9d7b.0e80 4 2 20 15 Gi0/2
SW2# show spanning-tree vlan 20
VLAN0020
Spanning tree enabled protocol ieee
Root ID Priority 32788
Address 1833.9d7b.0e80
Cost 4
Port 26 (GigabitEthernet0/2)
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Bridge ID Priority 32788 (priority 32768 sys-id-ext 20)
Address 1833.9d7b.1380
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 15 sec
```

```
Interface Role Sts Cost Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi0/1 Desg FWD 4 128.25 P2p
Gi0/2 Root FWD 4 128.26 P2p
```

Both commands identify SW2's G0/2 port as its RP, so if you follow the suggestions, the next switch to try in a sim question would be the switch on the other end of SW2's G0/2 interface.

Determining the Root Port on Non-root Switches

Determining the RP of a switch when show command output is available is relatively easy.

The challenge comes more when an exam question makes you think through how the switches choose the RP based on the root cost of each path to the root switch with some tiebreakers as necessary.

As a review, each non-root switch has one, and only one, RP for a VLAN. To choose its RP, a switch listens for incoming hello bridge protocol data units (BPDU). For each received hello, the switch adds the cost listed in the hello BPDU to the cost of the incoming interface (the interface on which the hello was received). That total is the root cost over that path. The lowest root cost wins, and the local switch uses its local port that is part of the least root cost path as its root port.

For the exam, if the question has a diagram of the LAN, most humans work better with a slightly different way to look at the problem. Instead of thinking about hello messages and so on, approach the question as this: the sum of all outgoing port costs between the non-root switch and the root. Repeating a familiar example, with a twist, Figure 11.7 shows the calculation of the root cost. Note that SW3's Gi0/3 port has yet again had its cost configured to a different value.

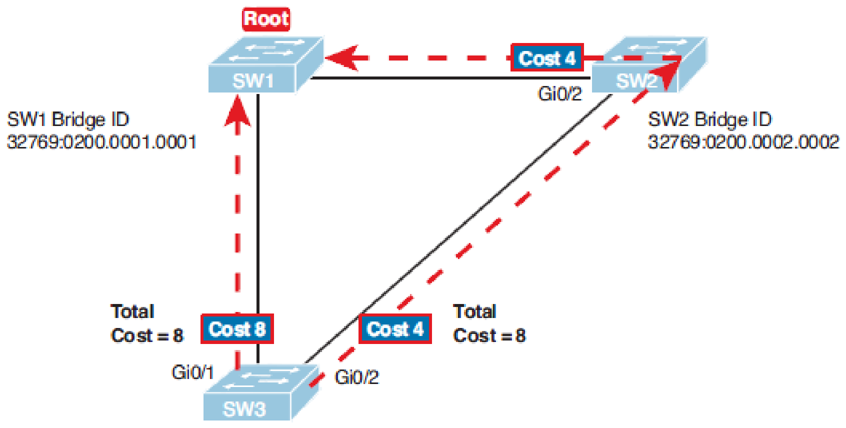


Figure 11.7 SW3's Root Cost Calculation Ends in a Tie

STP Tiebreakers When Choosing the Root Port
The figure shows the easier process of adding the STP costs of the outgoing interfaces over each from SW3, a non-root, to SW1, the root. It also shows a tie (on purpose), to talk about the tiebreakers.

When a switch chooses its root port, the first choice is to choose the local port that is part of the least root cost path. When those costs tie, the switch picks the port connected to the neighbor with the lowest BID. This tiebreaker usually breaks the tie, but not always. So, for completeness, the three tiebreakers are, in the order a switch uses them, as follows:

1. Choose based on the lowest neighbor bridge ID.
2. Choose based on the lowest neighbor port priority.
3. Choose based on the lowest neighbor internal port number.

The next tiebreaker is a configurable option: the neighboring switch's port priority on each neighboring switch interface. Cisco switch ports default to a setting of 128, with a range of values from 0 through 255, with lower being better (as usual). In this example, the network engineer has set SW2's F0/16 interface with the spanning-tree vlan 10 port-priority 112 command. SW1 learns that the neighbor has a port priority of 112 on the top link and 128 on the bottom, so SW1 uses its top (F0/14) interface as the root port.

If the port priority ties, which it often does due to the default values, STP relies on an internal port numbering on the neighbor. Cisco switches assign an internal integer to identify each interface on the switch. The non-root looks for the neighbor's lowest internal port number (as listed in the hello messages) and chooses its RP based on the lower number.

Cisco switches use an obvious numbering, with Fa0/1 having the lowest number, then Fa0/2, then Fa0/3, and so on. So, in Figure SW2, SW2's Fa0/16 would have a lower internal port number than Fa0/17; SW1 would learn those numbers in the hello; and SW1 would use its Fa0/14 port as its RP.

Suggestions for Attacking Root Port Problems on the Exam
Exam questions that make you think about the RP can be easy if you know where to look and the output of a few key commands are available. However, the more conceptual the question, the more you have to calculate the root cost over each path, correlate that to different show commands, and put the ideas together. The following list makes a few suggestions about how to approach STP problems on the exam:

1. If available, look at the show spanning-tree and show spanning-tree root commands. These both list the root port, and the first of these also lists the root cost.
2. The show spanning-tree command list cost in two places: the root cost at the top, in the section about the root switch; and the interface cost, at the bottom, in the per-interface section. Be careful, though; the cost at the bottom is the interface cost, not the root cost !
3. For problems where you have to calculate a switch's root cost:

a) Memorize the default cost values: 100 for 10 Mbps, 19 for 100 Mbps, 4 for 1 Gbps, and 2 for 10 Gbps.
b) Look for any evidence of the spanning-tree cost configuration command on an interface, because it overrides the default cost. Do not assume default costs are used.

c) When you know a default cost is used, if you can, check the current actual speed as well. Cisco switches choose STP cost defaults based on the current speed, not the maximum speed.

Troubleshoot and Resolve routing issues

Troubleshoot routing issues

The best troubleshooting tools are show and debug commands, specifically show ip protocols and various routing protocol debugging commands. Let's take a look.

First, the show ip protocols command will show you the routing protocols configured on your router. Hopefully you have only one. Here is an example:

```
Router#sh ip protocols
Routing Protocol is "rip"
Outgoing update filter list for all interfaces is not set
Incoming update filter list for all interfaces is not set
Sending updates every 30 seconds, next due in 27 seconds
Invalid after 180 seconds, hold down 180, flushed after 240
Redistributing: rip
Default version control: send version 2, receive version 2
Interface Send Recv Triggered RIP Key-chain
FastEthernet0/1 2 2
Serial0/0/1 2 2
Automatic network summarization is not in effect
Maximum path: 4
Routing for Networks:
10.0.0.0
Passive Interface(s):
FastEthernet0/0
Serial0/0/0
```

Routing Information Sources:

```
Gateway Distance Last Update
10.1.11.2 120 00:00:00
10.1.5.1 120 00:00:02
Distance: (default is 120)
```

The above router output shows the router is running RIPv2, the interfaces participating in the routing process, the next hop gateways (neighbors) and the administrative distance. Also, you can see that the maximum path is 4, which means that RIP will load-balance across four equal cost links by default.

There are dozens of debugging commands you can use, and for RIP, the debug ip rip command is the best tool for debugging RIP routing.

```
*Mar 17 19:34:00.123: RIP: sending v2 update to 224.0.0.9 via
Serial0/0/1 (10.1.5.2)
*Mar 17 19:34:00.123: RIP: build update entries
*Mar 17 19:34:00.123: 10.1.10.0/24 via 0.0.0.0, metric 1, tag 0
*Mar 17 19:34:00.123: 10.1.11.0/24 via 0.0.0.0, metric 1, tag 0
*Mar 17 19:34:00.123: 10.1.12.0/24 via 0.0.0.0, metric 2, tag 0col
*Mar 17 19:34:03.795: RIP: received v2 update from 10.1.5.1 on
Serial0/0/1
```

The above debug output shows we are running RIPv2, with multicast address 224.0.0.9. For EIGRP I am going to show you two commands. The debug eigrp packet command and the debug ip eigrp notification command:

```
Router#debug eigrp packet
EIGRP Packets debugging is on
(UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB,
SIAQUERY, SIAREPLY)
*Mar 21 23:17:35.050: EIGRP: Sending HELLO on FastEthernet0/1
*Mar 21 23:17:35.050: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
*Mar 21 23:17:35.270: EIGRP: Received HELLO on Serial0/1/0 nbr 10.1.4.2
*Mar 21 23:17:35.270: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:35.294: EIGRP: Received HELLO on Serial0/0/0 nbr 10.1.2.2
*Mar 21 23:17:35.294: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:38.014: EIGRP: Received HELLO on Serial0/2/0 nbr 10.1.5.2
*Mar 21 23:17:38.014: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
```

The EIGRP 224.0.0.10 multicast is sent out every five seconds, and the Hello packets are sent out of every active interface, as well as all the interfaces that we have neighbors connected to. Did you notice the AS number is provided in the update? This is because if a neighbor doesn’t have the same AS number, the Hello update would just be discarded.

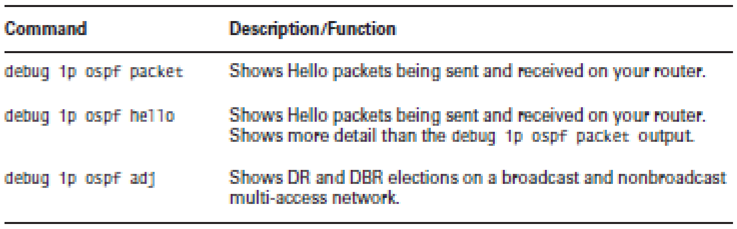
The debug ip eigrp notification command (called debug ip eigrp events on pre
12.4 routers) shouldn’t show you anything at all! That’s right—the only time you’ll see output from this command is if there’s a problem on your network, or you’ve added or deleted a network from a router in your internetwork. Since I have a problem-free network, I’m going to shut down an interface on my router in order to see some output:

```
Router(config)#int f0/1
Router(config-if)#shut
*Mar 21 23:25:43.506: IP-EIGRP(Default-IP-Routing-Table:10): Callback:
route_adjust FastEthernet0/1
*Mar 21 23:25:43.506: IP-EIGRP: Callback: ignored connected AS 0 10.1.1.0/24
*Mar 21 23:25:43.506: into: eigrp AS 10
*Mar 21 23:25:43.506: IP-EIGRP(Default-IP-Routing-Table:10): Callback:
callbackup_routes 10.1.1.0/24
```

```
Corp(config-if)#n
*Mar 21 23:25:45.506: %LINK-5-CHANGED: Interface FastEthernet0/1,
changed state to administratively down
*Mar 21 23:25:46.506: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/1, changed state to down
Router(config-if)#no shut
Router(config-if)#^Z
*Mar 21 23:25:49.570: %LINK-3-UPDOWN: Interface FastEthernet0/1,
changed state to up
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:10): Callback:
lostroute 10.1.1.0/24
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:0): Callback:
redist connected (config change) FastEthernet0/1
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:0): Callback:
redist connected (config change) Serial0/0/0
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:0): Callback:
redist connected (config change) Serial0/0/1
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:0): Callback:
redist connected (config change) Serial0/1/0
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:0): Callback:
redist connected (config change) Serial0/2/0
*Mar 21 23:25:49.570: IP-EIGRP(Default-IP-Routing-Table:10): Callback:
route_adjust FastEthernet0/1
```

Debugging is a great tool for any protocol, so let's take a look in Table 1.1 at a few debugging commands for troubleshooting OSPF.

Table 11.1: Debugging Commands for Troubleshooting OSPF



Command	Description/Function
`debug ip ospf packet`	Shows Hello packets being sent and received on your router.
`debug ip ospf hello`	Shows Hello packets being sent and received on your router. Shows more detail than the `debug ip ospf packet` output.
`debug ip ospf adj`	Shows DR and DBR elections on a broadcast and nonbroadcast multi-access network.

I'll start by showing you the output from the router I have, using the debug ip ospf packet command:

```
Router#debug ip ospf packet
OSPF packet debugging is on
*Mar 23 01:20:42.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
*Mar 23 01:20:45.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/1
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.4
aid:0.0.0.0 chk:8074 aut:0 auk: from Serial0/2/0
*Mar 23 01:20:52.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
```

```
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
*Mar 23 01:20:55.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
```

In the above output, you can see that the router is both sending and receiving Hello packets every 10 seconds from neighbor (adjacent) routers. The next command will provide the same information, but with more detail. For example, you can see the multicast address used (224.0.0.5) and the area:

```
Router#debug ip ospf hello
*Mar 23 01:18:41.103: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/1/0 from 10.1.4.1
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
FastEthernet0/1 from 10.1.1.1
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/0 from 10.1.2.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/2/0 from 10.1.5.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/1 from 10.1.3.1
*Mar 23 01:18:42.199: OSPF: Rcv hello from 172.16.10.3 area 0 from
Serial0/1/0 10.1.4.2
*Mar 23 01:18:42.199: OSPF: End of hello processing
*Mar 23 01:18:45.519: OSPF: Rcv hello from 172.16.10.2 area 0 from
Serial0/0/0 10.1.2.2
*Mar 23 01:18:45.519: OSPF: End of hello processing
```

The last debug command I'm going show you is the debug ip ospf adj command that will show us elections as they occur on broadcast and nonbroadcast multi-access networks:

