Foundation Topics

Accessing the Cisco Catalyst 2960 Switch CLI

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

Cisco Catalyst switches also support other methods to both monitor and configure a switch. For example, a switch can provide a web interface, so that an engineer can open a web browser to connect to a web server running in the switch. Switches also can be controlled and operated using network management software, as discussed briefly in the ICND2 book.

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

Cisco Catalyst Switches and the 2960 Switch

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

Cisco positions the 2960 series (family) of switches as full-featured, low-cost wiring closet switches for enterprises. That means that you would expect to use 2960 switches as access switches, as shown in Figure 6-12 in Chapter 6, "Building Ethernet LANs with Switches."

Figure 7-1 shows a photo of the 2960 switch series from Cisco. Each switch is a different specific model of switch inside the 2960 series. For example, three of the five switches have 48 RJ-45 UTP 10/100 ports, meaning that these ports can autonegotiate the use of 10BASE-T or 100BASE-T Ethernet. These switches also have a few 10/100/1000 interfaces on the right, intended to connect to the core of an enterprise campus LAN.

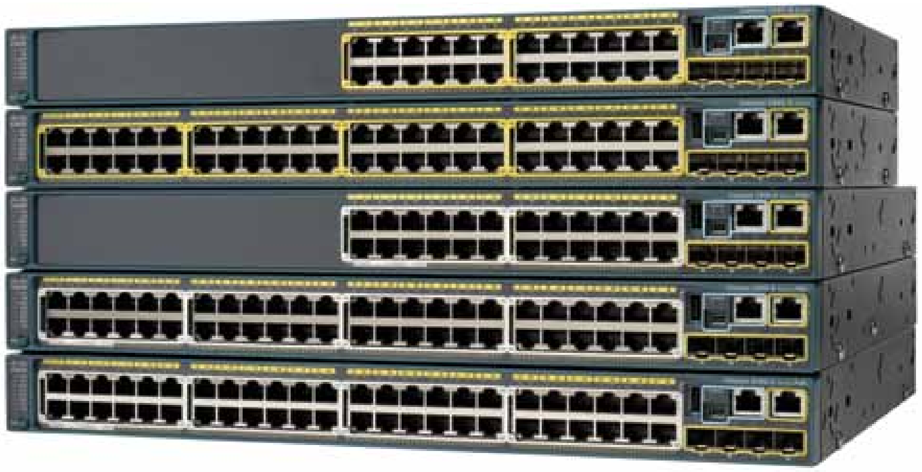


Figure 7-1 *Cisco 2960 Catalyst Switch Series*

Cisco refers to a switch's physical connectors as either *interfaces* or *ports*. Each interface has a number in the style *x*/*y*, where *x* and *y* are two different numbers. On a 2960, the numbering of 10/100 interfaces starts at 0/1, the second is 0/2, and so on. The interfaces also have names; for example, "interface FastEthernet 0/1" is the first of the 10/100 interfaces. Any Gigabit-capable interfaces would be called "GigabitEthernet" interfaces. For example, the first 10/100/1000 interface on a 2960 would be "interface GigabitEthernet 1/1."

Switch Status from LEDs

When an engineer needs to examine how a switch is working to verify its current status and to troubleshoot any problems, the vast majority of the time is spent using commands from the Cisco IOS CLI. However, the switch hardware does include several LEDs that provide some status and troubleshooting information, both during the time right after the switch has been powered on and during ongoing operations. Before moving on to discuss the CLI, this brief section examines the switch LEDs and their meanings.

Most Cisco Catalyst switches have some LEDs, including an LED for each physical Ethernet interface. For example, Figure 7-2 shows the front of a 2960 series switch, with five LEDs on the left, one LED over each port, and a mode button.

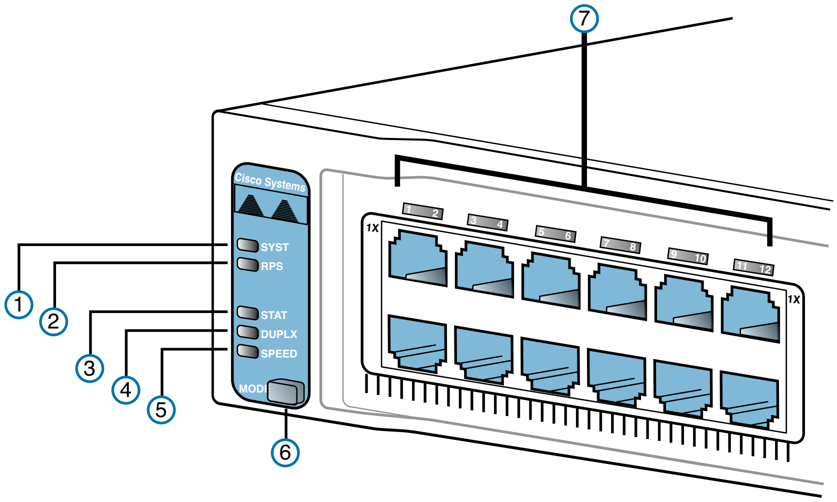


Figure 7-2 *2960 LEDs and a Mode Button*

The figure points out the various LEDs, with various meanings. Table 7-1 summarizes the LEDs, and additional explanations follow the table.

Table 7-1 LEDs and One Button in Figure 7-2

Number in Figure 7-2	Name	Description
1	SYST (system)	Implies the overall system status.
2	RPS (Redundant Power Supply)	Suggests the status of the extra (redundant) power supply.
3	STAT (Status)	If on (green), implies that each port LED implies that port's status.
4	DUPLX (duplex)	If on (green), each port LED implies that port's duplex (on/green is full; off means half).

Number in Figure 7-2	Name	Description
5	SPEED	If on (green), each port LED implies the speed of that port, as follows: off means 10 Mbps, solid green means 100 Mbps, and flashing green means 1 Gbps.
6	MODE	A button that cycles the meaning of the LEDs through three states (STAT, DUPLX, SPEED).
7	Port	LED that has different meanings, depending on the port mode as toggled using the mode button.

A few specific examples can help make sense of the LEDs. For example, consider the SYST LED for a moment. This LED provides a quick overall status of the switch, with three simple states on most 2960 switch models:

- **Off:** The switch is not powered on.
- **On (green):** The switch is powered on and operational (Cisco IOS has been loaded).
- **On (amber):** The system has power, but is not functioning properly.

So, a quick look at the SYST LED on the switch tells you whether the switch is working and, if it isn't, whether this is because of a loss of power (the SYST LED is off) or some kind of problem loading IOS (LED amber). In this last case, the typical response is to power the switch off and back on again. If the same failure occurs, a call to the Cisco Technical Assistance Center (TAC) is typically the next step.

Besides the straightforward SYST LED, the port LEDs—the LEDs sitting above or below each Ethernet port—mean something different depending on which of three port LED modes is currently used on the switch. The switches have a mode button (labeled with the number 6 in Figure 7-2) that, when pressed, cycles the port LEDs through three modes: STAT, DUPLX, and SPEED. The current port LED mode is signified by a solid green STAT, DUPLX, or SPEED LED (the lower three LEDs on the left part of Figure 7-2, labeled 3, 4, and 5). To move to another port LED mode, the engineer simply presses the mode button another time or two.

For example, in STAT (status) mode, each port LED implies the state of the matching port, as follows:

- **Off:** The link is currently not working (including if shut down).
- **Solid green:** The link is working, but there's no current traffic.
- **Flashing green:** The link is working, and traffic is currently passing over the interface.
- **Flashing amber:** The port is blocked by spanning tree.

In contrast, in SPEED port LED mode, the port LEDs imply the operating speed of the interface, with an unlit LED meaning 10 Mbps, a solid green light meaning 100 Mbps, and flashing green meaning 1000 Mbps (1 Gbps).

The particular details of how each LED works differ between different Cisco switch families and with different models inside the same switch family. So, memorizing the specific meaning of particular LED combinations is probably not required, and this chapter does not attempt to cover all combinations for even a single switch. However, it is important to remember the general ideas, the concept of a mode button that changes the meaning of the port LEDs, and the three meanings of the SYST LED mentioned earlier in this section.

The vast majority of the time, switches power up just fine and load Cisco IOS, and then the engineer simply accesses the CLI to operate and examine the switch. Next, the chapter focuses on the details of how to access the CLI.

Accessing the Cisco IOS CLI

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

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

The switch CLI can be accessed through three popular methods—the console, Telnet, and Secure Shell (SSH). Two of these methods (Telnet and SSH) use the IP network in which the switch resides to reach the switch. The console is a physical port built specifically to allow access to the CLI. Figure 7-3 depicts the options.

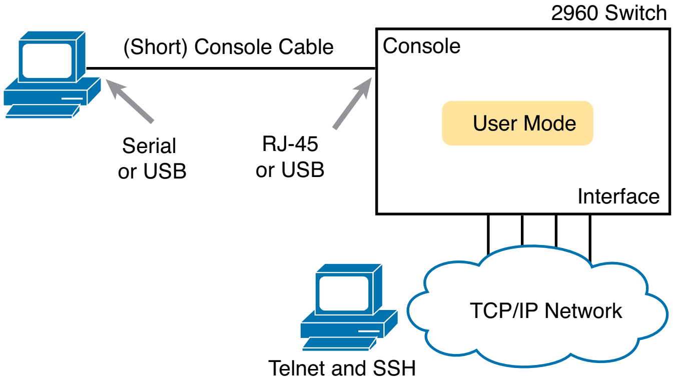


Figure 7-3 *CLI Access*

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

Cabling the Console Connection

The physical console connection, both old and new, uses three main components: the physical console port on the switch, a physical serial port on the PC, and a cable that works with the console and serial ports. However, the physical cabling details have changed slowly over time, mainly because of advances and changes with PC hardware.

Older console connections use a PC serial port, a console cable, and an RJ-45 connector on the switch. The PC serial port typically has a D-shell connector (roughly rectangular) with nine pins (often called a DB-9). Older switches, as well as some current models, use an RJ-45 connector for the console port. Figure 7-4 shows the cabling on the left.

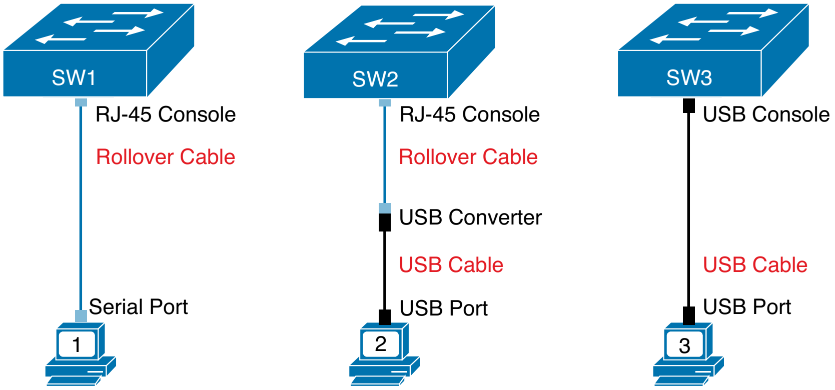


Figure 7-4 *Console Connection to a Switch*

You can use either a purpose-built console cable (which ships with new Cisco switches and routers) or make your own console cable using UTP cables and a standard RJ-45–to–DB-9 converter plug. You can buy the converter plug at most computer stores. Then, for the UTP cabling, the cable uses rollover cable pinouts, rather than any of the standard Ethernet cabling pinouts. Instead, it uses eight wires, rolling the wire at pin 1 to pin 8, pin 2 to pin 7, pin 3 to pin 6, and so on.

PCs have migrated away from using serial ports to instead use Universal Serial Bus (USB) ports for serial communications. Cisco has also begun building newer routers and switches with USB ports for console access as well. In the simplest form, you can use any USB port on the PC, with a USB cable, connected to the USB console port on the switch or router, as shown on the far right side of Figure 7-4.

The middle part of the figure shows yet another common option. Many PCs no longer have serial ports, but many existing Cisco routers and switches have only an RJ-45 console port and no USB console port. To connect such a PC to a router or switch console, you need some kind of converter that converts from the older console cable to a USB connector, as shown in the middle of Figure 7-4.

NOTE When using the USB options, you typically also need to install a software driver so that your PC's OS knows that the device on the other end of the USB connection is the console of a Cisco device.

Configuring the Terminal Emulator for the Console

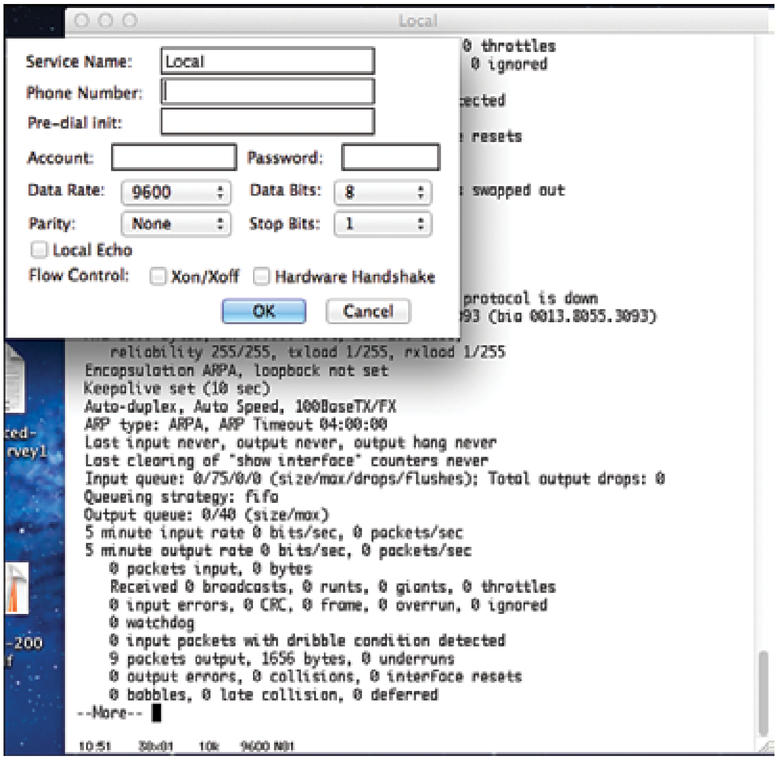
After the PC is physically connected to the console port, a terminal emulator software package must be installed and configured on the PC. The terminal emulator software treats all data as text. It accepts the text typed by the user and sends it over the console connection to the switch. Similarly, any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

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

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

- No parity bits
- 1 stop bit

Figure 7-5 shows one such terminal emulator, Zterm Pro. The image shows the window created by the emulator software in the background, with some output of a **show** command. The foreground, in the upper left, shows a settings window that lists the default console settings as listed just before this paragraph.