```
Router#debug ip ospf adj
OSPF adjacency events debugging is on
*Mar 23 01:24:34.823: OSPF: Interface FastEthernet0/1 going Down
*Mar 23 01:24:34.823: OSPF: 172.16.10.1 address 10.1.1.1 on
FastEthernet0/1 is dead, state DOWN
*Mar 23 01:24:34.823: OSPF: Neighbor change Event on interface
FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: DR/BDR election on FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: DR: none BDR: none
*Mar 23 01:24:34.823: OSPF: Flush network LSA immediately
*Mar 23 01:24:34.823: OSPF: Remember old DR 172.16.10.1 (id)
*Mar 23 01:24:35.323: OSPF: We are not DR to build Net Lsa for
interface FastEthernet0/1
```

```
*Mar 23 01:24:35.323: OSPF: Build router LSA for area 0, router ID
172.16.10.1, seq 0x80000006
*Mar 23 01:24:35.347: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 148 LSA count 1
*Mar 23 01:24:40.703: OSPF: Interface FastEthernet0/1 going Up
*Mar 23 01:24:41.203: OSPF: Build router LSA for area 0, router ID
172.16.10.1, seq 0x80000007
*Mar 23 01:24:41.231: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 160 LSA count 1
```

Troubleshoot and Resolve OSPF problems

Configure, verify, and troubleshoot OSPF
Open Shortest Path First (OSPF) is an open standard routing protocol that's been implemented by a wide variety of network vendors, including Cisco. If you have multiple routers and not all of them are Cisco (what!), then you can't use EIGRP, can you? So, your remaining CCNA objective options are basically RIP, RIPv2, and OSPF.

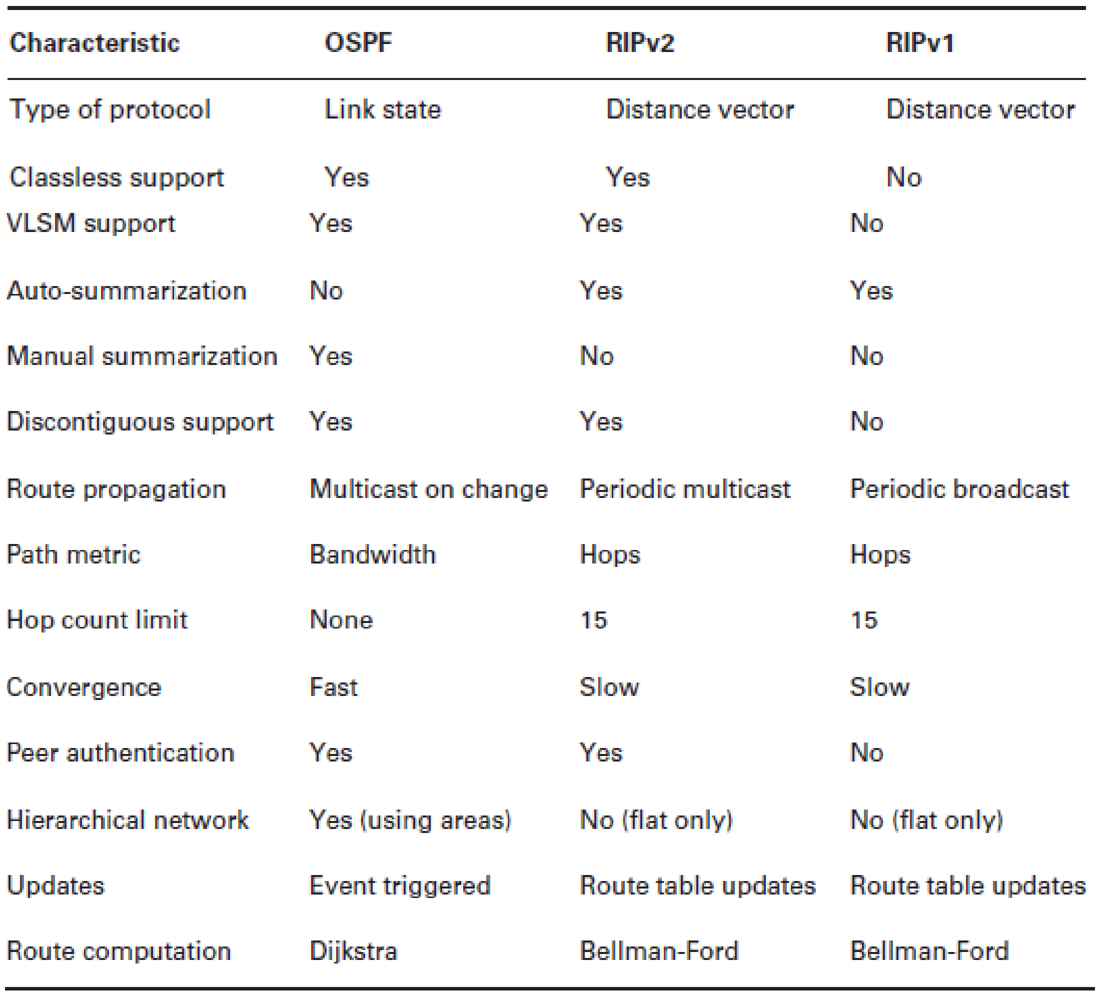
OSPF works by using the Dijkstra algorithm. First, a shortest path tree is constructed, and then the routing table is populated with the resulting best paths. OSPF converges quickly, although perhaps not as quickly as EIGRP, and it supports multiple, equal-cost routes to the same destination. Like EIGRP, it does support both IP and IPv6 routed protocols.

OSPF provides the following features:

- Consists of areas and autonomous systems
- Minimizes routing update traffic
- Allows scalability
- Supports VLSM/CIDR
- Has unlimited hop count
- Allows multi-vendor deployment (open standard)

OSPF is the first link-state routing protocol that most people are introduced to, so it's useful to see how it compares to more traditional distance-vector protocols such as RIPv2 and RIPv1. Table 11.2 gives you a comparison of these three protocols.

TABLE 11. 2 OSPF and RIP comparison



Characteristic	OSPF	RIPv2	RIPv1
Type of protocol	Link state	Distance vector	Distance vector
Classless support	Yes	Yes	No
VLSM support	Yes	Yes	No
Auto-summarization	No	Yes	Yes
Manual summarization	Yes	No	No
Discontiguous support	Yes	Yes	No
Route propagation	Multicast on change	Periodic multicast	Periodic broadcast
Path metric	Bandwidth	Hops	Hops
Hop count limit	None	15	15
Convergence	Fast	Slow	Slow
Peer authentication	Yes	Yes	No
Hierarchical network	Yes (using areas)	No (flat only)	No (flat only)
Updates	Event triggered	Route table updates	Route table updates
Route computation	Dijkstra	Bellman-Ford	Bellman-Ford

OSPF has many features beyond the few I've listed in Table 11.8, and all of them contribute to a fast, scalable, and robust protocol that can be actively deployed in thousands of production networks.

OSPF is supposed to be designed in a hierarchical fashion, which basically means that you can separate the larger internetwork into smaller internetworks called areas. This is the best design for OSPF.

The following are reasons for creating OSPF in a hierarchical design:

- To decrease routing overhead
- To speed up convergence
- To confine network instability to single areas of the network

This does not make configuring OSPF easier, but more elaborate and difficult.
Figure 11.8 shows a typical OSPF simple design. Notice how each router connects to the backbone—called area 0, or the backbone area. OSPF must have an area 0, and all other areas should connect to this area.Routers that connect other areas to the backbone area within an AS are called Area Border Routers (ABRs). Still, at least one interface of the ABR must be in area 0.

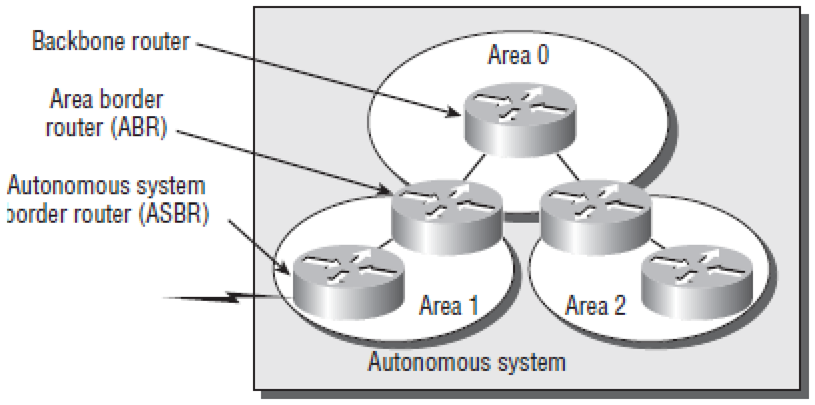


FIGURE 11.8 OSPF design example

OSPF runs inside an autonomous system, but it can also connect multiple autonomous systems. The router that connects this ASs is called an Autonomous System Boundary Router (ASBR).
Ideally, you would create other areas of networks to help keep route updates to a minimum and to keep problems from propagating throughout the network.

Configuring OSPF Areas
After identifying the OSPF process, you need to identify the interfaces that you want to activate OSPF communications on as well as the area in which each resides. This will also configure the networks you're going to advertise to others. OSPF uses wildcards in the configuration—which are also used in access-list configurations.
Here's an OSPF basic configuration example for you:

```
Lab_A#config t
Lab_A(config)#router ospf 1
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area ?
<0-4294967295> OSPF area ID as a decimal value
A.B.C.D OSPF area ID in IP address format
Lab_A(config-router)#network 10.0.0.0 0.255.255.255
area 0
```

Remember, the OSPF Process ID number is irrelevant. It can be the same on every router on the network, or it can be different—doesn't matter. It's locally significant and just enables the OSPF routing on the router.

We're going to start by taking a quick look at the routing table of the Corp router:
So, let's issue a show ip route command on the Corp router:

```
10.0.0.0/24 is subnetted, 12 subnets
O 10.1.11.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
O 10.1.10.0 [110/65] via 10.1.5.2, 00:01:31, Serial0/2/0
O 10.1.9.0 [110/74] via 10.1.4.2, 00:01:31, Serial0/1/0
O 10.1.8.0 [110/65] via 10.1.4.2, 00:01:31, Serial0/1/0
```

```
O 10.1.12.0 [110/66] via 10.1.5.2, 00:01:31, Serial0/2/0
C 10.1.3.0 is directly connected, Serial0/0/1
C 10.1.2.0 is directly connected, Serial0/0/0
C 10.1.1.0 is directly connected, FastEthernet0/1
O 10.1.7.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
O 10.1.6.0 [110/74] via 10.1.3.2, 00:01:32, Serial0/0/1
[110/74] via 10.1.2.2, 00:01:32, Serial0/0/0
C 10.1.5.0 is directly connected, Serial0/2/0
C 10.1.4.0 is directly connected, Serial0/1/0
```

The Corp router shows the found routes for all 12 of our networks, with the O representing OSPF internal routes (the Cs are obviously our directly connected networks). It also found the dual routes to networks 10.1.6.0 and 10.1.7.0. I removed the bandwidth and delay commands from under the interface, so the defaults are being used to determine the metric. But remember, OSPF only uses bandwidth to determine the best path to a network.

The show ip ospf Command

The show ip ospf command is used to display OSPF information for one or all OSPF processes running on the router. Information contained therein includes the Router ID, area information, SPF statistics, and LSA timer information. Let’s check out the output from the Corp router:

```
Corp#sh ip ospf
Routing Process "ospf 132" with ID 10.1.5.1
Start time: 04:32:04.116, Time elapsed: 01:27:10.156
Supports only single TOS(TOS0) routes
Supports opaque LSA
Supports Link-local Signaling (LLS)
Supports area transit capability
Router is not originating router-LSAs with maximum metric
Initial SPF schedule delay 5000 msecs
Minimum hold time between two consecutive SPFs 10000 msecs
Maximum wait time between two consecutive SPFs 10000 msecs
Incremental-SPF disabled
Minimum LSA interval 5 secs
Minimum LSA arrival 1000 msecs
LSA group pacing timer 240 secs
Interface flood pacing timer 33 msecs
Retransmission pacing timer 66 msecs
Number of external LSA 0. Checksum Sum 0x000000
Number of opaque AS LSA 0. Checksum Sum 0x000000
Number of DCbitless external and opaque AS LSA 0
Number of DoNotAge external and opaque AS LSA 0
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
Number of areas transit capable is 0
External flood list length 0
Area BACKBONE(0)
Number of interfaces in this area is 5
```

```
Area has no authentication
SPF algorithm last executed 00:14:52.220 ago
SPF algorithm executed 14 times
Area ranges are
Number of LSA 6. Checksum Sum 0x03C06F
Number of opaque link LSA 0. Checksum Sum 0x000000

Number of DCbitless LSA 0
Number of indication LSA 0
Number of DoNotAge LSA 0
Flood list length 0
```

Notice the Router ID (RID) of 10.1.5.1, which is the highest IP address configured on the router.

The show ip ospf database Command
Using the show ip ospf database command will give you information about the number of routers in the internetwork (AS) plus the neighboring router's ID (this is the topology database I mentioned earlier). Unlike the show ip eigrp topology command, this command shows the "OSPF routers," not each and every link in the AS as EIGRP does.

The output is broken down by area. Here's a sample output, again from Corp:

```
Corp#sh ip ospf database
OSPF Router with ID (10.1.5.1) (Process ID 132)
Router Link States (Area 0)
Link ID ADV Router Age Seq# Checksum Link count
10.1.5.1 10.1.5.1 72 0x80000002 0x00F2CA 9
10.1.7.1 10.1.7.1 83 0x80000004 0x009197 6
10.1.9.1 10.1.9.1 73 0x80000001 0x00DA1C 4
10.1.11.1 10.1.11.1 67 0x80000005 0x00666A 4
10.1.12.1 10.1.12.1 67 0x80000004 0x007631 2
Net Link States (Area 0)
Link ID ADV Router Age Seq# Checksum
10.1.11.2 10.1.12.1 68 0x80000001 0x00A337
```

You can see all five routers and the RID of each router (the highest IP address on each router). The router output shows the link ID—remember that an interface is also a link—and the RID of the router on that link under the ADV router, or advertising router.

The show ip ospf interface Command
The show ip ospf interface command displays all interface-related OSPF information.
Data is displayed about OSPF information for all interfaces or for specified interfaces. (I'll bold some of the important things.)

```
Corp#sh ip ospf interface f0/1
FastEthernet0/1 is up, line protocol is up
Internet Address 10.1.1.1/24, Area 0
```

```
Process ID 132, Router ID 10.1.5.1, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DR, Priority 1
Designated Router (ID) 10.1.5.1, Interface address 10.1.1.1
No backup designated router on this network
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
oob-resync timeout 40
Hello due in 00:00:01
Supports Link-local Signaling (LLS)
Index 1/1, flood queue length 0
Next 0x0(0)/0x0(0)
Last flood scan length is 0, maximum is 0
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 0, Adjacent neighbor count is 0
Suppress hello for 0 neighbor(s)
```

The following information is displayed by this command:

- Interface IP address
- Area assignment
- Process ID
- Router ID
- Network type
- Cost
- Priority
- DR/BDR election information (if applicable)
- Hello and Dead timer intervals
- Adjacent neighbor information

The reason I used the show ip ospf interface f0/1 command is that I knew that there would be a designated router elected on the FastEthernet broadcast multi-access network.

We'll get into DR and DBR elections in detail in a minute.

The show ip ospf neighbor Command

The show ip ospf neighbor command is super-useful because it summarizes the pertinent OSPF information regarding neighbors and the adjacency state. If a DR or BDR exists, that information will also be displayed.

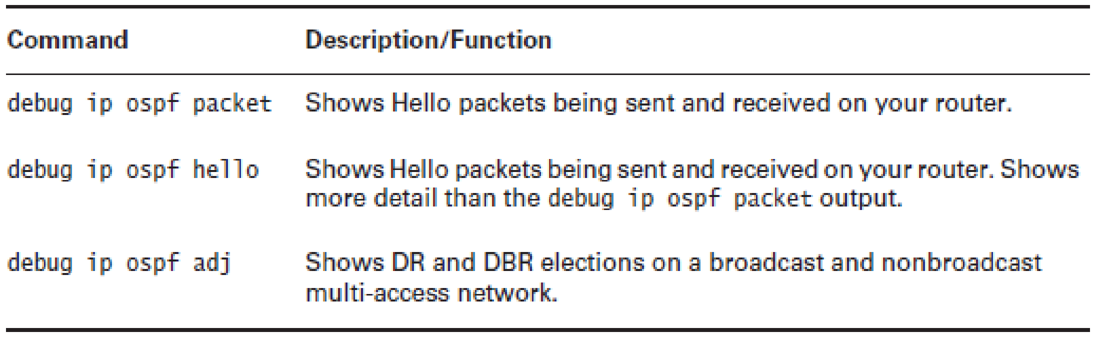
Since there's an Ethernet link (broadcast multi-access) on the Corp router, there's going to be an election to determine who will be the designated router and who will be the non-designated router. You can see that the 871W became the designated router, and it won because it had the highest IP address on the network. You can change this, but that's the default.

The reason that the Corp connections to R1, R2, and R3 don't have a DR or BDR listed in the output is that by default, elections don't happen on point-to-point links. But you can see that the Corp router is fully adjacent to all three routers (and on both connections to R1) from its output.

Debugging OSPF

Debugging is a great tool for any protocol, so let's take a look in Table 11.3 at a few debugging commands for troubleshooting OSPF.

TABLE 11.3 Debugging Commands for Troubleshooting OSPF



Command	Description/Function
`debug ip ospf packet`	Shows Hello packets being sent and received on your router.
`debug ip ospf hello`	Shows Hello packets being sent and received on your router. Shows more detail than the debug `ip ospf packet` output.
`debug ip ospf adj`	Shows DR and DBR elections on a broadcast and nonbroadcast multi-access network.

I'll start by showing you the output from the Corp router I got using the debug ip ospf packet command:

```
Corp#debug ip ospf packet
OSPF packet debugging is on
*Mar 23 01:20:42.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
Corp#
*Mar 23 01:20:45.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/1
*Mar 23 01:20:45.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.4
aid:0.0.0.0 chk:8074 aut:0 auk: from Serial0/2/0
*Mar 23 01:20:52.199: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.3
aid:0.0.0.0 chk:8075 aut:0 auk: from Serial0/1/0
*Mar 23 01:20:55.507: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/0
*Mar 23 01:20:55.527: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.2
aid:0.0.0.0 chk:8076 aut:0 auk: from Serial0/0/1
*Mar 23 01:20:55.531: OSPF: rcv. v:2 t:1 l:48 rid:172.16.10.4
aid:0.0.0.0 chk:8074 aut:0 auk: from Serial0/2/0
```

In the preceding output, you can see that our router is both sending and receiving Hello packets every 10 seconds from neighbor (adjacent) routers. The next command will provide us with the same information, but with more detail. For example, you can see the multicast address used (224.0.0.5) and the area:

```
Corp#debug ip ospf hello
*Mar 23 01:18:41.103: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/1/0 from 10.1.4.1
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
FastEthernet0/1 from 10.1.1.1
```

```
*Mar 23 01:18:41.607: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/0 from 10.1.2.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/2/0 from 10.1.5.1
*Mar 23 01:18:41.611: OSPF: Send hello to 224.0.0.5 area 0 on
Serial0/0/1 from 10.1.3.1
*Mar 23 01:18:42.199: OSPF: Rcv hello from 172.16.10.3 area 0 from
Serial0/1/0 10.1.4.2
*Mar 23 01:18:42.199: OSPF: End of hello processing
*Mar 23 01:18:45.519: OSPF: Rcv hello from 172.16.10.2 area 0 from
Serial0/0/0 10.1.2.2
*Mar 23 01:18:45.519: OSPF: End of hello processing
*Mar 23 01:18:45.543: OSPF: Rcv hello from 172.16.10.2 area 0 from
Serial0/0/1 10.1.3.2
*Mar 23 01:18:45.543: OSPF: End of hello processing
*Mar 23 01:18:45.543: OSPF: Rcv hello from 172.16.10.4 area 0 from
Serial0/2/0 10.1.5.2
*Mar 23 01:18:45.543: OSPF: End of hello processing
```

The last debug command I'm going show you is the debug ip ospf adj command, which will show you elections as they occur on broadcast and non-broadcast multi-access networks:

```
Corp#debug ip ospf adj
OSPF adjacency events debugging is on
*Mar 23 01:24:34.823: OSPF: Interface FastEthernet0/1 going Down
*Mar 23 01:24:34.823: OSPF: 172.16.10.1 address 10.1.1.1 on
FastEthernet0/1 is dead, state DOWN
*Mar 23 01:24:34.823: OSPF: Neighbor change Event on interface
FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: DR/BDR election on FastEthernet0/1
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect BDR 0.0.0.0
*Mar 23 01:24:34.823: OSPF: Elect DR 0.0.0.0
*Mar 23 01:24:34.823: DR: none BDR: none
*Mar 23 01:24:34.823: OSPF: Flush network LSA immediately
*Mar 23 01:24:34.823: OSPF: Remember old DR 172.16.10.1 (id)
*Mar 23 01:24:35.323: OSPF: We are not DR to build Net Lsa for
interface FastEthernet0/1
*Mar 23 01:24:35.323: OSPF: Build router LSA for area 0, router ID
172.16.10.1, seq 0x80000006
*Mar 23 01:24:35.347: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 148 LSA count 1
*Mar 23 01:24:40.703: OSPF: Interface FastEthernet0/1 going Up
*Mar 23 01:24:41.203: OSPF: Build router LSA for area 0, router ID
172.16.10.1, seq 0x80000007
*Mar 23 01:24:41.231: OSPF: Rcv LS UPD from 172.16.10.2 on Serial0/0/1
length 160 LSA count 1
```

Troubleshoot and Resolve EIGRP problems

Configure, verify, and troubleshoot EIGRP
There are two modes from which EIGRP commands are entered: router configuration mode and interface configuration mode. Router configuration mode enables the protocol, determines which networks will run EIGRP, and sets global characteristics. Interface configuration mode allows customization of summaries, metrics, timers, and bandwidth.

To start an EIGRP session on a router, use the router eigrp command followed by the autonomous system number of your network. You then enter the network numbers connected to the router using the network command followed by the network number. Let's look at an example of enabling EIGRP for autonomous system 20 on a router connected to two networks, with the network numbers being 10.3.1.0/24 and 172.16.10.0/24:

```
Router#config t
Router(config)#router eigrp 20
Router(config-router)#network 172.16.0.0
Router(config-router)#network 10.0.0.0
```

Remember—as with RIP, you use the classful network address, which is all subnet and host bits turned off. Understand that the AS number is irrelevant—that is, as long as all routers use the same number! You can use any number from 1 to 65,535.

Verifying EIGRP
There are several commands that can be used on a router to help you troubleshoot and verify the EIGRP configuration. Table 11.4 contains all of the most important commands that are used in conjunction with verifying EIGRP operation and offers a brief description of what each command does.

TABLE 11. 4 EIGRP Troubleshooting Commands

Command	Description/Function
`show ip route`	Shows the entire routing table
`show ip route eigrp`	Shows only EIGRP entries in the routing table
`show ip eigrp neighbors`	Shows all EIGRP neighbors
`show ip eigrp topology`	Shows entries in the EIGRP topology table
`debug eigrp packet`	Shows Hello packets sent/received between adjacent routers
`Debug ip eigrp notification`	Shows EIGRP changes and updates as they occur on your network

I'll demonstrate how you would use the commands in Table 11.10 by using them on our internetwork that we just configured—not including the dis-contiguous network example.

```
The following router output is from the Corp router in the example:
Corp#sh ip route
10.0.0.0/24 is subnetted, 12 subnets
D 10.1.11.0 [90/2172416] via 10.1.5.2, 00:01:05, Serial0/2/0
D 10.1.10.0 [90/2195456] via 10.1.5.2, 00:01:05, Serial0/2/0
D 10.1.9.0 [90/2195456] via 10.1.4.2, 00:01:05, Serial0/1/0
D 10.1.8.0 [90/2195456] via 10.1.4.2, 00:01:05, Serial0/1/0
D 10.1.12.0 [90/2172416] via 10.1.5.2, 00:01:05, Serial0/2/0
C 10.1.3.0 is directly connected, Serial0/0/1
C 10.1.2.0 is directly connected, Serial0/0/0
C 10.1.1.0 is directly connected, FastEthernet0/1
D 10.1.7.0 [90/2195456] via 10.1.2.2, 00:01:06, Serial0/0/0
D 10.1.6.0 [90/2195456] via 10.1.2.2, 00:01:06, Serial0/0/0
C 10.1.5.0 is directly connected, Serial0/2/0
C 10.1.4.0 is directly connected, Serial0/1/0
```

You can see that all routes are there in the routing table (10.1.3.0 shows that it's directly connected again), and we have only one link to networks 10.1.6.0 and 10.1.7.0! Notice that EIGRP routes are indicated with simply a D designation (DUAL) and that the default AD of these routes is 90. This represents internal EIGRP routes. Let's take a look at the R1 router table now that we've changed the metrics:

```
R1#sh ip route
10.0.0.0/24 is subnetted, 12 subnets
D 10.1.11.0 [90/2684416] via 10.1.2.1, 00:00:09, Serial0/0/0
D 10.1.10.0 [90/2707456] via 10.1.2.1, 00:00:09, Serial0/0/0
D 10.1.9.0 [90/2707456] via 10.1.2.1, 00:00:09, Serial0/0/0
D 10.1.8.0 [90/2707456] via 10.1.2.1, 00:00:09, Serial0/0/0
D 10.1.12.0 [90/2684416] via 10.1.2.1, 00:00:09, Serial0/0/0
```

```
C 10.1.3.0 is directly connected, Serial0/0/1
C 10.1.2.0 is directly connected, Serial0/0/0
D 10.1.1.0 [90/2172416] via 10.1.2.1, 00:00:09, Serial0/0/0
C 10.1.7.0 is directly connected, FastEthernet0/1
C 10.1.6.0 is directly connected, FastEthernet0/0
D 10.1.5.0 [90/2681856] via 10.1.2.1, 00:00:09, Serial0/0/0
D 10.1.4.0 [90/2681856] via 10.1.2.1, 00:00:09, Serial0/0/0
```

Now we have only one route to each remote network, and the 10.1.3.0 network is our backup link. Obviously, it would be better if we could use both links at the same time, but in my example, I made the 10.1.3.0 network a backup link.

Let’s go back to the Corp router and see what it shows us in the neighbor table:

```
Corp#sh ip eigrp neighbors
IP-EIGRP neighbors for process 10
H Address Interface Hold Uptime SRTT RTO Q Seq
(sec) (ms) Cnt Num
1 10.1.3.2 Se0/0/1 14 00:35:10 1 200 0 81
3 10.1.5.2 Se0/2/0 10 02:51:22 1 200 0 31
2 10.1.4.2 Se0/1/0 13 03:17:20 1 200 0 20
0 10.1.2.2 Se0/0/0 10 03:19:37 1 200 0 80
```

We read the information in this output like this:

- The H field indicates the order in which the neighbor was discovered.
- The hold time is how long this router will wait for a Hello packet to arrive from a specific neighbor.
- The uptime indicates how long the neighbor-ship has been established.
- The SRTT field is the smooth round-trip timer—an indication of the time it takes for a round-trip from this router to its neighbor and back. This value is used to determine how long to wait after a multicast for a reply from this neighbor. If a reply isn’t received in time, the router will switch to using unicasts in an attempt to complete the communication.

Now let’s see what’s in the Corp topology table by using the show ip eigrp topology command—this should be interesting!