Service Name: Local
Phone Number:
Pre-dial init:
Account: Password:
Data Rate: 9600 Data Bits: 8
Parity: None Stop Bits: 1
□ Local Echo
Flow Control: □ Xon/Xoff □ Hardware Handshake

```
                                          0 throttles
                                           0 ignored

                                         ected

                                         resets

                                         swapped out

                                         protocol is down
                                        93 (bia 0013.8055.3093)
     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 never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (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
     9 packets output, 1656 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
 --More--
```

Figure 7-5 *Terminal Settings for Console Access*

Accessing the CLI with Telnet and SSH

The TCP/IP Telnet application allows a terminal emulator to communicate with another willing device. The process works much like what happens with an emulator on a PC connected to the console, except that the data flows over a TCP/IP network, instead of over a console cable. However, Telnet uses an IP network to send and receive the data, rather than a specialized cable and physical port on the device. The Telnet application protocols call the terminal emulator a *Telnet client* and the device that listens for commands and replies to them a *Telnet server*. Telnet is a TCP-based application layer protocol that uses well-known port 23.

To use Telnet, the user must install a Telnet client software package on his or her PC. (As mentioned earlier, most terminal emulator software packages today include both Telnet and SSH client functions.) The switch runs Telnet server software by default, but the switch does need to have an IP address configured so that it can send and receive IP packets. (Chapter 8, "Configuring Ethernet Switching," covers switch IP address configuration in greater detail.) Additionally, the network between the PC and switch needs to be up and working so that the PC and switch can exchange IP packets.

Many network engineers habitually use a Telnet client to monitor switches. The engineer can sit at his or her desk without having to walk to another part of the building—or go to another state or country—and still get into the CLI of that device.

While Telnet works well, many network engineers instead use *Secure Shell (SSH)* to overcome a serious security problem with Telnet. Telnet sends all data (including any username and password for login to the switch) as clear-text data. SSH encrypts the contents of all messages, including the passwords, avoiding the possibility of someone capturing packets in the network and stealing the password to network devices.

Secure Shell (SSH) does the same basic things as Telnet, but with added security. The user uses a terminal emulator that supports SSH. Like Telnet, SSH uses TCP, using well-known port 22 instead of Telnet's 23. As with Telnet, the SSH server (on the switch) receives the text from each SSH client, processes the text as a command, and sends messages back to the client.

Password Security for CLI Access

A Cisco switch, with default settings, remains relatively secure when locked inside a wiring closet, because by default, a switch allows console access only. However, when you enable Telnet and/or SSH access, you need to enable password security so that only authorized people have access to the CLI. Also, just to be safe, you should password-protect the console as well.

To add basic password checking for the console and for Telnet, the engineer needs to configure a couple of basic commands. The configuration process is covered a little later in this chapter, but you can get a general idea of the commands by looking in the last column of Table 7-2. The table lists the two commands that configure the console and vty passwords (used by Telnet users). After it is configured, the switch supplies a simple password prompt (as a result of the **login** command), and the switch expects the user to enter the password listed in the **password** command.

Table 7-2 CLI Password Configuration: Console and Telnet

Access From	Password Type	Sample Configuration
Console	Console password	**line console 0** **login** **password faith**
Telnet	vty password	**line vty 0 15** **login** **password love**

Cisco switches refer to the console as a console line—specifically, console line 0. Similarly, switches support 16 concurrent Telnet sessions, referenced as virtual terminal (vty) lines 0 through 15. (The term *vty* refers to an old name for terminal emulators.) The **line vty 0 15** configuration command tells the switch that the commands that follow apply to all 16 possible concurrent virtual terminal connections to the switch (0 through 15), which includes Telnet as well as SSH access.

After adding the configuration shown in Table 7-2, a user connecting to the console would be prompted for a password, and he or she would have to supply the word **faith** in this case. New Telnet users would also be prompted for a password, with **love** being the required password. Also, with this configuration, no username is required—just a simple password.

Configuring SSH requires a little more effort than the console and Telnet password configuration examples shown in Table 7-3. SSH uses public key cryptography to exchange a shared session key, which in turn is used for encryption. Additionally, SSH requires slightly better login security, requiring at least a password and a username. The section "Configuring Usernames and Secure Shell (SSH)" in Chapter 8 shows the configuration steps and a sample configuration to support SSH.

User and Enable (Privileged) Modes

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

Cisco IOS supports a more powerful EXEC mode called *enable* mode (also known as *privileged* mode or *privileged EXEC* mode). Enable mode gets its name from the **enable** command, which moves the user from user mode to enable mode, as shown in Figure 7-6. The other name for this mode, privileged mode, refers to the fact that powerful (or privileged) commands can be executed there. For example, you can use the **reload** command, which tells the switch to reinitialize or reboot Cisco IOS, only from enable mode.

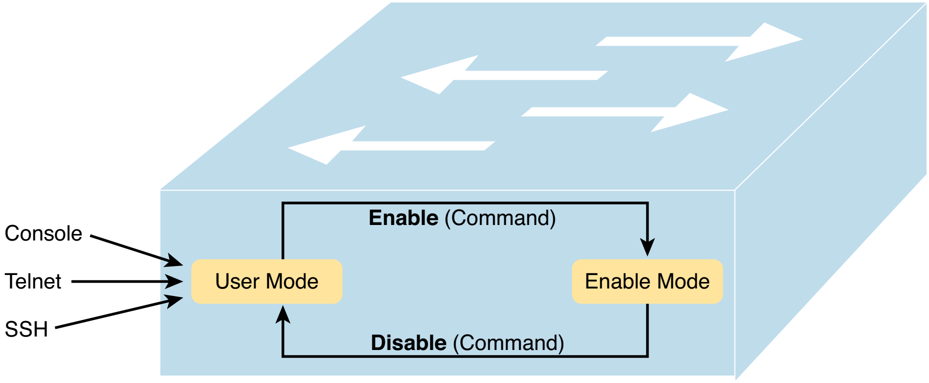


Figure 7-6 *User and Privileged Modes*

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

Example 7-1 shows the output that you could see in a Telnet window. In this case, the user connects with Telnet and tries the **reload** command. The **reload** command tells the switch to reinitialize or reboot Cisco IOS. IOS allows this powerful command to be used only from enable mode, so in the example, IOS rejects the **reload** command when used in user mode and accepts the command when the user is in enable mode.