```
Corp#sh ip eigrp topology
IP-EIGRP Topology Table for AS(10)/ID(10.1.5.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
r - reply Status, s - sia Status
P 10.1.11.0/24, 1 successors, FD is 2172416
via 10.1.5.2 (2172416/28160), Serial0/2/0
P 10.1.10.0/24, 1 successors, FD is 2172416
via 10.1.5.2 (2195456/281600), Serial0/2/0
P 10.1.9.0/24, 1 successors, FD is 2195456
via 10.1.4.2 (2195456/281600), Serial0/1/0
P 10.1.8.0/24, 1 successors, FD is 2195456
via 10.1.4.2 (2195456/72960), Serial0/1/0
```

```
P 10.1.12.0/24, 1 successors, FD is 2172416
via 10.1.5.2 (2172416/28160), Serial0/2/0
P 10.1.3.0/24, 1 successors, FD is 76839936
via Connected, Serial0/0/1
via 10.1.2.2 (9849856/7719936), Serial0/0/0

P 10.1.2.0/24, 1 successors, FD is 2169856
via Connected, Serial0/0/0
via 10.1.2.2 (2681856/551936), Serial0/0/0
P 10.1.1.0/24, 1 successors, FD is 28160
via Connected, FastEthernet0/1
P 10.1.7.0/24, 1 successors, FD is 793600
via 10.1.2.2 (2195456/281600), Serial0/0/0
via 10.1.3.2 (77081600/281600), Serial0/0/1
P 10.1.6.0/24, 1 successors, FD is 793600
via 10.1.2.2 (2195456/281600), Serial0/0/0
via 10.1.3.2 (77081600/281600), Serial0/0/1
P 10.1.5.0/24, 1 successors, FD is 2169856
via Connected, Serial0/2/0
P 10.1.4.0/24, 1 successors, FD is 2169856
via Connected, Serial0/1/0
```

Notice that every route is preceded by a P. This means that the route is in the passive state, which is a good thing because routes in the active state (A) indicate that the router has lost its path to this network and is searching for a replacement. Each entry also indicates the feasible distance, or FD, to each remote network plus the next-hop neighbor through which packets will travel to their destination. Plus, each entry also has two numbers in parentheses. The first indicates the feasible distance, and the second the advertised distance to a remote network.

Now here's where things get interesting—notice that under the 10.1.7.0 and 10.1.6.0 outputs there are two links to each network and that the feasible distance and advertised distance are different. What this means is that we have one successor to the networks and one feasible successor—a backup route! So very cool! You need to remember that even though both routes to network 10.1.6.0 and 10.1.7.0 are in the topology table, only the successor route (the one with the lowest metrics) will be copied and placed into the routing table.

EIGRP will load-balance across both links automatically when they are of equal variance (equal cost), but EIGRP can also load-balance across unequal-cost links as well if we use the variance command. The variance metric is set to 1 by default, meaning that only equal-cost links will load balance.

You can change the metric anywhere up to 128. Changing a variance value enables

EIGRP to install multiple, loop-free routes with unequal cost in a local routing table.

So basically, if the variance is set to 1, only routes with the same metric as the successor will be installed in the local routing table. And, for example, if the variance is set to 2, any EIGRP learned route with a metric less than two times the successor metric will be installed in the local routing table (if it is already a feasible successor).

Now's a great time for us to check out some debugging outputs. First, let's use the debug eigrp packet command that will show our Hello packets being sent between neighbor routers:

```
Corp#debug eigrp packet
EIGRP Packets debugging is on
(UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB,
SIAQUERY, SIAREPLY)
Corp#
*Mar 21 23:17:35.050: EIGRP: Sending HELLO on FastEthernet0/1
*Mar 21 23:17:35.050: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
*Mar 21 23:17:35.270: EIGRP: Received HELLO on Serial0/1/0 nbr 10.1.4.2
*Mar 21 23:17:35.270: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:35.294: EIGRP: Received HELLO on Serial0/0/0 nbr 10.1.2.2
*Mar 21 23:17:35.294: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
*Mar 21 23:17:38.014: EIGRP: Received HELLO on Serial0/2/0 nbr 10.1.5.2
*Mar 21 23:17:38.014: AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ
un/rely 0/0 peerQ un/rely 0/0
```

Troubleshoot and Resolve inter-VLAN routing problems

Configure, verify, and troubleshoot inter-VLAN routing

By default, only hosts that are members of the same VLAN can communicate. To change this and allow inter-VLAN communication, you need a router or a layer 3 switch. I'm going to start with the router approach.

To support ISL or 802.1Q routing on a Fast Ethernet interface, the router's interface is divided into logical interfaces—one for each VLAN. These are called sub-interfaces. From a Fast Ethernet or Gigabit interface, you can set the interface to trunk with the encapsulation command:

```
ISR#config t
ISR(config)#int f0/0.1
ISR(config-subif)#encapsulation ?
dot1Q IEEE 802.1Q Virtual LAN
ISR(config-subif)#encapsulation dot1Q ?
<1-4094> IEEE 802.1Q VLAN ID
```

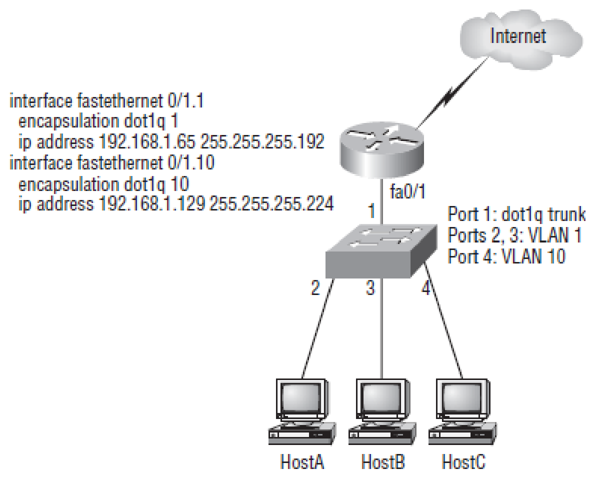
Notice that my 2811 router (named ISR) only supports 802.1Q. We'd need an older-model router to run the ISL encapsulation, but why bother?

The sub-interface number is only locally significant, so it doesn't matter which sub-interface numbers are configured on the router. Most of the time, I'll configure a sub-interface with the same number as the VLAN I want to route. It's easy to remember that way, since the sub-interface number is used only for administrative purposes.

It's really important that you understand that each VLAN is a separate subnet. True, I know—they don't have to be. But it really is a good idea to configure your VLANs as separate subnets, so just do those.

Now, I need to make sure you're fully prepared to configure inter-VLAN routing, as well as determine the port IP addresses of hosts connected in a switched VLAN environment. And as always, it's also a good idea to be able to fix any problems that may arise. To set you up for success, let me give you few examples.

First, start by looking at Figure 11.9, and read the router and switch configuration within it. By this point default gateways of each of the hosts in the VLANs.



```
interface fastethernet 0/1.1
 encapsulation dot1q 1
 ip address 192.168.1.65 255.255.255.192
interface fastethernet 0/1.10
 encapsulation dot1q 10
 ip address 192.168.1.129 255.255.255.224
```

FIGURE 11.9 Configuring Inter-VLAN example 1

The next step after that is to figure out which subnets are being used. By looking at the router configuration in the figure, you can see that we're using 192.168.1.64/26 with VLAN 1 and 192.168.1.128/27 with VLAN 10. And by looking at the switch configuration, you can see that ports 2 and 3 are in VLAN 1 and port 4 is in VLAN 10. This means that HostA and HostB are in VLAN 1, and HostC is in VLAN 10.

Here's what the hosts' IP addresses should be:

- HostA: 192.168.1.66, 255.255.255.192, default gateway 192.168.1.65
- HostB: 192.168.1.67, 255.255.255.192, default gateway 192.168.1.65
- HostC: 192.168.1.130, 255.255.255.224, default gateway 192.168.1.129

The hosts could be any address in the range—I just choose the first available IP address after the default gateway address. That wasn't so hard, was it? Now, again using Figure 11.10, let's go through the commands necessary to configure switch port 1 to establish a link with the router and provide inter-VLAN communication using the IEEE version for encapsulation. Keep in mind that the commands can vary slightly depending on what type of switch you're dealing with.

For a 2960 switch, use the following:

```
2960#config t
2960(config)#interface fa0/1
2960(config-if)#switchport mode trunk
```

As you already know, the 2960 switch can only run the 802.1Q encapsulation, so there's no need to specify it. You can't anyway! For a 3560, it's basically the same, but since it can run ISL and 802.1Q, you have to specify the trunking protocol you're going to use.

Let's take a look at Figure 11.10 and see what we can learn from it. This figure shows three VLANs, with two hosts in each of them.

The router in Figure 11.10 is connected to the fa0/1 switch port, and VLAN 2 is configured on port f0/6. Looking at the diagram, these are the things that Cisco expects you to know:

- The router is connected to the switch using sub-interfaces.
- The switch port connecting to the router is a trunk port.
- The switch ports connecting to the clients and the hub are access ports, not trunk ports.

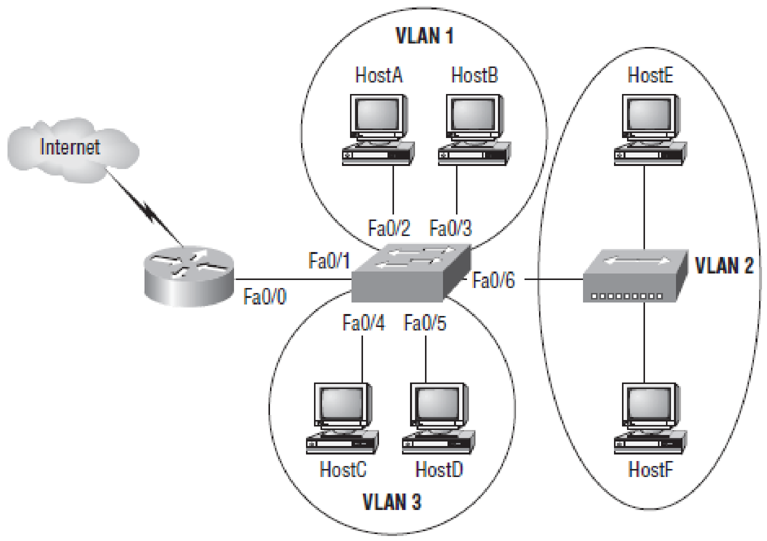


FIGURE 11.10 Inter-VLAN example 2

The configuration of the switch would look something like this:

```
2960#config t
2960(config)#int f0/1
2960(config-if)#switchport mode trunk
2960(config-if)#int f0/2
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/3
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/4
2960(config-if)#switchport access vlan 3
2960(config-if)#int f0/5
2960(config-if)#switchport access vlan 3
2960(config-if)#int f0/6
2960(config-if)#switchport access vlan 2
```

Before we configure the router, we need to design our logical network:

```
VLAN 1: 192.168.10.16/28
VLAN 2: 192.168.10.32/28
VLAN 3: 192.168.10.48/28
```

The configuration of the router would then look like this:

```
ISR#config t
ISR(config)#int f0/0
ISR(config-if)#no ip address
ISR(config-if)#no shutdown
ISR(config-if)#int f0/0.1
ISR(config-subif)#encapsulation dot1q 1
ISR(config-subif)#ip address 192.168.10.17 255.255.255.240
ISR(config-subif)#int f0/0.2
ISR(config-subif)#encapsulation dot1q 2
ISR(config-subif)#ip address 192.168.10.33 255.255.255.240
ISR(config-subif)#int f0/0.3
ISR(config-subif)#encapsulation dot1q 3
ISR(config-subif)#ip address 192.168.10.49 255.255.255.240
```

The hosts in each VLAN would be assigned an address from their subnet range, and the default gateway would be the IP address assigned to the router's sub-interface in that VLAN. Now, let's take a look at another figure and see if you can determine the switch and router configurations without looking at the answer—no cheating! Figure 11.11 shows a router connected to a 2960 switch with two VLANs. One host in each VLAN is assigned an IP address.

What are your router and switch configurations based on these IP addresses?

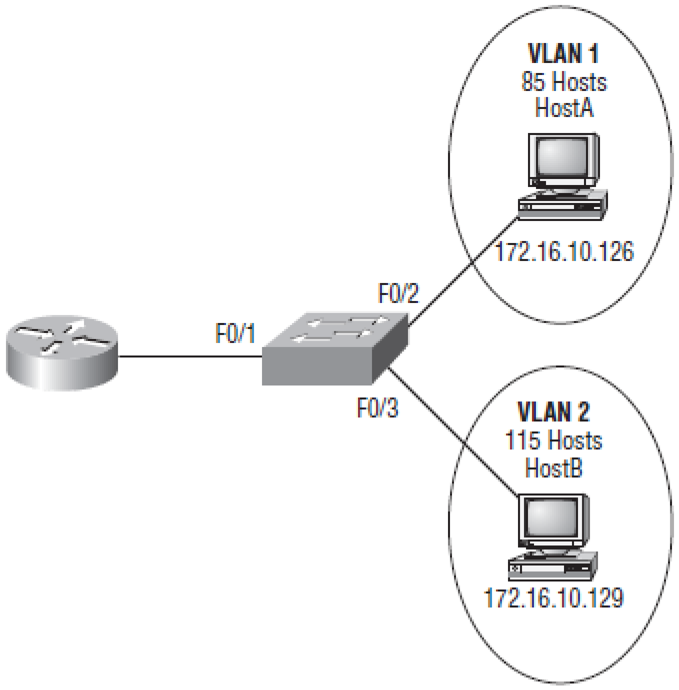


FIGURE 11.11 Inter-VLAN example 3

Since the hosts don't list a subnet mask, you have to look for the number of hosts used in each VLAN to figure out the block size. VLAN 1 has 85 hosts and VLAN 2 has 115 hosts. Each of these will fit in a block size of 128, which is a /25 mask, or 255.255.255.128. You should know by now that the subnets are 0 and 128; the 0 subnet (VLAN 1) has a host range of 1–126, and the 128 subnet (VLAN 2) has a range of 129–254. You can almost be fooled since HostA has an IP address of 126, which makes it almost seem that HostA and B are in the same subnet. But they're not, and you're way too smart by now to be fooled by this one!

Here is the switch configuration:

```
2960#config t
2960(config)#int f0/1
2960(config-if)#switchport mode trunk
2960(config-if)#int f0/2
2960(config-if)#switchport access vlan 1
2960(config-if)#int f0/3
2960(config-if)#switchport access vlan 2
```

Here is the router configuration:

```
ISR#config t
ISR(config)#int f0/0
ISR(config-if)#no ip address
ISR(config-if)#no shutdown
ISR(config-if)#int f0/0.1
ISR(config-subif)#encapsulation dot1q 1
ISR(config-subif)#ip address 172.16.10.1 255.255.255.128
ISR(config-subif)#int f0/0.2
ISR(config-subif)#encapsulation dot1q 2
ISR(config-subif)#ip address 172.16.10.254 255.255.255.128
```

I used the first address in the host range for VLAN 1 and the last address in the range for VLAN 2, but any address in the range would work. You just have to configure the host's default gateway to whatever you make the router's address.

Now, before we go on to the next example, I need to make sure that you know how to set the IP address on the switch. Since VLAN 1 is typically the administrative VLAN, we'll use an IP address from that pool of addresses. Here's how to set the IP address of the switch (I'm not nagging, but you really should already know this!):

```
2960#config t
2960(config)#int vlan 1
2960(config-if)#ip address 172.16.10.2 255.255.255.128
2960(config-if)#no shutdown
```

Yes, you have to do a no shutdown on the VLAN interface.
One more example, and then we'll move on to VTP—another important subject that you definitely don't want to miss! By looking at the router configuration, what are the IP address, mask, and default gateway of HostA? Use the last IP address in the range for HostA's address:

If you really look carefully at the router configuration (the hostname in this figure is just Router), there is a simple and quick answer. Both subnets are using a /28, or 255.255.255.240 mask, which is a block size of 16. The router's address for VLAN 1 is in subnet 128. The next subnet is 144, so the broadcast address of VLAN 1 is 143 and the valid host range is 129–142. So, the host address would be this:

```
IP Address: 192.168.10.142
Mask: 255.255.255.240
Default Gateway: 192.168.10.129
```

Troubleshoot and Resolve WAN implementation issues

Serial interfaces

Troubleshoot WAN implementation issues

If you have a point-to-point link, but the encapsulations aren't the same, the link will never come up. Figure 11.12 shows one link with PPP and one with HDLC.

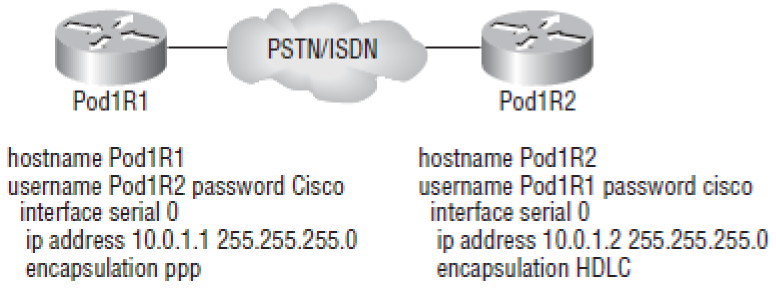

PSTN/ISDN
Pod1R1
Pod1R2
hostname Pod1R1
username Pod1R2 password Cisco
interface serial 0
ip address 10.0.1.1 255.255.255.0
encapsulation ppp
hostname Pod1R2
username Pod1R1 password cisco
interface serial 0
ip address 10.0.1.2 255.255.255.0
encapsulation HDLC

FIGURE 11.12 Mismatched WAN encapsulations

Look at router Pod1R1 in this output:

```
Pod1R1#sh int s0/0
Serial0/0 is up, line protocol is down
Hardware is PowerQUICC Serial
Internet address is 10.0.1.1/24
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
reliability 254/255, txload 1/255, rxload 1/255
Encapsulation PPP, loopback not set
Keepalive set (10 sec)
LCP REQsent
Closed: IPCP, CDPCP
```

The serial interface is down, and LCP is sending requests but will never receive any responses because router Pod1R2 is using the HDLC encapsulation. To fix this problem, you would have to go to router Pod1R2 and configure the PPP encapsulation on the serial interface. One more thing— even though the usernames are configured and they're wrong, it doesn't matter because the command ppp authentication chap isn't used under the serial interface configuration and the username command isn't relevant in this example.

PPP

Mismatched IP Addresses

A tricky problem to spot is if you have HDLC or PPP configured on your serial interface, but your IP addresses are wrong. Things seem to be just fine because the interfaces will show that they are up. Take a look at Figure 11.13, and see if you can see what I mean—the two routers are connected with different subnets—router Pod1R1 with 10.0.1.1/24 and router Pod1R2 with 10.2.1.2/24.

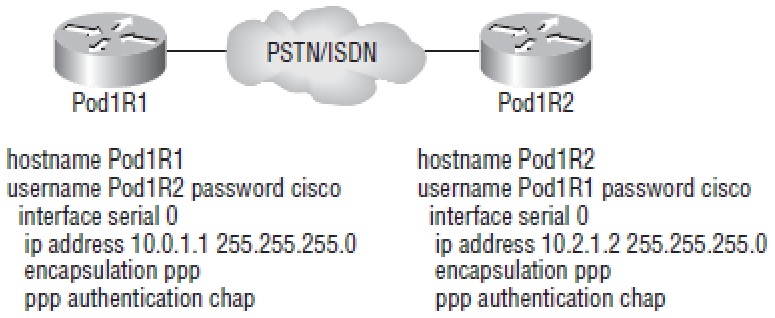


FIGURE 11.13 Mismatched IP addresses

This will never work. But as I said, take a look at the output:

```
Pod1R1#sh int s0/0
Serial0/0 is up, line protocol is up
Hardware is PowerQUICC Serial
Internet address is 10.0.1.1/24
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation PPP, loopback not set
Keepalive set (10 sec)
LCP Open
Open: IPCP, CDPCP
```

See that? The IP addresses between the routers are wrong, but the link looks like it's working fine. This is because PPP, like HDLC and Frame Relay, is a layer 2 WAN encapsulation and doesn't care about IP addresses at all. So, yes, the link is up, but you can't use IP across this link since it's misconfigured.

To find and fix this problem, you can use the show running-config or the show interfaces command on each router, or you can use the show cdp neighbors detail command:

```
Pod1R1#sh cdp neighbors detail
-------------------------
Device ID: Pod1R2
Entry address(es):
IP address: 10.2.1.2
```

You can view and verify the directly connected neighbor's IP address and then solve your problem.

Frame relay

Troubleshooting Frame Relay Networks

Troubleshooting Frame Relay networks isn't any harder than troubleshooting any other type of network as long as you know what to look for, which is what I'm going to cover now. We'll go over some basic problems that commonly occur in Frame Relay configuration and how to solve them.

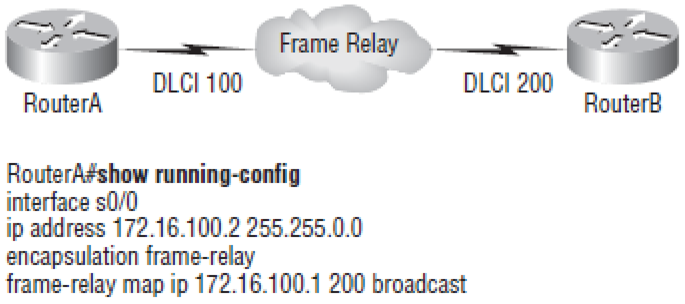
First on the list are serial encapsulation problems. As you learned recently, there are two Frame Relay encapsulations: Cisco and IETF. Cisco is the default, and it means that you have a Cisco router on each end of the Frame Relay network. If you don't have a Cisco router on the remote end of your Frame Relay network, then you need to run the IETF encapsulation as shown here:

```
RouterA(config)#int s0
RouterA(config-if)#encapsulation frame-relay ?
ietf Use RFC1490 encapsulation
<cr>
RouterA(config-if)#encapsulation frame-relay ietf
```

Once you verify that you're using the correct encapsulation, you then need to check out your Frame Relay mappings. For example, take a look at Figure 11.6.

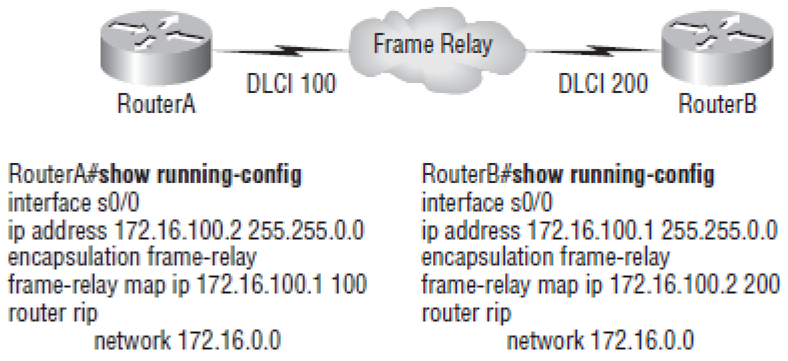
So, why can't RouterA talk to RouterB across the Frame Relay network? To find that out, take a close look at the frame-relay map statement. See the problem now? You cannot use a remote DLCI to communicate to the Frame Relay switch; you must use your DLCI number!

The mapping should have included DLCI 100 instead of DLCI 200.
Now that you know how to ensure that you have the correct Frame Relay encapsulation, and that DLCIs are only locally significant, let's look into some routing protocol problems typically associated with Frame Relay. See if you can find a problem with the two configurations in Figure 11.14.



```
RouterA#show running-config
interface s0/0
ip address 172.16.100.2 255.255.0.0
encapsulation frame-relay
frame-relay map ip 172.16.100.1 200 broadcast
```

FIGURE 11.14 Frame Relay mappings



```
RouterA#show running-config
interface s0/0
ip address 172.16.100.2 255.255.0.0
encapsulation frame-relay
frame-relay map ip 172.16.100.1 100
router rip
        network 172.16.0.0
```