Example 7-1 *Navigating Between Different EXEC Modes on Switch Certskills1*

```
Press RETURN to get started.

User Access Verification

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

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

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

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

So far, this chapter has pointed out some of the first things you should know when unpacking and installing a switch. The switch will work without any configuration—just plug in the power and Ethernet cables, and it works. However, you should at least connect to the switch console port and configure passwords for the console, Telnet, SSH, and the enable secret password. Next, this chapter examines some of the CLI features that exist regardless of how you access the CLI.

CLI Help Features

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

Table 7-3 summarizes command-recall help options available at the CLI. Note that, in the first column, *command* represents any command. Likewise, *parm* represents a command's parameter. For example, the third row lists *command* **?**, which means that commands such as **show ?** and **copy ?** would list help for the **show** and **copy** commands, respectively.

Table 7-3 Cisco IOS Software Command Help

What You Enter	What Help You Get
?	Help for all commands available in this mode.
help	Text describing how to get help. No actual command help is given.
Command **?**	Text help describing all the first parameter options for the command.
com?	A list of commands that start with **com**.
*command parm***?**	This style of help lists all parameters beginning with the **parameter typed so far.** (Notice that there is no space between *parm* and the **?**.)
command parm<Tab>	If you press the Tab key midword, the CLI either spells the rest of this parameter at the command line or does nothing. If the CLI does nothing, it means that this string of characters represents more than one possible next parameter, so the CLI does not know which one to spell out.
command parm1 **?**	If a space is inserted before the question mark, the CLI lists all the next parameters and gives a brief explanation of each.

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

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

Cisco IOS stores the commands that you enter in a history buffer, storing ten commands by default. The CLI allows you to move backward and forward in the historical list of commands and then edit the command before reissuing it. These key sequences can help you use the CLI more quickly on the exams. Table 7-4 lists the commands used to manipulate previously entered commands.

Table 7-4 Key Sequences for Command Edit and Recall

Keyboard Command	What Happens
Up arrow or Ctrl-P	This displays the most recently used command. If you press it again, the next most recent command appears, until the history buffer is exhausted. (The P stands for previous.)
Down arrow or Ctrl-N	If you have gone too far back into the history buffer, these keys take you forward to the more recently entered commands. (The N stands for next.)
Left arrow or Ctrl-B	This moves the cursor backward in the currently displayed command without deleting characters. (The B stands for back.)
Right arrow or Ctrl-F	This moves the cursor forward in the currently displayed command without deleting characters. (The F stands for forward.)
Backspace	This moves the cursor backward in the currently displayed command, deleting characters.
Ctrl-A	This moves the cursor directly to the first character of the currently displayed command.
Ctrl-E	This moves the cursor directly to the end of the currently displayed command.
Ctrl-R	This redisplays the command line with all characters. It's useful when messages clutter the screen.
Ctrl-D	This deletes a single character.
Ctrl-Shift-6	Interrupts the current command.

The debug and show Commands

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

The **debug** command has a similar role as compared with the **show** command. Like the **show** command, **debug** has many options. However, instead of just listing messages about the current status, the **debug** command asks the switch to continue monitoring different processes in the switch. The switch then sends ongoing messages to the user when different events occur.

The effects of the **show** and **debug** commands can be compared to a photograph (**show** command) and a movie (**debug** command). A **show** command shows what's true at a single point in time, and it takes less effort. A **debug** command shows what's true over time, but it requires more effort. As a result, the **debug** command requires more CPU cycles, but it lets you watch what is happening in a switch while it is happening.

Cisco IOS handles the messages from the **show** and **debug** commands very differently. IOS sends the output of **show** commands to the user that issued the **show** command, and to no other users. However, IOS reacts to **debug** commands by creating log messages related to that **debug** command's options. Any user logged in can choose to view the log messages, just by using the **terminal monitor** command from enable mode.

IOS also treats the **show** command as a very short-lived event and the **debug** command as an ongoing task. The options enabled by a single **debug** command are not disabled until the user takes action or until the switch is reloaded. A **reload** of the switch disables all currently enabled debug options. To disable a single debug option, repeat the same **debug** command with those options, prefaced by the word **no**. For example, if the **debug spanning-tree** command had been issued earlier, issue the **no debug spanning-tree** command to disable that same debug. Also, the **no debug all** and **undebug all** commands disable all currently enabled debugs.

Be aware that some **debug** options create so many messages that Cisco IOS cannot process them all, possibly resulting in a crash of Cisco IOS. You might want to check the current switch CPU utilization with the **show process** command before issuing any **debug** command. To be more careful, before enabling an unfamiliar **debug** command option, issue a **no debug all** command and then issue the **debug** that you want to use. Then quickly retrieve the **no debug all** command using the up arrow or Ctrl-P key sequence twice. If the debug quickly degrades switch performance, the switch might be too busy to listen to what you are typing. The process described in this paragraph saves a bit of typing and can be the difference between preventing the switch from failing or not.

Configuring Cisco IOS Software

You must understand how to configure a Cisco switch to succeed on the exam and in real networking jobs. This section covers the basic configuration processes, including the concept of a configuration file and the locations in which the configuration files can be stored. Although this section focuses on the configuration process, and not on the configuration commands themselves, you should know all the commands covered in this chapter for the exams, in addition to the configuration processes.

C*onfiguration mode* is another mode for the Cisco CLI, similar to user mode and privileged mode. User mode lets you issue nondisruptive commands and displays some information. Privileged mode supports a superset of commands compared to user mode, including commands that might harm the switch. However, none of the commands in user or privileged mode changes the switch's configuration. Configuration mode accepts *configuration commands*—commands that tell the switch the details of what to do and how to do it. Figure 7-7 illustrates the relationships among configuration mode, user EXEC mode, and privileged EXEC mode.

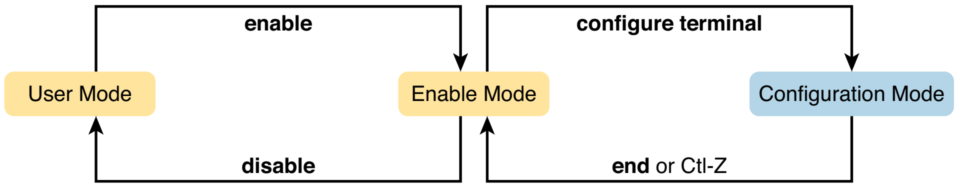


Figure 7-7 *CLI Configuration Mode Versus EXEC Modes*

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

Configuration Submodes and Contexts

Configuration mode itself contains a multitude of subcommand modes. *Context-setting commands* move you from one configuration subcommand mode, or context, to another. These context-setting commands tell the switch the topic about which you will enter the next few configuration commands. More importantly, the context tells the switch the topic you care about right now, so when you use the **?** to get help, the switch gives you help about that topic only.

NOTE *Context-setting* is not a Cisco term—it's just a term used here to help make sense of configuration mode.

The **interface** command is one of the most commonly used context-setting configuration commands. For example, the CLI user could enter interface configuration mode by entering the **interface FastEthernet 0/1** configuration command. Asking for help in interface configuration mode displays only commands that are useful when configuring Ethernet interfaces. Commands used in this context are called *subcommands*—or, in this specific case, *interface subcommands*. When you begin practicing with the CLI with real equipment, the navigation between modes can become natural. For now, consider Example 7-2, which shows the following:

- Movement from enable mode to global configuration mode by using the **configure terminal** EXEC command
- Using a **hostname Fred** global configuration command to configure the switch's name
- Movement from global configuration mode to console line configuration mode (using the **line console 0** command)
- Setting the console's simple password to **hope** (using the **password hope** line subcommand)
- Movement from console configuration mode to interface configuration mode (using the **interface** command)
- Setting the speed to 100 Mbps for interface Fa0/1 (using the **speed 100** interface subcommand)
- Movement from interface configuration mode back to global configuration mode (using the **exit** command)

Example 7-2 *Navigating Between Different Configuration Modes*

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

The text inside parentheses in the command prompt identifies the configuration mode. For example, the first command prompt after you enter configuration mode lists (config), meaning global configuration mode. After the **line console 0** command, the text expands to (config-line), meaning line configuration mode. Table 7-5 shows the most common command prompts in configuration mode, the names of those modes, and the context-setting commands used to reach those modes.

Key Topic

Table 7-5 Common Switch Configuration Modes

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

You should practice until you become comfortable moving between the different configuration modes, back to enable mode, and then back into the configuration modes. However, you can learn these skills just doing labs about the topics in later chapters of the book. For now, Figure 7-8 shows most of the navigation between global configuration mode and the four configuration submodes listed in Table 7-5.

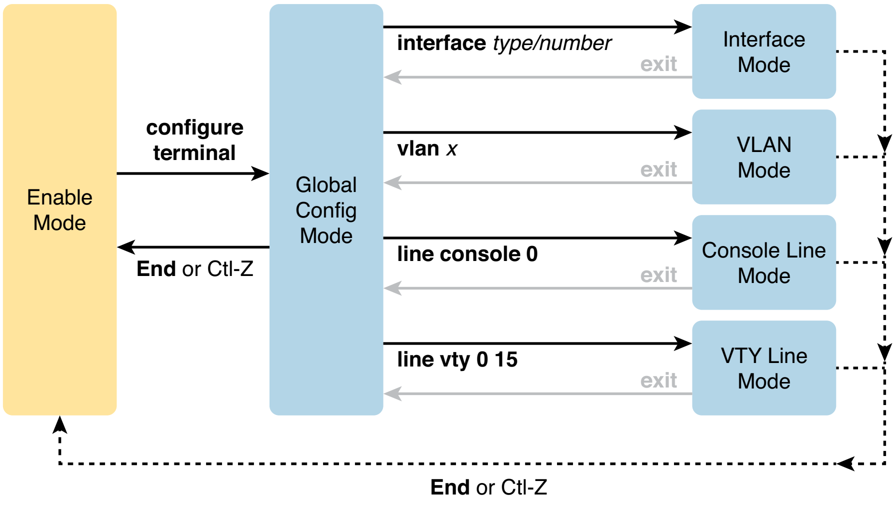


Figure 7-8 *Navigation In and Out of Switch Configuration Modes*

NOTE You can also move directly from one configuration submode to another, without first using the **exit** command to move back to global configuration mode. Just use the commands listed in bold in the center of the figure.

No set rules exist for what commands are global commands or subcommands. Generally, however, when multiple instances of a parameter can be set in a single switch, the command used to set the parameter is likely a configuration subcommand. Items that are set once for the entire switch are likely global commands. For example, the **hostname** command is a global command because there is only one host name per switch. Conversely, the **duplex** command is an interface subcommand to allow the switch to use a different setting on the different interfaces.

Storing Switch Configuration Files

When you configure a switch, it needs to use the configuration. It also needs to be able to retain the configuration in case the switch loses power. Cisco switches contain random-access memory (RAM) to store data while Cisco IOS is using it, but RAM loses its contents when the switch loses power. To store information that must be retained when the switch loses power, Cisco switches use several types of more permanent memory, none of which has any moving parts. By avoiding components with moving parts (such as traditional disk drives), switches can maintain better uptime and availability.

The following list details the four main types of memory found in Cisco switches, as well as the most common use of each type:

- **RAM:** Sometimes called DRAM, for dynamic random-access memory, RAM is used by the switch just as it is used by any other computer: for working storage. The running (active) configuration file is stored here.
- **ROM:** Read-only memory (ROM) stores a bootstrap (or boothelper) program that is loaded when the switch first powers on. This bootstrap program then finds the full Cisco IOS image and manages the process of loading Cisco IOS into RAM, at which point Cisco IOS takes over operation of the switch.
- **Flash memory:** Either a chip inside the switch or a removable memory card, flash memory stores fully functional Cisco IOS images and is the default location where the switch gets its Cisco IOS at boot time. Flash memory also can be used to store any other files, including backup copies of configuration files.
- **NVRAM:** Nonvolatile RAM (NVRAM) stores the initial or startup configuration file that is used when the switch is first powered on and when the switch is reloaded.

Figure 7-9 summarizes this same information in a briefer and more convenient form for memorization and study.

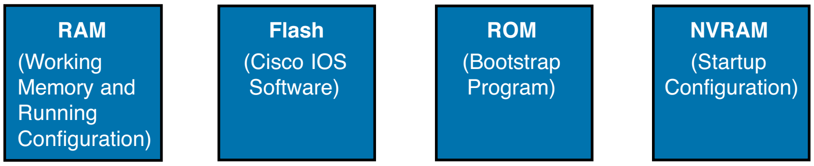


Figure 7-9 *Cisco Switch Memory Types*

Cisco IOS stores the collection of configuration commands in a *configuration file*. In fact, switches use multiple configuration files—one file for the initial configuration used when powering on, and another configuration file for the active, currently used running configuration as stored in RAM. Table 7-6 lists the names of these two files, their purpose, and their storage location.

Key Topic

Table 7-6 Names and Purposes of the Two Main Cisco IOS Configuration Files

Configuration Filename	Purpose	Where It Is Stored
Startup config	Stores the initial configuration used anytime the switch reloads Cisco IOS.	NVRAM
Running config	Stores the currently used configuration commands. This file changes dynamically when someone enters commands in configuration mode.	RAM

Essentially, when you use configuration mode, you change only the running config file. This means that the configuration example earlier in this chapter (Example 7-2) updates only the running config file. However, if the switch lost power right after that example, all that configuration would be lost. If you want to keep that configuration, you have to copy the running config file into NVRAM, overwriting the old startup config file.

Example 7-3 demonstrates that commands used in configuration mode change only the running configuration in RAM. The example shows the following concepts and steps:

Step 1. The original **hostname** command on the switch, with the startup config file matching the running config file.

Step 2. The **hostname** command changes the host name, but only in the running config file.

Step 3. The **show running-config** and **show startup-config** commands are shown, with only the **hostname** commands displayed for brevity, to make the point that the two configuration files are now different.

Example 7-3 *How Configuration Mode Commands Change the Running Config File, Not the Startup Config File*

```
! Step 1 next (two commands)
!
hannah# show running-config
! (lines omitted)
hostname hannah
! (rest of lines omitted)

hannah# show startup-config
! (lines omitted)
hostname hannah
! (rest of lines omitted)
! Step 2 next. Notice that the command prompt changes immediately after
! the hostname command.
hannah# configure terminal
hannah(config)# hostname jessie
jessie(config)# exit
! Step 3 next (two commands)
!
jessie# show running-config
! (lines omitted)
hostname jessie
! (rest of lines omitted - notice that the running configuration reflects the
!  changed hostname)
jessie# show startup-config
! (lines omitted)
hostname hannah
! (rest of lines omitted - notice that the changed configuration is not
! shown in the startup config)
```

NOTE Cisco uses the term *reload* to refer to what most PC operating systems call rebooting or restarting. In each case, it is a reinitialization of the software. The **reload** EXEC command causes a switch to reload.

Copying and Erasing Configuration Files

If you want to keep the new configuration commands you add in configuration mode (so that the changes are present the next time the system is rebooted), like the **hostname jessie** command in Example 7-3, you need to use the command **copy running-config startup-config.** This command overwrites the current startup config file with what is currently in the running configuration file.

The **copy** command can be used to copy files in a switch, most typically a configuration file or a new version of Cisco IOS Software. The most basic method for moving configuration files in and out of a switch is to use the **copy** command to copy files between RAM or NVRAM on a switch and a TFTP server. The files can be copied between any pair, as shown in Figure 7-10.

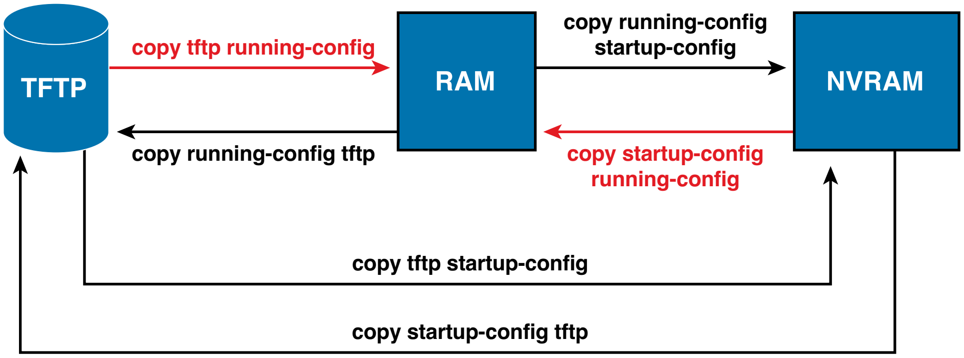


Figure 7-10 *Locations for Copying and Results from Copy Operations*

The commands for copying Cisco IOS configurations can be summarized as follows:

```
copy {tftp | running-config | startup-config} {tftp | running-config | startup-config}
```

The first set of parameters enclosed in braces ({ }) is the “from” location; the next set of parameters is the “to” location.

The **copy** command always replaces the existing file when the file is copied into NVRAM or into a TFTP server. In other words, it acts as if the destination file was erased and the new file completely replaced the old one. However, when the **copy** command copies a configuration file into the running config file in RAM, the configuration file in RAM is not replaced, but is merged instead. Effectively, any **copy** into RAM works just as if you entered the commands in the "from" configuration file in the order listed in the config file.

Who cares? Well, we do. If you change the running config and then decide that you want to revert to what's in the startup config file, the result of the **copy startup-config running-config** command might not cause the two files to actually match. One way to guarantee that the two configuration files match is to issue the **reload** command, which reloads, or reboots, the switch, which erases RAM and then copies the startup config into RAM as part of the reload process.

You can use three different commands to erase the contents of NVRAM. The **write erase** and **erase startup-config** commands are older, whereas the **erase nvram:** command is the more recent, and recommended, command. All three commands simply erase the contents of the NVRAM configuration file. Of course, if the switch is reloaded at this point, there is no initial configuration. Note that Cisco IOS does not have a command that erases the contents of the running config file. To clear out the running config file, simply erase the startup config file and then **reload** the switch.

NOTE Making a copy of all current switch and router configurations should be part of any network's overall security strategy, mainly so that you can replace a device's configuration if an attack changes the configuration.

Initial Configuration (Setup Mode)

Cisco IOS Software supports two primary methods of giving a switch an initial basic configuration—configuration mode, which has already been covered in this chapter, and setup mode. *Setup mode* leads a switch administrator by asking questions that prompt the administrator for basic configuration parameters. After the administrator answers the questions, IOS builds a configuration file, saves it as the startup config, and also loads it as the running config to start using the new configuration.

When a Cisco switch or router initializes, but the startup config file is empty, the switch or router asks the console user if he wants to use setup. Figure 7-11 shows the branches in the process. The left side of the figure, moving down, brings the user to the point at which IOS asks the user questions about what should be added to the configuration.

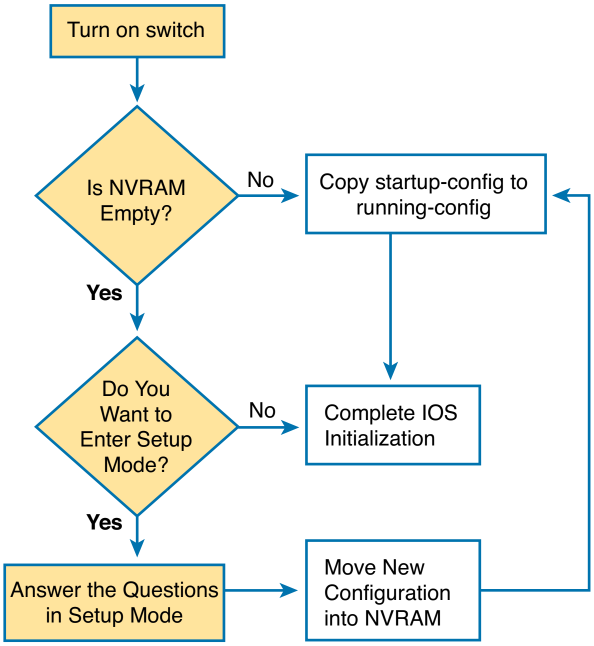

Turn on switch
Is NVRAM Empty?
No
Copy startup-config to running-config
Yes
Do You Want to Enter Setup Mode?
No
Complete IOS Initialization
Yes
Answer the Questions in Setup Mode
Move New Configuration into NVRAM

Figure 7-11 *Getting into Setup Mode*

Frankly, most network engineers never use setup mode, mainly because setup supports only a small percentage of modern switch configuration settings. However, you will still see some evidence of setup, because when you reload a switch or router that has no configuration, IOS will ask you whether you want to enter the "initial configuration dialogue" (the official term for setup mode). Just answer "no," as shown in Example 7-4, and use configuration mode to configure the device.

Example 7-4 *Initial Configuration Dialog (Setup)—Rejected*

```
--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no

Switch>
```

IOS Version and Other Reload Facts

To finish this first chapter about how Cisco IOS works, with the IOS CLI, this last topic looks at the switch **show version** command.

When a switch loads the IOS, it must do many tasks. The IOS software itself must be loaded into RAM. The IOS must become aware of the hardware available, for example, all the different LAN interfaces on the switch. After the software is loaded, the IOS keeps track of some statistics related to the current operation of the switch, like the amount of time since the IOS was last loaded and the reason why the IOS was most recently loaded.

The **show version** command lists these facts, plus many others. As you might guess from the command itself, the **show version** command does list information about the IOS, including the version of IOS software. However, as highlighted in Example 7-5, it lists many other interesting facts as well.

Example 7-5 *Example of a* **show version** *Command on a Cisco Switch*

```
SW1# show version
Cisco IOS Software, C2960 Software (C2960-LANBASEK9-M), Version 15.0(1)SE3, RELEASE
SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2012 by Cisco Systems, Inc.
Compiled Wed 30-May-12 14:26 by prod_rel_team

ROM: Bootstrap program is C2960 boot loader
BOOTLDR: C2960 Boot Loader (C2960-HBOOT-M) Version 12.2(44)SE5, RELEASE SOFTWARE (fc1)

SW1 uptime is 2 days, 22 hours, 2 minutes
System returned to ROM by power-on
System image file is "flash:c2960-lanbasek9-mz.150-1.SE3.bin"


This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use…
! Lines omitted for brevity

cisco WS-C2960-24TT-L (PowerPC405) processor (revision P0) with 65536K bytes of memory.
Processor board ID FCQ1621X6QC
Last reset from power-on
1 Virtual Ethernet interface
24 FastEthernet interfaces
2 Gigabit Ethernet interfaces
The password-recovery mechanism is enabled.

64K bytes of flash-simulated non-volatile configuration memory.
Base ethernet MAC Address       : 18:33:9D:7B:13:80
Motherboard assembly number     : 73-11473-11
Power supply part number        : 341-0097-03
Motherboard serial number       : FCQ162103ZL
Power supply serial number      : ALD1619B37W
Model revision number           : P0
Motherboard revision number     : A0
Model number                    : WS-C2960-24TT-L
System serial number            : FCQ1621X6QC
Top Assembly Part Number        : 800-29859-06
Top Assembly Revision Number    : C0
Version ID                      : V10
CLEI Code Number                : COMCX00ARB
Hardware Board Revision Number  : 0x01


Switch Ports Model              SW Version            SW Image
------ ----- -----              ----------            ----------
*    1 26    WS-C2960-24TT-L    15.0(1)SE3            C2960-LANBASEK9-M


Configuration register is 0xF
```

Working through the highlighted parts of the example, top to bottom, this command lists

- The IOS version
- Time since last load of the IOS
- Reason for last load of the IOS
- Number of Fast Ethernet interfaces (24)
- Number of Gigabit Ethernet interfaces (2)
- Switch model number

Review Activities

Chapter Summary

- Cisco uses the concept of a command-line interface (CLI) with its router products and most of its Catalyst LAN switch products. The CLI is a text-based interface in which the user, typically a network engineer, enters a text command and presses Enter.
- Like any other piece of computer hardware, Cisco switches need some kind of operating system software. Cisco calls this OS the Internetwork Operating System (IOS).
- The switch CLI can be accessed through four popular methods: the console, Telnet, Secure Shell (SSH), and the AUX port. Two of these methods (Telnet and SSH) use the IP network in which the switch resides to reach the switch. The console is a physical port built specifically to allow access to the CLI. The AUX port is designed to be connected to a modem.
- The physical console connection, both old and new, uses three main components: the physical console port on the switch, a physical serial port on the PC, and a cable that works with the console and serial ports.
- When the PC is physically connected to the console port, a terminal emulator software package must be installed and configured on the PC. The terminal emulator software treats all data as text. It accepts the text typed by the user and sends it over the console connection to the switch. Similarly, any bits coming into the PC over the console connection is displayed as text for the user to read.
- The emulator must be configured to use the PC's serial port to match the settings on the switch's console port settings. The default console port settings on a switch are as follows; note that the last three parameters are referred to collectively as "8N1.":
  - 9600 bits per second
  - No hardware flow control
  - 8-bit ASCII
  - No parity bits
  - 1 stop bit
- A Cisco switch, with default settings, remains relatively secure when locked inside a wiring closet because, by default, a switch allows console access only. However, when you enable Telnet and/or SSH access, you must enable password security so that only authorized people have access to the CLI.
- User EXEC mode, sometimes also called *user mode*, enables the user to look around but not break anything. The "EXEC mode" part of the name refers to the fact that in this mode, when you enter a command, the switch executes the command and then displays messages that describe the command's results.
- Cisco IOS supports a more powerful EXEC mode called *enable* mode (also known as *privileged* mode or *privileged EXEC* mode). Enable mode gets its name from the **enable** command, which moves the user from user mode to enable mode.
- The information supplied by using help depends on the CLI mode. For example, when **?** is entered in user mode, the commands allowed in user mode are displayed, but commands available only in enable mode (not in user mode) are not displayed. Also, help is available in configuration mode, which is the mode used to configure the switch. In fact, configuration mode has many different subconfiguration modes.
- Cisco IOS stores the commands that you enter in a history buffer, storing 10 commands by default. The CLI enables you to move backward and forward in the historical list of commands and then edit the command before reissuing it.

- The **show** command has a large variety of options, and with those options, you can find the status of almost every feature of Cisco IOS.
- The **debug** command has a similar role as the **show** command. Like the **show** command, **debug** has many options; however, instead of just listing messages about the current status, the **debug** command asks the switch to continue monitoring different processes in the switch. The switch then sends ongoing messages to the user when different events occur.
- *Configuration mode* is another mode for the Cisco CLI, similar to user mode and privileged mode. User mode lets you issue nondisruptive commands and displays some information. Privileged mode supports a superset of commands compared to user mode, including commands that might harm the switch. However, none of the commands in user or privileged mode changes the switch's configuration. Configuration mode accepts configuration commands that tell the switch the details of what to do and how to do it.
- Configuration mode itself contains a multitude of subcommand modes. *Context-setting commands* move you from one configuration subcommand mode, or context, to another.
- The **interface** command is one of the most commonly used context-setting configuration commands. For example, the CLI user could enter interface configuration mode by entering the **interface FastEthernet 0/1** configuration command.
- The startup-config file stores the initial configuration used any time the switch reloads Cisco IOS. It is stored in nonvolatile RAM (NVRAM).
- The running-config file stores the currently used configuration commands. This file changes dynamically when someone enters commands in the configuration mode. It is stored in RAM.
- If you want to keep the new configuration commands you add in configuration mode (so that the changes are present the next time the system is rebooted), you must use the command **copy running-config startup-config**.
- *Setup mode* leads a switch administrator by asking questions that prompt the administrator for basic configuration parameters. After the administrator answers the questions, IOS builds a configuration file, saves it as the startup-config, and loads it as the running-config to start using the new configuration.
- The show version command lists the following about the IOS:
  - The IOS version
  - Time since last load of the IOS
  - Reason for last load of the IOS
  - Number of FastEthernet interfaces
  - Number of GigabitEthernet interfaces
  - Switch model number

Review Questions

Answer these review questions. You can find the answers at the bottom of the last page of the chapter. For thorough explanations, see DVD Appendix C, "Answers to Review Questions."

1. In what modes can you execute the command **show mac address-table**? (Choose two answers.)

   A. User mode

   B. Enable mode

   C. Global configuration mode

   D. Interface configuration mode

2. In which of the following modes of the CLI could you issue the command **reload** to reboot the switch?

   A. User mode

   B. Enable mode

   C. Global configuration mode

   D. Interface configuration mode

3. Which of the following is a difference between Telnet and SSH as supported by a Cisco switch?

   A. SSH encrypts the passwords used at login, but not other traffic; Telnet encrypts nothing.

   B. SSH encrypts all data exchange, including login passwords; Telnet encrypts nothing.

   C. Telnet is used from Microsoft operating systems, and SSH is used from UNIX and Linux operating systems.

   D. Telnet encrypts only password exchanges; SSH encrypts all data exchanges.

4. What type of switch memory is used to store the configuration used by the switch when it is up and working?

   A. RAM

   B. ROM

   C. Flash

   D. NVRAM

   E. Bubble

5. What command copies the configuration from RAM into NVRAM?

   A. **copy running-config tftp**

   B. **copy tftp running-config**

   C. **copy running-config start-up-config**

   D. **copy start-up-config running-config**

   E. **copy startup-config running-config**

   F. **copy running-config startup-config**

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

   A. Using the **exit** command once

   B. Using the **end** command once

   C. Pressing the Ctrl-Z key sequence once

   D. Using the **quit** command

Review All the Key Topics

Review the most important topics from this chapter, noted with the Key Topic icon. Table 7-7 lists these key topics and where each is discussed.

Key Topic

Table 7-7 Key Topics for Chapter 7

Key Topic Element	Description	Page Number
Figure 7-4	Cabling options for a console connection	153
List	A Cisco switch's default console port settings	153
Table 7-5	A list of configuration mode prompts, the name of the configuration mode, and the command used to reach each mode	161
Figure 7-9	Types of memory in a switch	162
Table 7-6	The names and purposes of the two configuration files in a switch or router	163

Complete the Tables and Lists from Memory

Print a copy of DVD Appendix M, "Memory Tables," or at least the section for this chapter, and complete the tables and lists from memory. DVD Appendix N, "Memory Tables Answer Key," includes completed tables and lists for you to check your work.

Definitions of Key Terms

After your first reading of the chapter, try to define these key terms, but do not be concerned about getting them all correct at that time. Chapter 30 directs you in how to use these terms for late-stage preparation for the exam.

command-line interface (CLI), Telnet, Secure Shell (SSH), enable mode, user mode, configuration mode, startup config file, running config file

Command References

Table 7-8 lists and briefly describes the configuration commands used in this chapter.

Table 7-8 Chapter 7 Configuration Commands

Command	Mode and Purpose
line console 0	Global command that changes the context to console configuration mode.
line vty *1st-vty last-vty*	Global command that changes the context to vty configuration mode for the range of vty lines listed in the command.
login	Line (console and vty) configuration mode. Tells IOS to prompt for a password (no username).
password *pass-value*	Line (console and vty) configuration mode. Lists the password required if the **login** command (with no other parameters) is configured.
interface *type port-number*	Global command that changes the context to interface mode—for example, **interface FastEthernet 0/1**.
hostname *name*	Global command that sets this switch's host name, which is also used as the first part of the switch's command prompt.
exit	Moves back to the next higher mode in configuration mode.

Command	Mode and Purpose
end	Exits configuration mode and goes back to enable mode from any of the configuration submodes.
Ctrl-Z	This is not a command, but rather a two-key combination (pressing the Ctrl key and the letter Z) that together do the same thing as the **end** command.

Table 7-9 lists and briefly describes the EXEC commands used in this chapter.

Table 7-9 Chapter 7 EXEC Command Reference

Command	Purpose
no debug all **undebug all**	Enable mode EXEC command to disable all currently enabled debugs.
terminal monitor	EXEC command that tells Cisco IOS to send a copy of all syslog messages, including debug messages, to the Telnet or SSH user who issues this command.
reload	Enable mode EXEC command that reboots the switch or router.
copy *from-location to-location*	Enable mode EXEC command that copies files from one file location to another. Locations include the startup config and running config in RAM, files TFTP and RCP servers, and flash memory.
copy running-config startup-config	Enable mode EXEC command that saves the active config, replacing the startup config file used when the switch initializes.
copy startup-config running-config	Enable mode EXEC command that merges the startup config file with the currently active config file in RAM.
show running-config	Lists the contents of the running config file.
write erase **erase startup-config**	These enable mode EXEC commands to erase the startup config file.
quit	EXEC command that disconnects the user from the CLI session.
show startup-config	Lists the contents of the startup config (initial config) file.
enable	Moves the user from user mode to enable (privileged) mode and prompts for a password if one is configured.
disable	Moves the user from enable mode to user mode.
configure terminal	Enable mode command that moves the user into configuration mode.

Answers to Review Questions:

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

Chapter 8

Configuring Ethernet Switching

Cisco LAN switches perform their core function—forwarding Ethernet frames—without any configuration. You can buy a Cisco switch, plug in the right cables to connect various devices to the switch, plug in the power cable, and the switch works. However, in most networks, the network engineer wants to configure and use various switch features.

This chapter explains a large variety of switch features, broken into two halves of the chapter. The first half of the chapter explains many switch administrative features that happen to work the same way on routers and switches; this chapter keeps these common features together so that you can easily refer to them later when working with routers. The second half of the chapter shows how to configure some switch-specific features, many of which impact how a switch forwards frames.

This chapter covers the following exam topics:

IP Routing Technologies

- Configure SVI interfaces

Network Device Security

- Configure and verify network device security features such as
  - Device password security
  - Enable secret vs enable
  - SSH
  - VTYs
  - Service password
  - Describe external authentication methods
- Configure and verify Switch Port Security features such as
  - Sticky MAC
  - MAC address limitation
  - Static / dynamic
  - Violation modes
    - Err disable
    - Shutdown
    - Protect restrict
  - Shutdown unused ports
  - Err disable recovery
- Assign unused ports to an unused VLAN

Foundation Topics

Configuration of Features in Common with Routers

This first of the two major sections of this chapter examines the configuration of several features that are configured the exact same way on both switches and routers. In particular, this section examines how to secure access to the CLI, plus various settings for the console. Note that this section will refer to only switches, and not routers, but the commands apply to both.

Securing the Switch CLI

The first step to securing a switch is to secure access to the CLI. Securing the CLI includes protecting access to enable mode, because from enable mode, an attacker could reload the switch or change the configuration. At the same time, protecting user mode is also important, because attackers can see the status of the switch, learn about the network, and find new ways to attack the network.

For example, consider a user who accesses a switch from the console. The default console configuration settings allow a console user to reach both user mode and enable mode without supplying a password. These defaults make some sense, because when you use the console, you are typically sitting near or next to the switch. If you can touch the switch, even if the console had all the available password protections, you could still perform the switch password recovery/reset procedure in five minutes anyway and get into the switch. So, by default, console access is open. However, most network engineers add login security to the console as well.

NOTE To see the password recovery/reset procedures, go to Cisco.com and search for the phrase "password recovery." The first listed item probably will be a web page with password recovery details for most every product made by Cisco.

On the other hand, the default configuration settings do not allow a vty (Telnet or SSH) session into a switch, either to user mode or to enable mode. To allow these users to reach user mode, the switch first needs a working IP configuration, as well as login security on the vty lines. To allow access to enable mode, the switch must be configured with enable mode security as well.

This section examines many of the configuration details related to accessing user and enable mode on a switch or router. Switch IP configuration is covered later in this chapter in the section "Enabling IP for Remote Access." In particular, this section covers the following topics:

- Simple password security to user mode from (a) the console and (b) Telnet
- Secure Shell (SSH)
- Password encryption
- Enable mode passwords

Securing Access with Simple Passwords

Cisco switches can protect user mode with a simple password—with no username—for console and Telnet users. Console users must supply the *console password*, as configured in console line configuration mode. Telnet users must supply the *Telnet password*, also called the vty password, so called because the configuration sits in vty line configuration mode.

Cisco switches protect enable mode for any user with the *enable password*. The user, in user mode, types the **enable** EXEC command and is prompted for this enable password; if the user types the correct password, IOS moves the user to enable mode. Figure 8-1 shows the names of these passwords and the associated configuration modes.

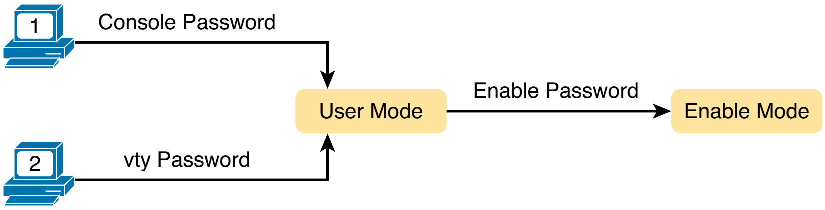


Figure 8-1 *Simple Password Security Concepts*

The configuration for these passwords does not require a lot of work. First, the console and vty password configuration uses the same two subcommands in console and vty line configuration modes, respectively. The **login** command tells IOS to use simple password security, and the **password** *password-value* command defines the password. IOS protects enable mode using the enable secret password, configured using the global command **enable secret** *password-value*.

NOTE The later section "Hiding the Enable Password" explains two options for configuring the password required by the **enable** command, as configured with the **enable secret** and **enable password** commands, and describes why the **enable secret** command is preferred.

Example 8-1 shows a sample configuration process that sets the console password, the vty (Telnet) password, the enable secret password, and a host name for the switch. The example shows the entire process, including command prompts, that provides some reminders of the different configuration modes explained in Chapter 7, "Installing and Operating Cisco LAN Switches."

Example 8-1 *Configuring Basic Passwords and a Host Name*