```
RouterB#show running-config
interface s0/0
ip address 172.16.100.1 255.255.0.0
encapsulation frame-relay
frame-relay map ip 172.16.100.2 200
router rip
        network 172.16.0.0
```

FIGURE 11.15 Frame Relay routing problems

Hmmmm, well, the configs look pretty good. Actually, they look great, so what's the problem? Well, remember that Frame Relay is a NBMA network by default, meaning that it doesn't send any broadcasts across the PVC. So, because the mapping statements do not have the broadcast argument at the end of the line, broadcasts, like RIP updates, won't be sent across the PVC.

Monitor NetFlow statistics

NetFlow uses flows to provide statistics for accounting, network monitoring, and network planning. A flow is a unidirectional stream of packets that arrives on a source interface (or VLAN) and has the same values for the keys. A key is an identified value for a field within the packet. You create a flow using a flow record to define the unique keys for your flow.
Cisco NX-OS supports the Flexible NetFlow feature that enables enhanced network anomalies and security detection. Flexible NetFlow allows you to define an optimal flow record for a particular application by selecting the keys from a large collection of predefined fields.
All key values must match for the packet to count in a given flow. A flow might gather other fields of interest, depending on the export record version that you configure. Flows are stored in the NetFlow cache.

You can export the data that NetFlow gathers for your flow by using an exporter and export this data to a remote NetFlow collector. Cisco NX-OS exports a flow as part of a NetFlow export User Datagram Protocol (UDP) datagram under the following circumstances:

- The flow has been inactive or active for too long.
- The flow cache is getting full.
- One of the counters (packets or bytes) has exceeded its maximum value.
- You have forced the flow to export.

You define the size of the data that you want to collect for a flow using a monitor. The monitor combines the flow record and exporter with the NetFlow cache information.

Cisco NX-OS can gather NetFlow statistics in either full or sampled mode. Cisco NX-OS analyzes all packets on the interface or subinterface for full NetFlow mode. For sampled mode, you configure the sampling algorithm and rate that Cisco NX-OS analyzes packets.

Flow Records

A flow record defines the keys that NetFlow uses to identify packets in the flow as well as other fields of interest that NetFlow gathers for the flow. You can define a flow record with any combination of keys and fields of interest. Cisco NX-OS supports a rich set of keys. A flow record also defines the types of counters gathered per flow. You can configure 32-bit or 64-bit packet or byte counters. Cisco NX-OS enables the following match fields as the defaults when you create a flow record:

- match interface input
- match interface output
- match flow direction

Troubleshoot ether channel problems

STP Convergence

STP puts each RP and DP into a forwarding state, and ports that are neither RP nor DP into a blocking state. Those states may remain as is for days, weeks, or months. But at some point, some switch or link will fail, a link may change speeds (changing the STP cost), or the STP configuration may change. Any of these events can cause switches to repeat their STP algorithm, which may in turn change their own RP and any ports that are DPs.

When STP converges based on some change, not all the ports have to change their state. For instance, a port that was forwarding, if it still needs to forward, just keeps on forwarding.
Ports that were blocking that still need to block keep on blocking. But when a port needs to change state, something has to happen, based on the following rules:

- For interfaces that stay in the same STP state, nothing needs to change.
- For interfaces that need to move from a forwarding state to a blocking state, the switch immediately changes the state to blocking.
- For interfaces that need to move from a blocking state to a forwarding state, the switch first moves the interface to listening state, then learning state, each for the time specified by the forward delay timer (default 15 seconds). Only then is the interface placed into forwarding state.

Troubleshooting EtherChannel

EtherChannels can prove particularly challenging to troubleshoot for a couple of reasons. First, you have to be careful to match the correct configuration, and there are many more incorrect configuration combinations than there are correct combinations. Second, many interface settings must match on the physical links, both on the local switch and on the neighboring switch, before a switch will add the physical link to the channel.

Incorrect Options on the channel-group Command
These rules can be summarized as follows, for a single EtherChannel:

1. On the local switch, all the channel-group commands for all the physical interfaces must use the same channel-group number.
2. The channel-group number can be different on the neighboring switches.
3. If using the on keyword, you must use it on the corresponding interfaces of both switches.
4. If you use the desirable keyword on one switch, the switch uses PAgP; the other switch must use either desirable or auto.
5. If you use the active keyword on one switch, the switch uses LACP; the other switch must use either active or passive.

These rules summarize the correct configuration options, but the options actually leave many more incorrect choices. The following list shows some incorrect configurations that the switches allow, even though they would result in the EtherChannel not working. The list compares the configuration on one switch to another based on the physical interface configuration. Each lists the reasons why the configuration is incorrect:

- Configuring the on keyword on one switch, and desirable, auto, active, or passive on the other switch. The on keyword does not enable PAgP, and does not enable LACP, and the other options rely on PAgP or LACP.
- Configuring the auto keyword on both switches. Both use PAgP, but both wait on the other switch to begin negotiations.
- Configuring the passive keyword on both switches. Both use LACP, but both wait on the other switch to begin negotiations.
- Configuring the active keyword on one switch and either desirable or auto on the other switch. The active keyword uses LACP, whereas the other keywords use PAgP.
- Configuring the desirable keyword on one switch and either active or passive on the other switch. The desirable keyword uses PAgP, whereas the other keywords use LACP.

Example 11.2 shows an example that matches the last item in the list. In this case, SW1’s two ports (F0/14 and F0/15) have been configured with the desirable keyword, and SW2’s matching F0/16 and F0/17 have been configured with the active keyword. The example lists some telling status information about the failure, with notes following the example.

Example 11.2 Ruling Out Switches as Root Based on Having a Root Port

```
SW1# show etherchannel summary
Flags: D - down P - bundled in port-channel
I - stand-alone s - suspended
H - Hot-standby (LACP only)
R - Layer3 S - Layer2
U - in use f - failed to allocate aggregator
M - not in use, minimum links not met
```

```
u - unsuitable for bundling
w - waiting to be aggregated
d - default port
Number of channel-groups in use: 1
Number of aggregators: 1
Group Port-channel Protocol Ports
------+-------------+-----------+-------------------------------------------
---
1 Po1(SD) PAgP Fa0/14(I) Fa0/15(I)
SW1# show interfaces status | include Po|14|15
Port Name Status Vlan Duplex Speed Type
Fa0/14 connected 301 a-full a-100 10/100BaseTX
Fa0/15 connected 301 a-full a-100 10/100BaseTX
Po1 notconnect unassigned auto auto
```

Start at the top, in the legend of the show etherchannel summary command. The D code letter means that the channel itself is down, with S meaning that the channel is a Layer 2 EtherChannel. Code I means that the physical interface is working independently from the PortChannel (described as “stand-alone”). Then, the bottom of that command’s output highlights Portchannel (Po1) as Layer 2 EtherChannel in a down state (SD), with F0/14 and F0/15 as stand-alone interfaces (I).

Interestingly, because the problem is a configuration mistake, the two physical interfaces still operate independently, as if the port channel did not exist. The last command in the example shows that while the Portchannel 1 interface is down, the two physical interfaces are in a connected state.

Configuration Checks Before Adding Interfaces to EtherChannels
Even when the channel-group commands have all been configured correctly, other configuration settings can cause problems as well. This last topic examines those configuration settings and their impact.

First, a local switch checks each new physical interface that is configured to be part of an EtherChannel, comparing each new link to the existing links. That new physical interface’s settings must be the same as the existing links; otherwise, the switch does not add the new link to the list of approved and working interfaces in the channel. That is, the physical interface remains configured as part of the port channel, but it is not used as part of the channel, often being placed into some nonworking state.

The list of items the switch checks includes the following:

- Speed
- Duplex
- Operational access or trunking state (all must be access, or all must be trunks)
- If an access port, the access VLAN
- If a trunk port, the allowed VLAN list (per the switchport trunk allowed command)
- If a trunk port, the native VLAN
- STP interface settings

In addition, switches check the settings on the neighboring switch. To do so, the switches either use PAgP or LACP (if already in use), or Cisco Discovery Protocol (CDP) if using manual configuration. The neighbor must match on all parameters in this list except the STP settings.

As an example, SW1 and SW2 again use two links in one EtherChannel. Before configuring the EtherChannel, SW1's F0/15 was given a different STP port cost than F0/14. Example 11.3 picks up the story just after configuring the correct channel-group commands, when the switch is deciding whether to use F0/14 and F0/15 in this EtherChannel.

Example 11.3 Local Interfaces Fail in EtherChannel Because of Mismatched STP Cost

```
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error
detected on
Po1, putting Fa0/14 in err-disable state
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error
detected on
Po1, putting Fa0/15 in err-disable state
*Mar 1 23:18:56.132: %PM-4-ERR_DISABLE: channel-misconfig (STP) error
detected on
Po1, putting Po1 in err-disable state
*Mar 1 23:18:58.120: %LINK-3-UPDOWN: Interface FastEthernet0/14, changed
state to
down
*Mar 1 23:18:58.137: %LINK-3-UPDOWN: Interface Port-channel1, changed state
to down
*Mar 1 23:18:58.137: %LINK-3-UPDOWN: Interface FastEthernet0/15, changed
state to
down
SW1# show etherchannel summary
Flags: D - down P - bundled in port-channel
I - stand-alone s - suspended
H - Hot-standby (LACP only)
R - Layer3 S - Layer2
U - in use f - failed to allocate aggregator
M - not in use, minimum links not met
u - unsuitable for bundling
w - waiting to be aggregated
d - default port
Number of channel-groups in use: 1