```
Switch> enable
Switch# configure terminal
Switch(config)# enable secret cisco
Switch(config)# hostname Emma
Emma(config)# line console 0
Emma(config-line)# password faith
Emma(config-line)# login
Emma(config-line)# exit
Emma(config)# line vty 0 15
Emma(config-line)# password love
Emma(config-line)# login
Emma(config-line)# end
Emma#
```

Because you have probably not done many configurations yourself yet, the next few paragraphs walk you through Example 8-1 a few lines at a time, to use this example as an exercise in how the CLI works. First, focus on the first four lines, with the command prompts that begin with "Switch." By the publisher's conventions, this book lists all text output displayed by the switch in nonbold text and all text typed by the user in bold text. For example, the first line shows a command prompt supplied by the switch of "Switch>" (the default prompt, by the way), with the user typing **enable**. The ">" at the end of the prompt tells us that the user is in user mode.

Those first few lines show the user beginning in user mode, and then moving to enable mode (using the **enable** EXEC command). Then the user moves to global configuration mode (by using the **configure terminal** EXEC command). As soon as the user is in global configuration mode, he enters two global configuration commands (**enable secret** and **hostname**) that add configuration that applies to the entire switch.

The first line of output in Example 8-1 that begins with “Emma” shows the beginning of the configuration of the console password. First, the user needs to enter console line configuration mode using the **line console 0** command. (With only one console per switch, it is always numbered as console 0.) Next, the **password** command lists the simple text password (faith), and the **login** command tells the switch to ask for the simple text password as defined by the password command.

Continuing through the example, the next few lines repeat the same process, but for all 16 vty lines (vty lines 0 through 15). The 16 vty lines means that the switch can accept 16 concurrent Telnet connections into the switch. As for the password, the configuration uses a different vty line password of “love.” Finally, the **end** command moves the user back to enable mode.

Moving on from the configuration, now focus on what the users will see because of the configuration in Example 8-1. A console user will now be prompted for a password (but no username), and he must type **hope**. Similarly, Telnet users will be prompted for a password, but no username, and they must type **love**. Both console and Telnet users must use the **enable** command, and use password “cisco,” to reach enable mode. And SSH users cannot yet log in to this switch, because more configuration is needed to support SSH.

Example 8-2 shows the resulting configuration in the switch named Emma. The gray lines highlight the new configuration. Note that some lines have been omitted from the full output from a switch just to reduce the volume of unrelated lines on the page.

Example 8-2 *Resulting Running Config File on Switch Emma*

```
Emma# show running-config
!
Building configuration...

Current configuration : 1333 bytes
!
version 12.2
!
hostname Emma
!
enable secret 5 $1$YXRN$11zOe1Lb0Lv/nHyTquobd.
!
spanning-tree mode pvst
spanning-tree extend system-id
!
interface FastEthernet0/1
!
interface FastEthernet0/2
!
! Several lines have been omitted here - in particular, lines for FastEthernet
! interfaces 0/3 through 0/23.
!
interface FastEthernet0/24
!
```

```
interface GigabitEthernet0/1
!
interface GigabitEthernet0/2
!
interface Vlan1
 no ip address
 no ip route-cache
!
!
line con 0
 password faith
 login
!
line vty 0 4
 password love
 login
!
line vty 5 15
 password love
 login
```

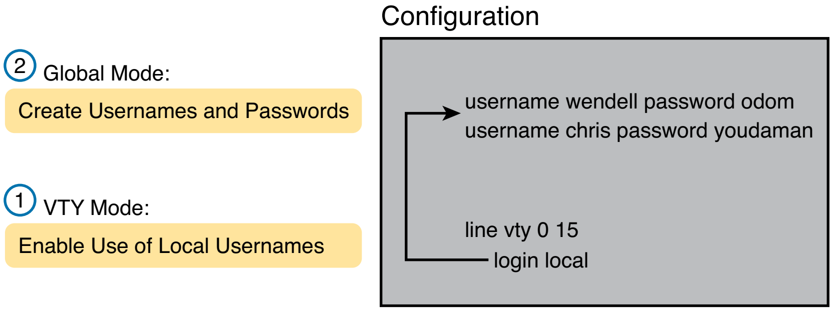
NOTE The output of the **show running-config** command, in the last six lines of Example 8-2, separates the first five vty lines (0 through 4) from the rest (5 through 15) for historical reasons.

Securing Access with Local Usernames and Passwords

A login method that uses simple text passwords (without usernames) works, but it requires that everyone know the same passwords. For example, all must know the same vty password to get access to the switch using Telnet.

Cisco switches support other login authentication methods that use a username and password so that each user has unique login details that do not have to be shared. One method configures the username/password pairs locally on the switch, and the other relies on an external server called an authentication, authorization, and accounting (AAA) server. (The server could be the same server used for logins for other servers in the network.) This book covers the configuration using locally configured usernames/passwords.

The migration from using the password-only login method to using locally configured usernames and passwords requires only some small configuration changes. The switch needs one or more **username** *name* **password** *password* global configuration commands to define the usernames and passwords. Then the vty and/or console line needs to be told to make use of a locally configured username and password (per the **login local** line subcommand). For example, Figure 8-2 shows the concept and configuration to migrate to using local usernames for Telnet users.



```
username wendell password odom
username chris password youdaman

line vty 0 15
 login local
```

Figure 8-2 *Configuring Switches to Use Local Username Login Authentication*

When a Telnet user connects to the switch configured as shown in Figure 8-2, the user will be prompted first for a username and then for a password. The username/password pair must be from the list of local usernames, or the login is rejected.

NOTE Figure 8-2 lists the configuration commands in the same order as seen in the IOS **show running-config** command.

Securing Access with External Authentication Servers

Using a local list of usernames and passwords on a switch or router works well in small networks. However, using locally configured username/password pairs means that every switch and router needs the configuration for all users who might need to log in to the devices. Then, when any changes need to happen, like an occasional change to the passwords, the configuration of all devices must be changed.

Cisco switches and routers support an alternative way to keep track of valid usernames and passwords by using an external AAA server. When using a AAA server for authentication, the switch (or router) simply sends a message to the AAA server asking whether the username and password are allowed, and the AAA server replies. Figure 8-3 shows an example, with the user first supplying his username/password, the switch asking the AAA server, and the server replying to the switch stating that the username/password is valid.

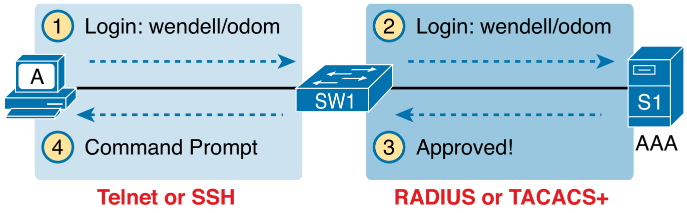


Figure 8-3 *Basic Authentication Process with an External AAA Server*

While the figure shows the general idea, note that the information flows with a couple of different protocols. On the left, the connection between the user and the switch or router uses Telnet or SSH. On the right, the switch and AAA server typically use either the RADIUS or TACACS+ protocol, both of which encrypt the passwords as they traverse the network.

Configuring Secure Shell (SSH)

To support SSH, Cisco switches require the base configuration used to support Telnet login with usernames, plus additional configuration. First, the switch already runs an SSH server by default, accepting incoming SSH connections from SSH clients. In addition, the switch needs a cryptography key, used to encrypt the data. The following list details the steps for a Cisco switch to support SSH using local usernames, with Step 3 listing the new commands specific to SSH support:

Step 1. Configure the vty lines to use usernames, with either locally configured usernames (using the **login local** command) or a AAA server.

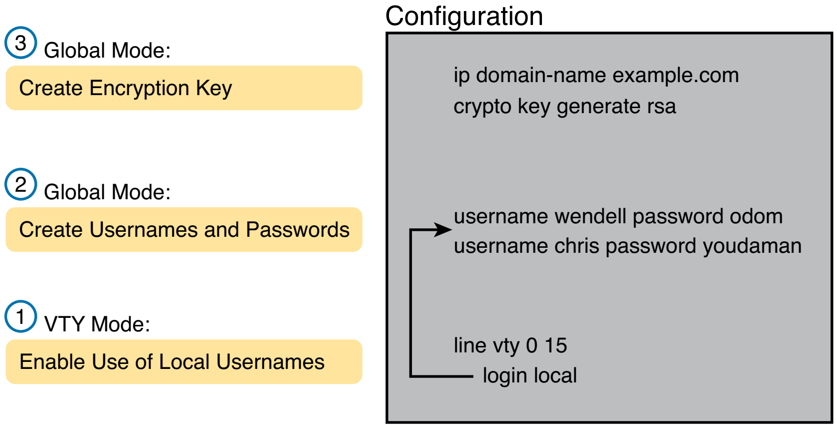
Step 2. If using locally defined usernames, add one or more **username** global configuration commands to configure username/password pairs.

Step 3. Configure the switch to generate a matched public and private key pair to use for encryption, using two commands:

- **A.** As a prerequisite for the next command, configure a DNS domain name with the **ip domain-name** *name* global configuration command.
- **B.** Create the encryption keys using the **crypto key generate rsa** global configuration command.

Step 4. (Optional) Enable SSH Version 2 using the **ip ssh version 2** global command for enhanced security.

Figure 8-4 shows the three steps, with examples of the required configuration commands. Note that this figure adds to the configuration shown in Figure 8-2, with only two more commands added to support SSH.



```
ip domain-name example.com
crypto key generate rsa

username wendell password odom
username chris password youdaman

line vty 0 15
 login local
```

Figure 8-4 *Configuring a Switch to Support Inbound SSH Login*

NOTE If you wonder why the figure shows the order from bottom to top, it is because the output of the **show running-config** command will list these configuration commands in the order shown in the figure.

Seeing the configuration happen in configuration mode, step by step, can be particularly helpful with SSH. Note in particular that the **crypto key** command actually prompts the user for more information and generates some messages while the key is being generated. Example 8-3 shows the commands in Figure 8-4 being configured, with the encryption key as the final step.

Example 8-3 *SSH Configuration Process*

```
Emma# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Emma(config)# line vty 0 15
! Step 1's key command happens next
Emma(config-line)# login local
Emma(config-line)# exit
!
! Step 2's command happens next
Emma(config)# username wendell password odom
Emma(config)# username chris password youdaman
!
! Step 3's two commands happen next
Emma(config)# ip domain-name example.com
Emma(config)# crypto key generate rsa
The name for the keys will be: Emma.example.com
Choose the size of the key modulus in the range of 360 to 2048 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 4 seconds)

Emma(config)# ip ssh version 2
Emma(config)# ^Z
Emma#
```

Two key commands give some information about the status of SSH on the switch. First, the **show ip ssh** command lists status information about the SSH server itself. The **show ssh** command then lists information about each SSH client currently connected into the switch. Example 8-4 shows samples of each, with user Wendell currently connected to the switch.

Example 8-4 *Displaying SSH Status*

```
Emma# show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
Minimum expected Diffie Hellman key size : 1024 bits
IOS Keys in SECSH format(ssh-rsa, base64 encoded):
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQC+/mp2iaeaGwjqkIgLNH+lN/04LTc2u6qHVHHv3hoq
/DDBd9vABNnJGsq8z0Hm9HcrSudC20N/cCuEb4x5+T9rvNkUeAqwEEoJALpdiWVOpBliomhPysvJi+m4
wI16AH31KI+GFCZv1AIjZSYHQEbvdCEqsYezAeKnPhvzTrUqaQ==

Emma# show ssh
Connection Version Mode Encryption  Hmac         State                 Username
0          2.0      IN  aes128-cbc  hmac-sha1    Session started       wendell
0          2.0      OUT aes128-cbc  hmac-sha1    Session started       wendell
%No SSHv1 server connections running.
```

Note that this example does use SSH Version 2 rather than Version 1. SSH v2 improves the underlying security algorithms over SSH v1 and adds some other small advantages, like banner support.

Finally, the switch supports both Telnet and SSH on the vty lines, but you can disable either or both for even tighter security. For example, your company might require that you avoid Telnet because of the security risk, so you need to disable Telnet support on the switch. Switches can control their support of Telnet and/or SSH on the vty lines using the **transport input {all | none | telnet | ssh}** vty subcommand, with the following options:

transport input all or **transport input telnet ssh:** Support both

transport input none: Support neither

transport input telnet: Support only Telnet

transport input ssh: Support only SSH

Encrypting and Hiding Passwords

Several of the configuration commands discussed so far in this chapter list passwords in clear text in the running config file (at least by default). In fact, of the commands discussed in this chapter so far, only the **enable secret** command automatically hides the password value. The other commands—the console and vty lines with the **password** command, plus the password in the **username password** command—store the password in clear text by default.

The next few sections discuss several options for hiding password values. Some tools use encryption, and some use a one-way hash algorithm. Regardless of the detail, the result is that the passwords cannot be seen by anyone who happens to see the output of the **show running-config** command.

Encrypting Passwords with the **service password** Command

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 some passwords using the **service password-encryption** global configuration command. This command affects how IOS stores passwords for the **password** command, in both console and vty modes, and the **username password** global command. The rules for the **service password-config** command are as follows:

- At the moment that the **service password-encryption** command is configured, IOS immediately encrypts all existing **password** commands (in console and vty modes) and **username password** (global command) passwords.
- While the **service password-encryption** command remains in the configuration, IOS encrypts these same passwords if their values are changed.
- At the moment the **no service password-encryption** command is used, disabling password encryption, IOS does nothing to the existing passwords, leaving them all as encrypted.
- From that point forward, while the **service password-encryption** command is no longer in the configuration, IOS stores any changed password values for these commands as clear text.

Example 8-5 shows an example of these details.

Example 8-5 *Encryption and the* service password-encryption *Command*

```
Switch3# show running-config | begin line vty
line vty 0 4
 password cisco
 login
line vty 5 15
 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
line vty 5 15
 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 | section vty
line vty 0 4
 password 7 070C285F4D06
 login
line vty 5 15
 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
line vty 5 15
 password 7 070C285F4D06
 login
end
```

NOTE The encryption type used by the **service password-encryption** command, as noted with the “7” in the **password** commands, is weak. You can search the Internet and find sites with tools to decrypt these passwords. In fact, you can take the encrypted password from this example, plug it into one of these sites, and it decrypts to “cisco.” So, the **service password-encryption** command will slow down the curious, but it will not stop a knowledgeable attacker.

Example 8-5 also shows several examples of the pipe function (|), available at the end of CLI **show** commands. The | at the end of a **show** command sends (pipes) the output of the command to another function, like the **begin** and **section** functions shown in Example 8-5. The **begin** function, as shown in the **show running-config | begin line vty** command in the example, takes the output from the command and starts listing the text beginning when the first occurrence of the listed text (“vty” in this case) shows up. The | **section vty** parameters, also seen in Example 8-5, display only the section of output about the vty lines.

Hiding the Enable Password

Switches can protect enable mode by requiring that the user supply an enable password after using the **enable** EXEC command. However, the configuration can be based on two different commands: the older **enable password** *password* global command and the newer (and preferred) **enable secret** *password* global command.

IOS allows you to configure neither, one or the other, or even both of these commands. Then the switch chooses what password to require of a user based on the following rules:

- **Both commands configured:** Use the **enable secret** *password* command.
- **Only one command configured:** Use the password in that one command.
- **Neither command configured (default):** Console users are allowed into enable mode without a password prompt, while others are rejected.

The newer **enable secret** command provides much better security compared to the older **enable password** command. The older **enable password** command stores the password as clear text, and the only option to encrypt it is the weak **service password-encryption** command. The newer **enable secret** command automatically encodes the password, using a different process than the **service password-encryption** command. This newer command applies a mathematical function to the password, called a Message Digest 5 (MD5) hash, storing the results of the formula in the **enable secret** command in the configuration file.

Example 8-6 shows the creation of the **enable secret** command, and describes how it hides the password text. The example first lists the **enable secret fred** command, as typed by the user. Later, the **show running-configuration** command shows that IOS changed the **enable secret** command, now listing encryption type 5 (meaning it is an MD5 hash). The gobbledygook long text string is the MD5 hash, preventing others from reading the password.

Example 8-6 *Encryption and the* **enable secret** *Command*

```
Switch3(config)# enable secret ?
  0      Specifies an UNENCRYPTED password will follow
  5      Specifies an ENCRYPTED secret will follow
  LINE   The UNENCRYPTED (cleartext) 'enable' secret
  level  Set exec level password
```

```
Switch3(config)# enable secret fred
Switch3(config)# ^Z
Switch3# show running-config | include enable secret

enable secret 5 $1$ZGMA$e8cmvkz4UjiJhVp7.maLE1

Switch3# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)# no enable secret
Switch3(config)# ^Z
```

The end of the example also shows an important point about deleting the enable secret password: After you are in enable mode, you can delete the enable secret password using the **no enable secret** command, without even having to enter the password value. You can also overwrite the old password by just repeating the command.

NOTE Example 8-6 shows another shortcut with working through long **show** command output, using the pipe to the **include** command. The **show running-config | include enable secret** command lists the output from **show running-config**, but only lines that include the case-sensitive text "enable secret."

Finally, note that Cisco has added another hash algorithm to the **enable secret** command for routers: SHA-256. This algorithm is stronger than MD5, with IOS listing this algorithm as encryption type 4. Over time, Cisco will likely add SHA-256 support to switches as well. Regardless, the effect of using SHA-256 or MD5 is the same: The user configures a command like **enable secret fred**, typing the clear-text password, and IOS stores the hash value as either MD5 (older IOS versions) or SHA-256 (newer IOS versions).

Hiding the Passwords for Local Usernames

Cisco added the **enable secret** command, and its better password protection, back in the 1990s. More recently, Cisco has added the **username** *user* **secret** *password* global command as an alternative to the **username** *user* **password** *password* command. Note that this command uses an SHA-256 (type 4) hash.

Today, the **username secret** command is preferred over the **username password** command, much like the **enable secret** command is preferred over the **enable password** command. However, note that a username can be configured with either the **username secret** command or the **username password** command, but not both.

Console and vty Settings

This section covers a few small configuration settings that affect the behavior of the CLI connection from the console and/or vty (Telnet and SSH).

Banners

Cisco switches can display a variety of banners depending on what a router or switch administrator is doing. A banner is simply some text that appears on the screen for the user. You can configure a router or switch to display multiple banners, some before login and some after. Table 8-1 lists the three most popular banners and their typical use.

Table 8-1 Banners and Their Use

Banner	Typical Use
Message of the Day (MOTD)	Shown before the login prompt. Used for temporary messages that can change from time to time, such as "Router1 down for maintenance at midnight."
Login	Shown before the login prompt but after the MOTD banner. Used for permanent messages such as "Unauthorized Access Prohibited."
Exec	Shown after the login prompt. Used to supply information that should be hidden from unauthorized users.

The **banner** global configuration command can be used to configure all three types of these banners. In each case, the type of banner is listed as the first parameter, with MOTD being the default option. The first nonblank character after the banner type is called a beginning delimiter character. The banner text can span several lines, with the CLI user pressing Enter at the end of each line. The CLI knows that the banner has been configured as soon as the user enters the same delimiter character again.

Example 8-7 shows the configuration process for all three types of banners from Table 8-1, followed by a sample user login session that shows the banners in use. The first banner in the example, the MOTD banner, omits the banner type in the **banner** command as a reminder that **motd** is the default banner type. The first two **banner** commands use a **#** as the delimiter character. The third **banner** command uses a Z as the delimiter, just to show that any character can be used. Also, the last **banner** command shows multiple lines of banner text.

Example 8-7 *Banner Configuration*

```
! Below, the three banners are created in configuration mode. Note that any
! delimiter can be used, as long as the character is not part of the message
! text.

SW1(config)# banner #
Enter TEXT message.  End with the character '#'.
(MOTD) Switch down for maintenance at 11PM Today #
SW1(config)# banner login #
Enter TEXT message.  End with the character '#'.
(Login) Unauthorized Access Prohibited!!!!
#
SW1(config)# banner exec Z
Enter TEXT message.  End with the character 'Z'.
(Exec) Company picnic at the park on Saturday
 Don’t tell outsiders!
Z
SW1(config)# end

! Below, the user of this router quits the console connection, and logs back in,
! seeing the motd and login banners, then the password prompt, and then the
! exec banner.
SW1#quit

SW1 con0 is now available

Press RETURN to get started.
```

```
(MOTD) Switch down for maintenance at 11PM Today
(Login) Unauthorized Access Prohibited!!!!

User Access Verification

Username: fred
Password:
(Exec) Company picnic at the park on Saturday
don't tell outsiders!
SW1>
```

History Buffer Commands

When you enter commands from the CLI, the last several commands are saved in the history buffer. As mentioned in Chapter 7, you can use the up-arrow key, or press Ctrl+P, to move back in the history buffer stack to retrieve a command you entered a few commands ago. This feature makes it very easy and fast to use a set of commands repeatedly. Table 8-2 lists some of the key commands related to the history buffer.

Key Topic

Table 8-2 Commands Related to the History Buffer

Command	Description
show history	Lists the commands currently held in the history buffer.
history size *x*	From console or vty line configuration mode, sets the default number of commands saved in the history buffer for the user(s) of the console or vty lines, respectively.
terminal history size *x*	From EXEC mode, this command allows a single user to set, just for this one login session, the size of his or her history buffer.

The **logging synchronous** and **exec-timeout** Commands

The next short section looks at a couple of ways to make using the console a little more user friendly, by asking the switch to not interrupt with log messages, and to control how long you can be connected to the console before getting forced out.

The console automatically receives copies of all unsolicited syslog messages on a switch. The idea is that if the switch needs to tell the network administrator some important and possibly urgent information, the administrator might be at the console and might notice the message.

The display of these messages at the console can be disabled and enabled with the **no logging console** and **logging console** global commands. For example, when working from the console, if you want to temporarily not be bothered by log messages, you can disable the display of these messages with the **no logging console** global configuration command, and then when finished, enable them again.

Unfortunately, IOS (by default) displays these syslog messages on the console's screen at any time—including right in the middle of a command you are entering, or in the middle of the output of a **show** command. Having a bunch of text show up unexpectedly can be a bit annoying.

IOS supplies a solution to this problem by telling the switch to display syslog messages only at more convenient times, such as at the end of output from a **show** command. To do so, just configure the **logging synchronous** console line subcommand.

Another way to improve the user experience at the console is to control timeouts from the console. By default, the switch automatically disconnects console and vty (Telnet and SSH) users after 5 minutes of inactivity. The **exec-timeout** *minutes seconds* line subcommand lets you set the length of that inactivity timer, with the special value of 0 minutes and 0 seconds meaning "never time out."

Example 8-8 shows the syntax for these two commands, both on the console line. Note that both can be applied to the vty lines as well, for the same reasons.

Example 8-8 *Defining Console Inactivity Timeouts and When to Display Log Messages*