Number of aggregators: 1

Group Port-channel Protocol Ports

------+-------------+-----------+-----------------------------------------
---

1 Po1(SD) - Fa0/14(D) Fa0/15(D)
```

The messages at the top of the example specifically state what the switch does when thinking about whether the interface settings match. In this case, SW1 detects the different STP costs. SW1 does not use F0/14, does not use F0/15, and even places them into an err-disabled state. The switch also puts the Port-Channel into err-disabled state. As a result, the Port Channel is not operational, and the physical interfaces are also not operational.

To solve this problem, you must reconfigure the physical interfaces to use the same STP settings. In addition, the port-channel and physical interfaces must be shut-down, and then no shutdown, to recover from the err-disabled state. (Note that when a switch applies the shutdown and no shutdown commands to a port channel, it applies those same commands to the physical interfaces, as well; so, just do the shut-down/no shutdown on the port-channel interface.)

WAN Technologies

Identify different WAN Technologies

Metro Ethernet

The emergence of Metro Ethernet as a viable method of providing both point-to-point and multipoint services has been driven by an abundance of new fiber deployment to business areas. Enterprise customers with years of Ethernet experience in the campus have developed such a comfort level and confidence with Ethernet that they are now asking their service providers for Ethernet as an access option. Ethernet might be the most scalable transport technology ever developed. Starting at 10 Mbps, it has now evolved to 10 Gbps, with plans for 40 Gbps. Several prominent methods exist for transporting Ethernet over Metro networks, including these key solution approaches:

- Delivering Ethernet services over dark fiber
- Delivering Ethernet services over SONET/Synchronous Digital
- Hierarchy (SDH) networks
- Delivering Ethernet services that use Resilient Packet Ring (RPR) technology

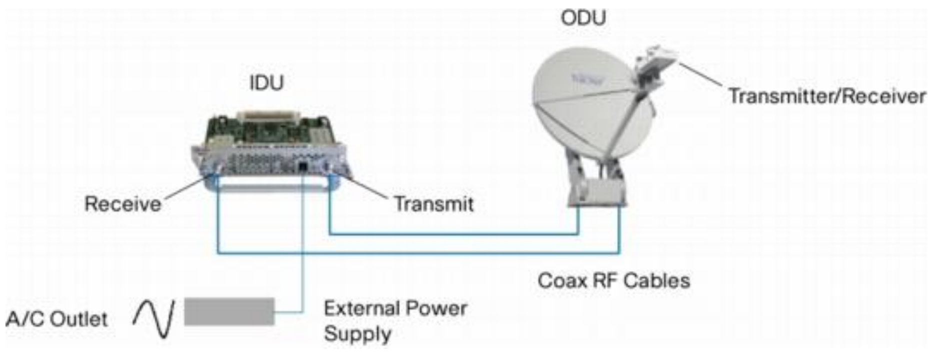
VSAT

The IP VSAT Satellite WAN Network Module for the Cisco Integrated Services Routers provides two-way broadband satellite connectivity for non-terrestrial WAN Backup, IP multicast, and broadband internet access.

The Cisco IP VSAT Satellite WAN Network Module (Figure 1) provides two-way broadband satellite connectivity for remote sites and branch offices on the Cisco 2800 and 3800 Series Integrated Services

Routers as well as the Cisco 2600XM, 2691, and 3700 Series Access Routers. Using the highly reliable and available-everywhere satellite network, the Cisco IP VSAT module can provide resilient WAN backup for enterprise branch offices in case of primary connectivity failure, thus enhancing network reliability. When combined with the advanced services available on integrated services routers, such as voice and wireless, a highly portable and self-contained emergency response communications kit can be created to provide instant and mobile communication at a disaster site. Satellite multicast provides an efficient and cost-effective way to deliver bandwidth-rich content such as video, audio, and software upgrades to multiple sites. Using the Cisco IP VSAT module in conjunction with the Cisco Enterprise CDN (Content Delivery Network) solution, enterprise customers can create a highly reliable and scalable content delivery system for their branch offices worldwide.

Completely IP-based, the Cisco IP VSAT Module supports the latest in satellite technology with respect to access scheme, modulation, and coding to provide maximum efficiency and performance from the satellite network. The module is compatible with satellite services that use Gilat SkyEdge hub systems. The module is the indoor unit (IDU) built into a network module form factor, and it connects to the outdoor unit (ODU), which consists of the dish antenna and the transmitter/receiver using coaxial RF cables.



The Cisco IP VSAT module provides for IP-based data, voice, and video communications over satellite, thus allowing enterprise, commercial, and government customers to enhance their business continuity solution, stream rich-media content using multicast, and set up instant and mobile communication kits for disaster preparedness.

MPLS

Multi-Protocol Label Switching (MPLS) is a data-carrying mechanism that emulates some properties of a circuit-switched network over a packet-switched network. MPLS is a switching mechanism that imposes labels (numbers) to packets and then uses those labels to forward packets. The labels are assigned on the edge MPLS of the network, and forwarding inside the MPLS network is done solely based on labels. Labels usually correspond to a path to layer 3 destination addresses (equal to IP destination-based routing). MPLS was designed to support forwarding of protocols other than TCP/IP. Because of this, label switching within the network is performed the same regardless of the layer 3 protocol. In larger networks, the result of MPLS labeling is that only the edge routers perform a routing lookup. All the core routers forward packets based on the labels, which makes forwarding the packets through the service provider's network faster. (Most companies are replacing their Frame Relay networks with MPLS today).

T1/E1

The Cisco Channelized T1/E1 and ISDN PRI High-Speed WAN Modules combine multiple T1/E1 WAN connectivity-Channelized T1/E1 and ISDN Primary Rate Interface (PRI), in the same card. Applications include fractional or full T1/E1 WAN connectivity, ISDN PRI for primary WAN link or WAN backup, and dial access aggregation. With flexible WAN connectivity options, together with integrated routing, security, voice, and wireless capabilities, the Cisco Integrated Services Routers can meet every need of enterprise-class branch offices today and in the future. Three versions are available, 1- and 2-port cards (Figure 1) in a single-wide high-speed WAN interface card (HWIC), and a 8-port cards (Figure 2) in a single-wide network module. The different versions help enable customers to deploy different port densities according to the needs of individual offices.
The modules can be used in T1 or E1 networks, selectable by software configuration. The integrated channel service unit/data service unit (CSU/DSU) function allows customers to consolidate customer premises equipment (CPE). The modules support balanced and unbalanced E1 connectivity and conform to the G.703 and G.704 standards for unframed and framed E1 modes. The Channelized T1/E1 and ISDN PRI modules work with the digital modem module in the Cisco 2800, 2900, 3800, and 3900 Series Integrated Services Routers to provide V.90- and V.92-compliant digital dial access aggregation.

The 1- and 2-port Channelized T1/E1 and ISDN PRI HWICs are updated versions of the E1/T1 ISDN PRI network modules, with the same functions and performance in a compact form factor of HWIC. You can save the network module slots for other LAN/WAN connectivity, and your deployment flexibility is greatly enhanced.

ISDN

ISDN Integrated Services Digital Network (ISDN) is a set of digital services that transmits voice and data over existing phone lines. ISDN offers a cost-effective solution for remote users who need a higher-speed connection than analog dial-up links can give them, and it's also a good choice to use as a backup link for other types of links like Frame Relay or T1 connections.

DSL

Digital subscriber line is a technology used by traditional telephone companies to deliver advanced services (high-speed data and sometimes video) over twisted-pair copper telephone wires. It typically has lower data-carrying capacity than HFC networks, and data speeds can be range limited by line lengths and quality. Digital subscriber line is not a complete end-to-end solution but rather a Physical layer transmission technology like dial-up, cable, or wireless. DSL connections are deployed in the last mile of a local telephone network—the local loop. The connection is set up between a pair of modems on either end of a copper wire that is run between the CPE and the Digital Subscriber Line Access Multiplexer (DSLAM). A DSLAM is the device located at the provider's CO and concentrates connections from multiple DSL subscribers.

Frame Relay

A packet-switched technology that made its debut in the early 1990s, Frame Relay is a high-performance Data Link and Physical layer specification. It's pretty much a successor to X.25, except that much of the technology in X.25 used to compensate for physical errors (noisy lines) has been eliminated. An upside to Frame Relay is that it can be more cost effective than point-to-point links, plus it typically runs at speeds of 64Kbps up to 45Mbps (T3). Another Frame Relay benefit is that it provides features for dynamic bandwidth allocation and congestion control.

Cable

A cable modem is a type of modem that provides access to a data signal sent over the cable television infrastructure. Cable modems are primarily used to deliver broadband Internet access, taking advantage of unused bandwidth on a cable television network. The bandwidth of business cable modem services typically ranges from 3 Mbps up to 30 Mbps or more. Current cable modem systems use the Ethernet frame format for data transmission over upstream and downstream data channels. Each of the downstream data channels and the associated upstream data channels on a cable network form an extended Ethernet WAN.

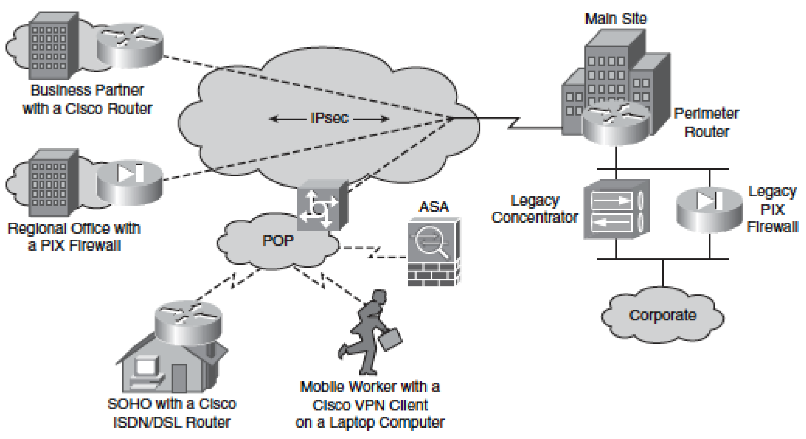
VPN

Cisco VPN solutions provide an Internet-based WAN infrastructure for connecting branch offices, home offices, business partner sites, and remote telecommuters to all or portions of a company network. With cost-effective, high-bandwidth Internet connectivity that is secured by encrypted VPN tunnels, you can reduce WAN bandwidth costs while increasing connectivity speeds.

By integrating advanced network intelligence and routing, Cisco VPNs reliably transport complex mission-critical traffic, such as voice and client-server applications, without compromising communications quality or security.

VPNs and Their Benefits

A VPN is an encrypted connection between private networks over a public network such as the Internet. The V stands for virtual, and the N stands for network. The information from a private network is securely transported over a public network, the Internet, to form a virtual network. The P stands for private. To remain private, the traffic is encrypted to keep the data confidential. Instead of using a dedicated Layer 2 connection such as a leased line, a VPN uses IPsec to form virtual connections that are routed through the Internet from the private network of the company to the remote site or employee host. The following figure shows some examples of using VPNs to connect different types of remote sites.


Business Partner with a Cisco Router
IPsec
Main Site
Perimeter Router
Regional Office with a PIX Firewall
POP
ASA
Legacy Concentrator
Legacy PIX Firewall
Corporate
SOHO with a Cisco ISDN/DSL Router
Mobile Worker with a Cisco VPN Client on a Laptop Computer

Configure and verify a basic WAN serial connection

As you can imagine, there are a few things that you need to know before connecting you're WAN in order to make sure that everything goes well. For starters, you have to understand the kind of WAN Physical layer implementation that Cisco provides as well as ensure that you're familiar with the various types of WAN serial connectors involved.

The good news is that Cisco serial connections support almost any type of WAN service.

Your typical WAN connection is a dedicated leased line using HDLC, PPP, and Frame Relay with speeds that can kick it up to 45Mbps (T3). HDLC, PPP, and Frame Relay can use the same Physical layer specifications, and I'll go over the various types of connections and then move on to telling you all about the WAN protocols specified in the CCNA objectives.

Serial Transmission

WAN serial connectors use serial transmission, something that takes place 1 bit at a time over a single channel.

Parallel transmission can pass at least 8 bits at a time, but all WANs use serial transmission.

Cisco routers use a proprietary 60-pin serial connector that you have to get from Cisco or a provider of Cisco equipment. Cisco also has a new, smaller proprietary serial connection that's about one-tenth the size of the 60-pin basic serial cable, called the "smart-serial "—I'm not sure why. But I do know that you have to make sure that you have the right type of interface in your router before using this cable connector.
The type of connector you have on the other end of the cable depends on your service provider and its particular end-device requirements. There are several different types of ends you'll run into:

- EIA/TIA-232
- EIA/TIA-449
- V.35 (used to connect to a CSU/DSU)
- EIA-530

Make sure that you're clear on these things: Serial links are described in frequency or cycles per second (hertz). The amount of data that can be carried within these frequencies is called bandwidth. Bandwidth is the amount of data in bits per second that the serial channel can carry.

Data Terminal Equipment and

Data Communication Equipment
By default, router interfaces are data terminal equipment (DTE) , and they connect into data communication equipment (DCE) like a channel service unit/data service unit (CSU/DSU) .

The CSU/DSU then plugs into a demarcation location (demarc) and is the service provider's last responsibility. Most of the time, the demarc is a jack that has an RJ-45 (8-pin modular) female connector, located in a telecommunications closet.

Actually, you may already have heard of demarcs. If you've ever had the glorious experience of reporting a problem to your service provider, they'll usually tell you everything tests out fine up to the demarc, so the problem must be the CPE, or customer premises equipment. In other words, it's your problem not theirs.

Figure 12.1 shows a typical DTE-DCE-DTE connection and the devices used in the network.

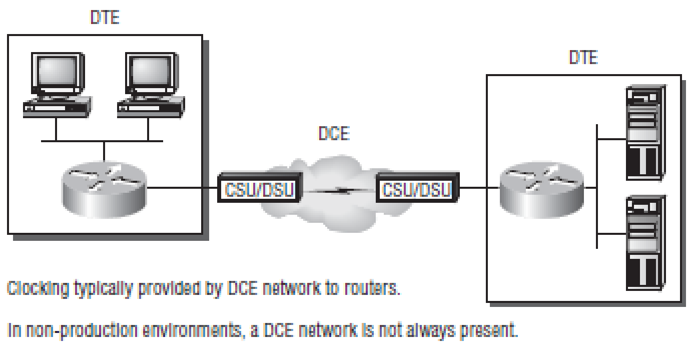

DTE
DTE
DCE
CSU/DSU
CSU/DSU
Clocking typically provided by DCE network to routers.
In non-production environments, a DCE network is not always present.

FIGURE 12.1 DTE-DCE-DTE WAN connection

The idea behind a WAN is to be able to connect two DTE networks through a DCE network. The DCE network includes the CSU/DSU, through the provider's wiring and switches, all the way to the CSU/DSU at the other end. The network's DCE device (CSU/DSU) provides clocking to the DTE-connected interface (the router's serial interface).
As mentioned, the DCE network provides clocking to the router; this is the CSU/DSU. If you have a nonproduction network and you're using a WAN crossover type of cable and do not have a CSU/DSU, then you need to provide clocking on the DCE end of the cable by using the clock rate command.

High-Level Data-Link Control (HDLC) Protocol

The High-Level Data-Link Control (HDLC) protocol is a popular ISO-standard, bit-oriented, Data Link layer protocol. It specifies an encapsulation method for data on synchronous serial data links using frame characters and checksums. HDLC is a point-to-point protocol used on leased lines. No authentication can be used with HDLC.

In byte-oriented protocols, control information is encoded using entire bytes. On the other hand, bit-oriented protocols use single bits to represent the control information. Some common bit-oriented protocols include SDLC, LLC, HDLC, TCP, and IP.
HDLC is the default encapsulation used by Cisco routers over synchronous serial links.

And Cisco's HDLC is proprietary—it won't communicate with any other vendor's HDLC implementation. But don't give Cisco grief for it— everyone's HDLC implementation is proprietary. Figure 12.2 shows the Cisco HDLC format.

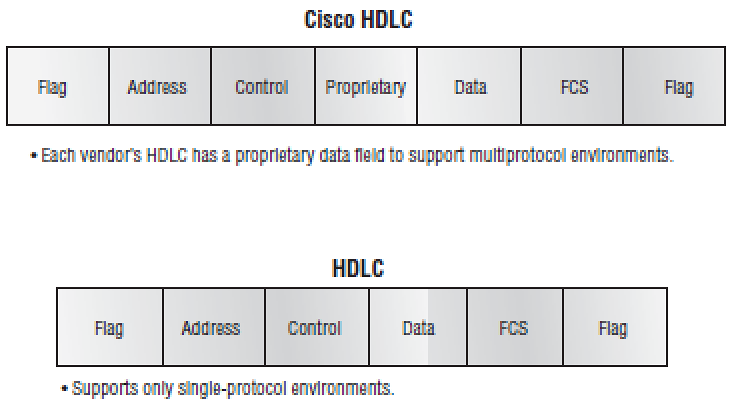

Cisco HDLC
Flag
Address
Control
Proprietary
Data
FCS
Flag
• Each vendor's HDLC has a proprietary data field to support multiprotocol environments.
HDLC
Flag
Address
Control
Data
FCS
Flag
• Supports only single-protocol environments.

FIGURE 12.2 Cisco HDLC frame format

As shown in the figure, the reason that every vendor has a proprietary HDLC encapsulation method is that each vendor has a different way for the HDLC protocol to encapsulate multiple Network layer protocols. If the vendors didn't have a way for HDLC to communicate the different layer 3 protocols, then HDLC would only be able to carry one protocol. This proprietary header is placed in the data field of the HDLC encapsulation.

Configuring HDLC on Cisco Routers

Configuring HDLC encapsulation on an interface is really pretty straightforward. To configure it from the CLI, follow these simple router commands:

```
Router#
config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
int s0
Router(config-if)#
encapsulation hdlc
Router(config-if)#
^Z
Router#
```

So, let's say you only have one Cisco router, and you need to connect to a non-Cisco router because you're other Cisco router is on order. What would you do? You couldn't use the default HDLC serial encapsulation because it wouldn't work. Instead, you would use something like PPP, an ISO-standard way of identifying the upper-layer protocols.

Point-to-Point Protocol (PPP)

Let's spend a little time on Point-to-Point Protocol (PPP). Remember that it's a Data Link layer protocol that can be used over either asynchronous serial (dial-up) or synchronous serial (ISDN) media. It uses Link Control Protocol (LCP) to build and maintain data-link connections.

Network Control Protocol (NCP) is used to allow multiple Network layer protocols (routed protocols) to be used on a point-to-point connection.

Since HDLC is the default serial encapsulation on Cisco serial links and it works great, why and when would you choose to use PPP? Well, the basic purpose of PPP is to transport layer 3 packets across a Data Link layer point-to-point link, and it's nonproprietary. So, unless you have all Cisco routers, you need PPP on your serial interfaces—the HDLC encapsulation is Cisco proprietary, remember? Plus, since PPP can encapsulate several layer 3 routed protocols and provide authentication, dynamic addressing, and callback, PPP could be the best encapsulation solution for you instead of HDLC. Figure 12.3 shows the protocol stack compared to the OSI reference model.

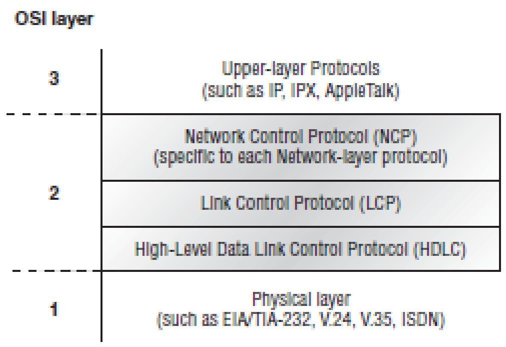

OSI layer
3
Upper-layer Protocols (such as IP, IPX, AppleTalk)
2
Network Control Protocol (NCP) (specific to each Network-layer protocol)
Link Control Protocol (LCP)
High-Level Data Link Control Protocol (HDLC)
1
Physical layer (such as EIA/TIA-232, V.24, V.35, ISDN)

FIGURE 12.3 Point-to-Point Protocol stack

PPP contains four main components:

EIA/TIA-232-C, V.24, V.35, and ISDN
A Physical layer international standard for serial communication. HDLC A method for encapsulating datagrams over serial links. LCP A method of establishing, configuring, maintaining, and terminating the point-to-point connection.

NCP
A method of establishing and configuring different Network layer protocols. NCP is designed to allow the simultaneous use of multiple Network layer protocols. Some examples of protocols here are IPCP (Internet Protocol Control Protocol) and IPXCP (Internetwork Packet Exchange Control Protocol).

Burn it into your mind that the PPP protocol stack is specified at the Physical and Data Link layers only. NCP is used to allow communication of multiple Network layer protocols by encapsulating the protocols across a PPP data link.

Configuring PPP on Cisco Routers
Configuring PPP encapsulation on an interface is the same as HDLC. To configure it from the CLI, follow these simple router commands:

```
Router#
config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int s0
Router(config-if)#encapsulation ppp
Router(config-if)#^Z
Router#
```

Configure and verify a PPP connection between Cisco routers

Configure and verify a PPP connection between Cisco routers

After you configure your serial interface to support PPP encapsulation, you can configure authentication using PPP between routers. First, you need to set the hostname of the router, if it's not already set. Then you set the username and password for the remote router that will be connecting to your router:
Here's an example:

```
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname RouterA
RouterA(config)#username RouterB password cisco
```

When using the hostname command, remember that the username is the hostname of the remote router that's connecting to your router. And it's case sensitive, too. Also, the password on both routers must be the same. It's a plain-text password that you can see with a show run command; you can encrypt the password by using the command service password-encryption.

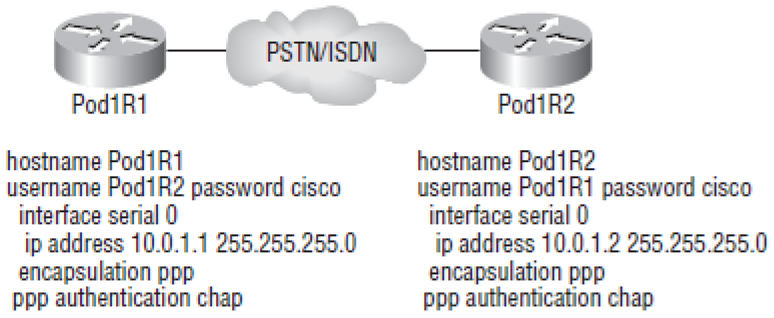
You must have a username and password configured for each remote system you plan to connect to. The remote routers must also be configured with usernames and passwords. Now, after you've set the hostname, usernames, and passwords, choose the authentication type, either CHAP or PAP:

```
RouterA#config t
Enter configuration commands, one per line. End with CNTL/Z.
RouterA(config)#int s0
RouterA(config-if)#ppp authentication chap pap
RouterA(config-if)#^Z
RouterA#
```

If both methods are configured on the same line, as shown here, then only the first method will be used during link negotiation—the second acts as a backup just in case the first method fails.

Verifying PPP Encapsulation

Okay—now that PPP encapsulation is enabled, let me show you how to verify that it's up and running. First, let's take a look at a figure of a sample network. Figure 12.4 shows two routers connected with either a point-to-point serial or ISDN connection.



```
hostname Pod1R1
username Pod1R2 password cisco
 interface serial 0
  ip address 10.0.1.1 255.255.255.0
 encapsulation ppp
 ppp authentication chap
```

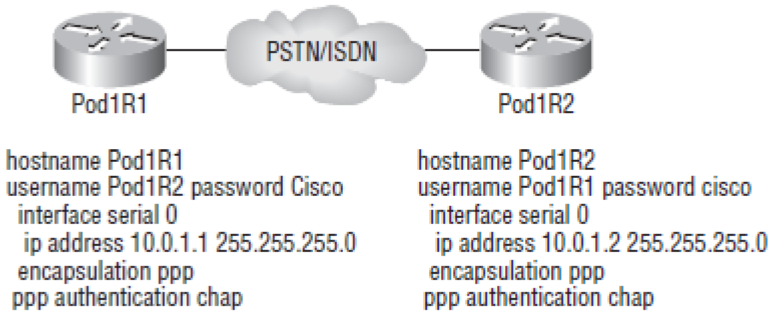
```
hostname Pod1R2
username Pod1R1 password cisco
 interface serial 0
  ip address 10.0.1.2 255.255.255.0
 encapsulation ppp
 ppp authentication chap
```

FIGURE 12.4 PPP authentication example

You can start verifying the configuration with the show interface command:

```
Pod1R1#sh int s0/0
Serial0/0 is up, line protocol is up
Hardware is PowerQUICC Serial
Internet address is 10.0.1.1/24
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
reliability 239/255, txload 1/255, rxload 1/255
Encapsulation PPP
loopback not set
Keepalive set (10 sec)
LCP Open
Open: IPCP, CDPCP
[output cut]
```

Notice that the sixth line lists encapsulation as PPP and the eighth line shows that the LCP is open. This means that it has negotiated the session establishment and all is good! The ninth line tells us that NCP is listening for the protocols IP and CDP.
But what will you see if everything isn't perfect? I'm going to type in the configuration shown in Figure 12.5 and find out.



```
hostname Pod1R1
username Pod1R2 password Cisco
 interface serial 0
  ip address 10.0.1.1 255.255.255.0
 encapsulation ppp
 ppp authentication chap
```

```
hostname Pod1R2
username Pod1R1 password cisco
 interface serial 0
  ip address 10.0.1.2 255.255.255.0
 encapsulation ppp
 ppp authentication chap
```

FIGURE 12.5 Failed PPP authentications

Okay—what's wrong here? Take a look at the usernames and passwords. Do you see the problem now? That's right; the C is capitalized on the Pod1R2 username command found in the configuration of router

Pod1R1. This is wrong because the usernames and passwords are case sensitive, remember? Let's take a look at the show interface command and see what happens:

```
Pod1R1#sh int s0/0
Serial0/0 is up, line protocol is down
Hardware is PowerQUICC Serial
Internet address is 10.0.1.1/24
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
reliability 243/255, txload 1/255, rxload 1/255
Encapsulation PPP, loopback not set
Keepalive set (10 sec)
LCP Closed
Closed: IPCP, CDPCP
```

First, notice in the first line of output that Serial0/0 is up, line protocol is down.
This is because there are no keep alives coming from the remote router. Next, notice that the LCP is closed because the authentication failed.

Debugging PPP Authentication
To display the CHAP authentication process as it occurs between two routers in the network, just use the command debug ppp authentication.

If your PPP encapsulation and authentication are set up correctly on both routers, and your usernames and passwords are all good, then the debug ppp authentication command will display output that looks like this:

```
d16h: Se0/0 PPP: Using default call direction
1d16h: Se0/0 PPP: Treating connection as a dedicated line
1d16h: Se0/0 CHAP: O CHALLENGE id 219 len 27 from "Pod1R1"
1d16h: Se0/0 CHAP: I CHALLENGE id 208 len 27 from "Pod1R2"
1d16h: Se0/0 CHAP: O RESPONSE id 208 len 27 from "Pod1R1"
1d16h: Se0/0 CHAP: I RESPONSE id 219 len 27 from "Pod1R2"
1d16h: Se0/0 CHAP: O SUCCESS id 219 len 4
1d16h: Se0/0 CHAP: I SUCCESS id 208 len 4
```

But if you have the username wrong, as we did previously in the PPP authentication failure example back in Figure 12.6, the output would look something like this:

```
1d16h: Se0/0 PPP: Using default call direction
1d16h: Se0/0 PPP: Treating connection as a dedicated line
1d16h: %SYS-5-CONFIG_I: Configured from console by console
1d16h: Se0/0 CHAP: O CHALLENGE id 220 len 27 from "Pod1R1"
1d16h: Se0/0 CHAP: I CHALLENGE id 209 len 27 from "Pod1R2"
1d16h: Se0/0 CHAP: O RESPONSE id 209 len 27 from "Pod1R1"
1d16h: Se0/0 CHAP: I RESPONSE id 220 len 27 from "Pod1R2"
1d16h: Se0/0 CHAP: O FAILURE id 220 len 25 msg is "MD/DES compare failed"
```

PPP with CHAP authentication is a three-way authentication, and if the username and passwords are not configured exactly the way they should be, then the authentication will fail and the link will be down.

Configure and verify Frame Relay on Cisco routers

Frame Relay is still one of the most popular WAN services deployed over the past decade, and there's a good reason for this—cost. And it's a rare network design or designer that has the privilege to ignore that all-important cost factor!

By default, Frame Relay is classified as a non-broadcast multi-access (NBMA) network, meaning it doesn't send any broadcasts like RIP updates across the network.

Frame Relay has at its roots a technology called X.25, and it essentially incorporates the components of X.25 that are still relevant to today's reliable and relatively "clean" telecommunications networks, while leaving out the no-longer-needed error-correction components.

It's substantially more complex than the simple leased-line networks. The leased-line networks are easy to conceptualize— but not as much when it comes to Frame Relay. It can be significantly more complex and versatile, which is why it's often represented as a "cloud" in networking graphics. I'll get to that in a minute—for right now, I'm going to introduce Frame Relay in concept and show you how it differs from simpler leased-line technologies.

Frame Relay Implementation and Monitoring

As I've said, there are a ton of Frame Relay commands and configuration options, but I'm going to zero in on the ones you really need to know when studying for the CCNA exam objectives.

I'm going to start with one of the simplest configuration options—two routers with a single PVC between them. Next, I'll show you a more complex configuration using sub-interfaces, and demonstrate some of the monitoring commands available to verify the configuration.

Single Interface

Let's get started by looking at a simple example. Say that we just want to connect two routers with single PVC. Here's how that configuration would look:

```
RouterA#config t
Enter configuration commands, one per line. End with CNTL/Z.
RouterA(config)#int s0/0
RouterA(config-if)#encapsulation frame-relay
RouterA(config-if)#ip address 172.16.20.1 255.255.255.0
RouterA(config-if)#frame-relay lmi-type ansi
RouterA(config-if)#frame-relay interface-dlci 101
RouterA(config-if)#^Z
RouterA#
```

The first step is to specify the encapsulation as Frame Relay. Notice that since I didn’t specify a particular encapsulation type—either Cisco or IETF—the Cisco default type was used. If the other router were non-Cisco, I would’ve specified IETF. Next, I assigned an IP address to the interface, then I specified the LMI type of ANSI (the default being Cisco) based on information provided by the telecommunications provider. Finally, I added the Data Link Connection Identifier (DLCI) of 101, which indicates the PVC we want to use (again, given to me by my ISP) and assumes that there’s only one PVC on this physical interface.

That’s all there is to it—if both sides are configured correctly, the circuit will come up.

Sub-interfaces

You probably know by now that we can have multiple virtual circuits on a single serial interface and yet treat each as a separate interface—I did mention this earlier. We can make this happen by creating sub-interfaces. Think of a sub-interface as a logical interface defined by the IOS software.

Several sub-interfaces will share a single hardware interface, yet for configuration purposes they operate as if they were separate physical interfaces, something known as multiplexing. To configure a router in a Frame Relay network so that it will avoid split horizon issues by not permitting routing updates, just configure a separate sub-interface for each PVC, with a unique DLCI and subnet assigned to the sub-interface. You define sub-interfaces using a command like int s0.subinterface number. First, you have to set the encapsulation on the physical serial interface, and then you can define the sub-interfaces—generally one sub-interface per PVC. Here’s an example:

```
RouterA(config)#int s0
RouterA(config-if)#encapsulation frame-relay
RouterA(config-if)#int s0.?
<0-4294967295> Serial interface number
RouterA(config-if)#int s0.16 ?
multipoint Treat as a multipoint link
point-to-point Treat as a point-to-point link
RouterA(config-if)#int s0.16 point-to-point
```

Multipoint This is when the router is the center of a star of virtual circuits that are using a single subnet for all routers’ serial interfaces connected to the frame switch. You’ll usually find this implemented with the hub router in this mode and the spoke routers in physical interface (always point-to-point) or point-to-point sub-interface mode.

Implement and troubleshoot PPPoE

A family of technologies that provide digital data transmission over the wires of a local telephone network. Typically, the download speed of consumer DSL services ranges from 256 to

24,000 kbps, depending on DSL technology, line conditions, and the service level that has been implemented. DSL implementations often use PPPoE or PPPoA. Both implementations offer standard PPP features such as authentication, encryption, and compression. PPPoE is a network protocol for encapsulating PPP frames in Ethernet frames. PPPoA is a network protocol for encapsulating PPP frames in ATM adaptation layer 5 (AAL5).

Point-to-Point Protocol over Ethernet encapsulates PPP frames in Ethernet frames and is usually used in conjunction with ADSL services. It gives you a lot of the familiar PPP features like authentication, encryption, and compression, but there's a downside—it has a lower maximum transmission unit (MTU) than standard Ethernet does, and if your firewall isn't solidly configured, this little attribute can really give you some grief!

Still somewhat popular in the United States, PPPoE on Ethernet's main feature is that it adds a direct connection to Ethernet interfaces while providing DSL support as well. It's often used by many hosts on a shared Ethernet interface for opening PPP sessions to various destinations via at least one bridging modem.

In a modern HFC network, typically 500 to 2,000 active data subscribers are connected to a certain cable network segment, all sharing the upstream and downstream bandwidth. (Hybrid fibre-coaxial, or HFC, is a telecommunications industry term for a network that incorporates both optical fiber and coaxial cable to create a broadband network.) The actual bandwidth for Internet service over a cable TV (CATV) line can be up to about 27Mbps on the download path to the subscriber, with about 2.5Mbps of bandwidth on the upload path. Typically, users get an access speed from 256Kbps to 6Mbps. This data rate varies greatly throughout the U.S.