```
line console 0
 login
 password cisco
 exec-timeout 0 0
 logging synchronous
```

NOTE This concludes the first half of this chapter. If you have not yet tried any commands on a router or switch, now would be a good time to pause from your reading and try some. If you have real gear, or the Pearson Simulator, do some labs about navigating the CLI, setting passwords, and other basic administration. If not, watch the videos from the DVD on CLI navigation and route configuration. Also, try a few labs from the ICND1 Simulator Lite on the DVD. Even if you do a lab on something you have not seen yet, you can get a little better idea about how to move around with the command-line interface.

LAN Switch Configuration and Operation

Cisco switches work very well when received from the factory, without any configuration added. Cisco switches leave the factory with default settings, with all interfaces enabled (a default configuration of **no shutdown**) and with autonegotiation enabled for ports that can use it (a default configuration of **duplex auto** and **speed auto**). All interfaces default to be part of VLAN 1 (**switchport access vlan 1**). All you have to do with a new Cisco switch is make all the physical connections—Ethernet cables and power cord—and the switch starts working.

In most enterprise networks, you will want the switch to operate with some different settings as compared with the factory defaults. The second half of this chapter discusses some of those settings, with Chapter 9 ("Implementing Ethernet Virtual LANs") discussing more. (Also note that the details in this section differ from the configuration on a router.) In particular, this section covers the following:

- IP for remote access
- Interface configuration (including speed and duplex)
- Port security
- Securing unused switch interfaces

Enabling IP for Remote Access

To allow Telnet or SSH access to the switch, and to allow other IP-based management protocols (for example, Simple Network Management Protocol) to function as intended, the switch needs an IP address. The IP address has nothing to do with how switches forward Ethernet frames; it simply exists to support overhead management traffic.

A switch's IP configuration works like a PC with a single Ethernet interface. For perspective, a PC has a CPU, with the operating system running on the CPU. It has an Ethernet network interface card (NIC). The OS configuration includes an IP address associated with the NIC, either configured or learned dynamically with DHCP. To support IP, the switch has the equivalent settings.

A switch uses concepts similar to a host, except that the switch can use a virtual NIC. Like a PC, a switch has a real CPU, running an OS (called IOS). The switch then uses a NIC-like concept called a *switched virtual interface (SVI)*, or more commonly, a *VLAN interface*, that acts like the switch's own NIC for connecting into a LAN to send IP packets. Like a host, the switch configuration assigns IP settings, like an IP address, to this VLAN interface, as seen in Figure 8-5.

Key Topic

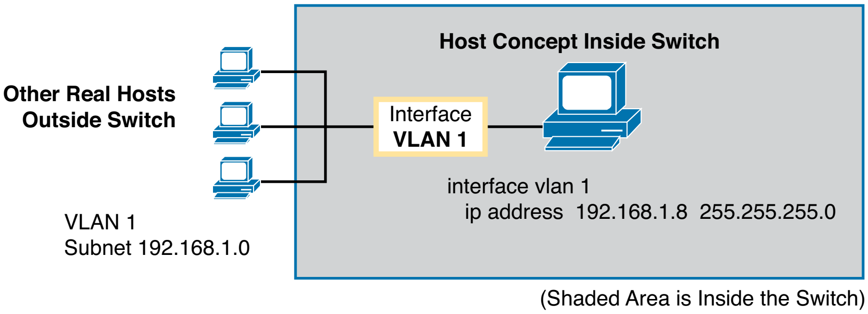

Host Concept Inside Switch
Other Real Hosts Outside Switch
Interface VLAN 1
interface vlan 1
ip address 192.168.1.8 255.255.255.0
VLAN 1
Subnet 192.168.1.0
(Shaded Area is Inside the Switch)

Figure 8-5 *Switch Virtual Interface (SVI) Concept Inside a Switch*

A typical Layer 2 Cisco LAN switch can use only one VLAN interface at a time, but the network engineer can choose which VLAN interface, putting the switch's management traffic into a particular VLAN. For example, Figure 8-6 shows a switch with some physical ports in two different VLANs (1 and 2). The network engineer needs to choose whether the switch IP address, used to access and manage the switch, should have an IP address in subnet 192.168.1.0 (in VLAN 1), or in subnet 192.168.2.0 (in VLAN 2).

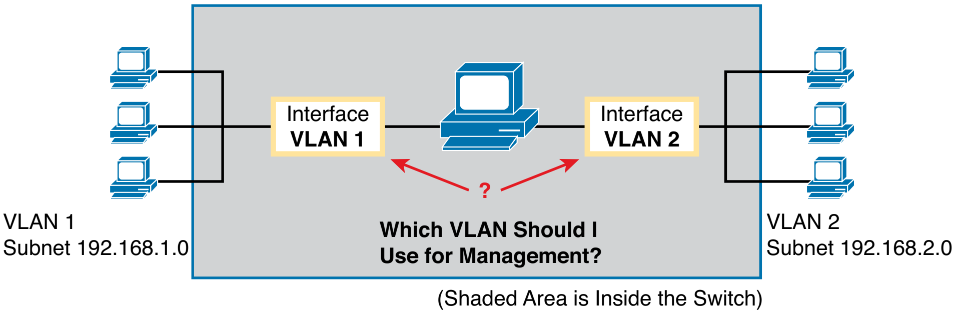


Figure 8-6 *Choosing One VLAN on Which to Configure a Switch IP Address*

NOTE Some Cisco switches, called *Layer 2 switches*, forward Ethernet frames as discussed in depth in Chapter 6, "Building Ethernet LANs with Switches." Other Cisco switches, called *multilayer switches* or *Layer 3 switches*, can also route IP packets using the Layer 3 logic normally used by routers. Layer 3 switches configure IP addresses on more than one VLAN interface at a time. This chapter assumes all switches are Layer 2 switches. Chapter 9 further defines the differences between these types of LAN switches.

Configuring IPv4 on a Switch

A switch configures its IPv4 address and mask on this special NIC-like *VLAN interface*. The following steps list the commands used to configure IPv4 on a switch, assuming that the IP address is configured to be in VLAN 1, with Example 8-9 that follows showing an example configuration.

Step 1. Enter VLAN 1 configuration mode using the **interface vlan 1** global configuration command.

Step 2. Assign an IP address and mask using the **ip address** *ip-address mask* interface subcommand.

Step 3. If not already enabled, enable the VLAN 1 interface using the **no shutdown** interface subcommand.

Step 4. Add the **ip default-gateway** *ip-address* global command to configure the default gateway.

Step 5. (Optional) Add the **ip name-server** *ip-address1 ip-address2 . . .* global command to configure the switch to use DNS to resolve names into their matching IP address.

Example 8-9 *Switch Static IP Address Configuration*

```
Emma# configure terminal
Emma(config)# interface vlan 1
Emma(config-if)# ip address 192.168.1.200 255.255.255.0
Emma(config-if)# no shutdown
00:25:07: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
00:25:08: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed
  state to up
Emma(config-if)# exit
Emma(config)# ip default-gateway 192.168.1.1
```

On a side note, this example shows a particularly important and common command: the **[no] shutdown** command. To administratively enable an interface on a switch, use the **no shutdown** interface subcommand; to disable an interface, use the **shutdown** interface subcommand. The messages shown in Example 8-9, immediately following the **no shutdown** command, are syslog messages generated by the switch stating that the switch did indeed enable the interface.

The switch can also use DHCP to dynamically learn its IPv4 settings. Basically, all you have to do is tell the switch to use DHCP on the interface, and enable the interface. Assuming that DHCP works in this network, the switch will learn all its settings. The following list details the steps, again assuming the use of interface VLAN 1, with Example 8-10 that follows showing an example.

Step 1. Enter VLAN 1 configuration mode using the **interface vlan 1** global configuration command, and enable the interface using the **no shutdown** command as necessary.

Step 2. Assign an IP address and mask using the **ip address dhcp** interface subcommand.

Example 8-10 *Switch Dynamic IP Address Configuration with DHCP*

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

Verifying IPv4 on a Switch

The switch IPv4 configuration can be checked in several places. First, you can always look at the current configuration using the **show running-config** command. Second, you can look at the IP address and mask information using the **show interface vlan** *x* command, which shows detailed status information about the VLAN interface in VLAN *x*. Finally, if using DHCP, use the **show dhcp lease** command to see the (temporarily) leased IP address and other parameters. (Note that the switch does not store the DHCP-learned IP configuration in the running-config file.) Example 8-11 shows sample output from these commands to match the configuration in Example 8-10.

Example 8-11 *Verifying DHCP-learned Information on a Switch*

```
Emma# show dhcp lease
Temp IP addr: 192.168.1.101  for peer on Interface: Vlan1
Temp  sub net mask: 255.255.255.0
   DHCP Lease server: 192.168.1.1, state: 3 Bound
   DHCP transaction id: 1966
   Lease: 86400 secs,  Renewal: 43200 secs,  Rebind: 75600 secs
Temp default-gateway addr: 192.168.1.1
   Next timer fires after: 11:59:45
   Retry count: 0   Client-ID: cisco-0019.e86a.6fc0-Vl1
   Hostname: Emma
Emma# show interfaces vlan 1
Vlan1 is up, line protocol is up
  Hardware is EtherSVI, address is 0019.e86a.6fc0 (bia 0019.e86a.6fc0)
  Internet address is 192.168.1.101/24
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
! lines omitted for brevity
Emma# show ip default-gateway
192.168.1.1
```

The output of the **show interfaces vlan 1** command lists two very important details related to switch IP addressing. First, this **show** command lists the interface status of the VLAN 1 interface—in this case, "up and up." If the VLAN 1 interface is not up, the switch cannot use its IP address to send and receive traffic. Notably, if you forget to issue the **no shutdown** command, the VLAN 1 interface remains in its default shutdown state and is listed as "administratively down" in the **show** command output.

Second, note that the output lists the interface's IP address on the third line. If you statically configure the IP address, as in Example 8-9, the IP address will always be listed. However, if you use DHCP, and DHCP fails, the **show interfaces vlan** *x* command will not list an IP address here. When DHCP works, you can see the IP address with this command, but it does not remind you whether the address is either statically configured or DHCP leased.

Configuring Switch Interfaces

IOS uses the term *interface* to refer to physical ports used to forward data to and from other devices. Each interface can be configured with several settings, each of which might differ from interface to interface.

IOS uses interface subcommands to configure these settings. For example, interfaces can be configured to use the **duplex** and **speed** interface subcommands to configure those settings statically, or an interface can use autonegotiation (the default). Example 8-12 shows how to configure duplex and speed, as well as the **description** command, which is simply a text description that can be configured by the administrator.

Example 8-12 *Interface Configuration Basics*

```
Emma# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Emma(config)# interface FastEthernet 0/1
Emma(config-if)# duplex full
Emma(config-if)# speed 100
Emma(config-if)# description Server1 connects here
Emma(config-if)# exit
Emma(config)# interface range FastEthernet 0/11 - 20
Emma(config-if-range)# description end-users connect_here
Emma(config-if-range)# ^Z
Emma#
Emma# show interfaces status
Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1     Server1 connects h notconnect   1            full    100 10/100BaseTX
Fa0/2                        notconnect   1            auto   auto 10/100BaseTX
Fa0/3                        notconnect   1            auto   auto 10/100BaseTX
Fa0/4                        connected    1          a-full  a-100 10/100BaseTX
Fa0/5                        notconnect   1            auto   auto 10/100BaseTX
Fa0/6                        connected    1          a-full  a-100 10/100BaseTX
Fa0/7                        notconnect   1            auto   auto 10/100BaseTX
Fa0/8                        notconnect   1            auto   auto 10/100BaseTX
Fa0/9                        notconnect   1            auto   auto 10/100BaseTX
Fa0/10                       notconnect   1            auto   auto 10/100BaseTX
Fa0/11    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/12    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/13    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/14    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/15    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/16    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/17    end-users connect  notconnect   1            auto   auto 10/100BaseTX
```

```
Fa0/18    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/19    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/20    end-users connect  notconnect   1            auto   auto 10/100BaseTX
Fa0/21                       notconnect   1            auto   auto 10/100BaseTX
Fa0/22                       notconnect   1            auto   auto 10/100BaseTX
Fa0/23                       notconnect   1            auto   auto 10/100BaseTX
Fa0/24                       notconnect   1            auto   auto 10/100BaseTX
Gi0/1                        notconnect   1            auto   auto 10/100/1000BaseTX
Gi0/2                        notconnect   1            auto   auto 10/100/1000BaseTX
```

You can see some of the details of interface configuration with both the **show running-config** command (not shown in the example) and the handy **show interfaces status** command. This command lists a single line for each interface, the first part of the interface description, and the speed and duplex settings. Several of the early entries in the output purposefully show some differences, as follows:

FastEthernet 0/1 (Fa0/1): This output lists the configured speed of 100 and duplex full; however, it lists a status of notconnect. The notconnect status means that the physical link is not currently working, including reasons like no cable being connected, the other device being powered off, or the other device putting the port in a shutdown state. In this case, no cable had been installed when the output was gathered.

FastEthernet 0/2 (Fa0/2): This port also has no cable installed yet, but it uses all default configuration. So, the highlighted output shows this interface with the default setting of auto (meaning autonegotiate).

FastEthernet 0/4 (Fa0/4): Like Fa0/2, this port has all default configuration, but was cabled to another device that is up, causing the status to be listed as "connect." This device also completed the autonegotiation process, so the output lists the resulting speed and duplex (**a-full** and **a-100**), in which the **a-** refers to the fact that these values were autonegotiated.

Also, note that for the sake of efficiency, you can configure a command on a range of interfaces at the same time using the **interface range** command. In the example, the **interface range FastEthernet 0/11 - 20** command tells IOS that the next subcommand(s) apply to interfaces Fa0/11 through Fa0/20.

NOTE Configuring both the speed and duplex on a Cisco switch interface disables autonegotiation.

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 attacks in which the attacker connects a laptop to some unused switch port. When that inappropriate device attempts to send frames to the switch interface, the switch can take different actions, ranging from simply issuing informational messages to effectively shutting down the interface.

Port security identifies devices based on the source MAC address of Ethernet frames the devices send. For example, in Figure 8-7, PC1 sends a frame, with PC1's MAC address as the source address. SW1's F0/1 interface can be configured with port security, and if so, SW1 would think about PC1's MAC address and whether PC1 was allowed to send frames into port F0/1.

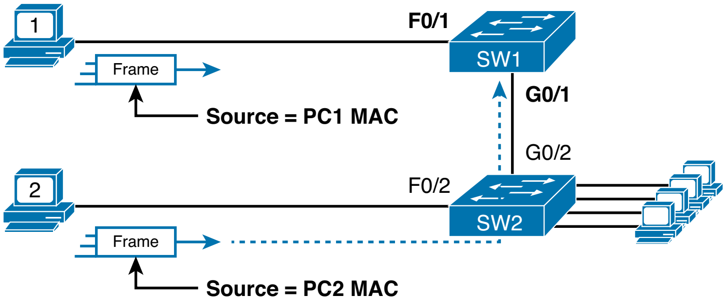


Figure 8-7 *Source MAC Addresses in Frames as They Enter a Switch*

Port security also has no restrictions on whether the frame came from a local device or it was forwarded through other switches. For example, switch SW1 could use port security on its G0/1 interface, checking the source MAC address of the frame from PC2, when forwarded up to SW1 from SW2.

Port security has several flexible options, but all operate with the same core concepts. First, switches enable port security per port, with different settings available per port. Each port has a maximum number of allowed MAC addresses, meaning that for all frames entering that port, only that number of *different* source MAC addresses can be used in different incoming frames before port security thinks a violation has occurred. When a frame with a new source MAC address arrives, pushing the number of MAC addresses past the allowed maximum, a port security violation occurs. At that point, the switch takes action—by default, discarding all future incoming traffic on that port.

The following list summarizes these ideas common to all variations of port security:

Key Topic

- Define a maximum number of source MAC addresses allowed for all frames coming in the interface.
- Watch all incoming frames, and keep a list of all source MAC addresses, plus a counter of the number of different source MAC addresses.
- When adding a new source MAC address to the list, if the number of MAC addresses pushes past the configured maximum, a port security violation has occurred. The switch takes action (the default action is to shutdown the interface).

While those rules define the basics, port security allows other options as well, including letting you configure the specific MAC address(es) allowed to send frames in an interface. For example, in Figure 8-7, switch SW1 connects through interface F0/1 to PC1, so the port security configuration could list PC1's MAC address as the specific allowed MAC address. But predefining MAC addresses for port security is optional: You can predefine all MAC addresses, none, or a subset of the MAC addresses.

You might like the idea of predefining the MAC addresses for port security, but finding the MAC address of each device can be a bother. Port security provides an easy way to discover the MAC addresses used off each port using a feature called *sticky secure MAC addresses*. With this feature, port security learns the MAC addresses off each port and stores those in the port security configuration (in the running-config file). This feature helps reduce the big effort of finding out the MAC address of each device.

As you can see, port security has a lot of detailed options. The next few sections walk you through these options to pull the ideas together.

Configuring Port Security

Port security configuration involves several steps. First, you need to disable the negotiation of a feature that is not discussed until Chapter 9: whether the port is an access or trunk port. For now, accept that port security requires a port to be configured to either be an access port or a trunking port. The rest of the commands enable port security, set the maximum allowed MAC addresses per port, and configure the actual MAC addresses, as detailed in this list:

Step 1. Make the switch interface either a static access or trunk interface, using the **switchport mode access** or the **switchport mode trunk** interface subcommands, respectively.

Step 2. Enable port security using the **switchport port-security** interface subcommand.

Step 3. (Optional) Override the default maximum number of allowed MAC addresses associated with the interface (1) by using the **switchport port-security maximum** *number* interface subcommand.

Step 4. (Optional) Override the default action to take upon a security violation (shutdown) using the **switchport port-security violation** {**protect** | **restrict** | **shutdown**} interface subcommand.

Step 5. (Optional) Predefine any allowed source MAC address(es) for 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 6. (Optional) Tell the switch to "sticky learn" dynamically learned MAC addresses with the **switchport port-security mac-address sticky** interface subcommand.

Figure 8-8 and Example 8-13 show four examples of port security, each with different details just to show the different options.

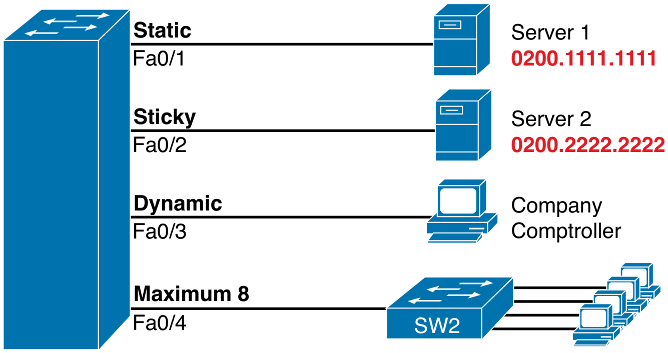


Figure 8-8 *Port Security Configuration Example*

Example 8-13 *Variations on Port Security Configuration*

```
SW1# show running-config
(Lines omitted for brevity)

interface FastEthernet0/1
 switchport mode access
 switchport port-security
 switchport port-security mac-address 0200.1111.1111
!
```

```
interface FastEthernet0/2
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
!
interface FastEthernet0/3
 switchport mode access
 switchport port-security
!
interface FastEthernet0/4
 switchport mode access
 switchport port-security
 switchport port-security maximum 8
```

First, scan the configuration for all four interfaces in Example 8-13, focusing on the first two interface subcommands. Note that all four interfaces in the example use the same first two interface subcommands, matching the first two configuration steps noted before Figure 8-8. The **switchport port-security** command enables port security, with all defaults, with the **switchport mode access** command meeting the requirement to configure the port as either an access or trunk port.

Next, scan all four interfaces again, and note that the configuration differs on each interface after those first two interface subcommands. Each interface simply shows a different example for perspective.

The first interface, FastEthernet 0/1, adds one optional port security subcommand: **switchport port-security mac-address 0200.1111.1111**, which defines a specific source MAC address. With the default maximum source address setting of 1, only frames with source MAC 0200.1111.1111 will be allowed in this port. When a frame with a source other than 0200.1111.1111 enters F0/1, the switch will take the default violation action and disable the interface.

As a second example, FastEthernet 0/2 uses the same logic as FastEthernet 0/1, except that it uses the sticky learning feature instead of predefining a MAC address with the **switchport port-security mac-address sticky** command. The end of upcoming Example 8-14 shows the running config file that lists the sticky-learned MAC address in this case.

NOTE Port security does not save the configuration of the sticky addresses, so use the **copy running-config startup-config** command if desired.

The other two interfaces do not predefine MAC addresses, nor do they sticky-learn the MAC addresses. The only difference between these two interfaces' port security configuration is that FastEthernet 0/4 supports eight MAC addresses, because it connects to another switch and should receive frames with multiple source MAC addresses. Interface F0/3 uses the default maximum of one MAC address.

Verifying Port Security

Example 8-14 lists the output of two examples of the **show port-security interface** command. This command lists the configuration settings for port security on an interface, plus it lists several important facts about the current operation of port security, including information about any security violations. The two commands in the example show interfaces F0/1 and F0/2, based on Example 8-13's configuration.

Example 8-14 *Using Port Security to Define Correct MAC Addresses of Particular Interfaces*

```
SW1# show port-security interface fastEthernet 0/1
Port Security              : Enabled
Port Status                : Secure-shutdown
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0013.197b.5004:1
Security Violation Count   : 1

SW1# show port-security interface fastEthernet 0/2
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 1
Last Source Address:Vlan   : 0200.2222.2222:1
Security Violation Count   : 0

SW1# 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
```

The first two commands in Example 8-14 confirm that a security violation has occurred on FastEthernet 0/1, but no violations have occurred on FastEthernet 0/2. The **show port-security interface fastethernet 0/1** command shows that the interface is in a *secure-shutdown* state, which means that the interface has been disabled because of port security. In this case, another device connected to port F0/1, sending a frame with a source MAC address other than 0200.1111.1111, is causing a violation. However, port Fa0/2, which used sticky learning, simply learned the MAC address used by Server 2.

The bottom of Example 8-14, as compared to the configuration in Example 8-13, shows the changes in the running-config because of sticky learning, with the **switchport port-security mac-address sticky 0200.2222.2222** interface subcommand.

Port Security Actions

Finally, the switch can be configured to use one of three actions when a violation occurs. All three options cause the switch to discard the offending frame, but some of the options make the switch take additional actions. The actions include the sending of syslog messages to the console, sending SNMP trap messages to the network management station, and disabling the interface. Table 8-3 lists the options of the **switchport port-security violation {protect | restrict | shutdown}** command and their meanings.

Table 8-3 Actions When Port Security Violation Occurs

Option on the switchport port-security violation Command	Protect	Restrict	Shutdown*
Discards offending traffic	Yes	Yes	Yes
Sends log and SNMP messages	No	Yes	Yes
Disables the interface, discarding all traffic	No	No	Yes

***shutdown** is the default setting.

Note that the shutdown option does not actually add the **shutdown** subcommand to the interface configuration. Instead, IOS puts the interface in an *error disabled* (err-disabled) state, which makes the switch stop all inbound and outbound frames. To recover from this state, someone must manually disable the interface with the **shutdown** interface command and then enable the interface with the **no shutdown** command.

Securing Unused Switch Interfaces

The default settings on Cisco switches work great if you want to buy a switch, unbox it, plug it in, and have it immediately work without any other effort. Those same defaults have an unfortunate side effect for worse security. With all default configuration, unused interfaces might be used by an attacker to gain access to the LAN. So, Cisco makes some general recommendations to override the default interface settings to make the unused ports more secure, as follows:

- Administratively disable the interface using the **shutdown** interface subcommand.
- Prevent VLAN trunking by making the port a nontrunking interface using the **switchport mode access** interface subcommand.
- Assign the port to an unused VLAN using the **switchport access vlan** *number* interface subcommand.
- Set the native VLAN to not be VLAN 1, but to instead be an unused VLAN, using the **switchport trunk native vlan** *vlan-id* interface subcommand. (The native VLAN is discussed in Chapter 9.)

Frankly, if you just shutdown the interface, the security exposure goes away, but the other tasks prevent any immediate problems if someone else comes around and enables the interface by configuring a **no shutdown** command.

Review Activities

Chapter Summary

- The first step in securing a switch is to secure access to the CLI. Securing the CLI includes protecting access to enable mode, because from enable mode an attacker could reload the switch or change the configuration.
- Cisco switches protect enable mode for any user with the *enable password*. The user, in user mode, types the **enable** EXEC command and is prompted for this enable password. If the user types the correct password, IOS moves the user to enable mode.
- The console and vty password configuration uses the same two subcommands in console and vty line configuration modes, respectively. The **login** command tells IOS to use simple password security, and the **password** *password_value* command defines the password. IOS protects enable mode using the enable secret password, configured using the global command **enable secret** *password_value*.
- The migration from using the password-only login method to using locally configured usernames and passwords requires only some small configuration changes. The switch needs one or more **username** *name* **password** *password* global configuration commands to define the usernames and passwords.
- Cisco switches and routers support an alternative way to keep track of valid usernames and passwords by using an external AAA server. When using a AAA server for authentication, the switch (or router) simply sends a message to the AAA server asking whether the username and password are allowed, and the AAA server replies.
- To support SSH, Cisco switches require the base configuration used to support Telnet login with usernames, plus additional configuration. First, the switch already runs an SSH server by default, accepting incoming SSH connections from both SSH version 1 and version 2 clients. In addition, the switch needs a cryptography key, used to encrypt the data.
- 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 some passwords using the **service password-encryption** global configuration command.
- The **banner** global configuration command can be used to configure all three types of banners:
  - **The message of the day (MOTD):** Shown before the login prompt. For temporary messages that might change from time to time, such as "Router1 down for maintenance at midnight."
  - **The login banner:** Shown before the login prompt but after the MOTD banner. For permanent messages, such as "Unauthorized Access Prohibited."
  - **The Exec banner:** Shown after the login prompt. Used to supply information that should be hidden from unauthorized users.
- A switch configures its IPv4 address and mask on this special NIC-like *VLAN interface*. The following steps list the commands used to configure IPv4 on a switch, assuming the IP address is configured to be in VLAN 1:

  Step 1. Enter VLAN 1 configuration mode using the **interface vlan 1** global configuration command.

  Step 2. Assign an IP address and mask using the **ip address** *ip-address mask* interface subcommand.

  Step 3. If not already enabled, enable the VLAN 1 interface using the **no shutdown** interface subcommand.

Step 4. Add the **ip default-gateway** *ip-address* global command to configure the default gateway.

Step 5. (Optional) Add the **ip name-server** *ip-address1 ip-address2…* global command to configure the switch to use DNS to resolve names into their matching IP addresses.

- To administratively enable an interface on a switch, use the **no shutdown** interface subcommand. To disable an interface, use the **shutdown** interface subcommand.
- If the network engineer knows what devices should be cabled and connected to particular interfaces on a switch, the engineer can use *port security* to restrict that interface so that only the expected devices can use it. This reduces exposure to attacks in which the attacker connects a laptop to some unused switch port. When that inappropriate device attempts to send frames to the switch interface, the switch can take different actions, ranging from simply issuing informational messages to effectively shutting down the interface.

Review Questions

Answer these review questions. You can find the answers at the bottom of the last page of the chapter. For thorough explanations, see DVD Appendix C, "Answers to Review Questions."

1. Imagine that you have configured the **enable secret** command, followed by the **enable password** command, from the console. You log out of the switch and log back in at the console. Which command defines the password that you had to enter to access privileged mode?

A. **enable password**

B. **enable secret**

C. Neither

D. The **password** command, if it's configured

2. An engineer had formerly configured a Cisco 2960 switch to allow Telnet access so that the switch expected a password of **mypassword** from the Telnet user. The engineer then changed the configuration to support Secure Shell. Which of the following commands could have been part of the new configuration? (Choose two answers.)

A. A **username** *name* **password** *password* vty mode subcommand

B. A **username** *name* **password** *password* global configuration command

C. A **login local** vty mode subcommand

D. A **transport input ssh** global configuration command

3. The following command was copied and pasted into configuration mode when a user was telnetted into a Cisco switch:

```
banner login this is the login banner
```

Which of the following is true about what occurs the next time a user logs in from the console?

A. No banner text is displayed.

B. The banner text "his is" is displayed.

C. The banner text "this is the login banner" is displayed.

D. The banner text "Login banner configured, no text defined" is displayed.

4. Which of the following is required when configuring port security with sticky learning?

A. Setting the maximum number of allowed MAC addresses on the interface with the **switchport port-security maximum** interface subcommand

B. Enabling port security with the **switchport port-security** interface subcommand

C. Defining the specific allowed MAC addresses using the **switchport port-security mac-address** interface subcommand

D. All the other answers list required commands

5. An engineer's desktop PC connects to a switch at the main site. A router at the main site connects to each branch office through a serial link, with one small router and switch at each branch. Which of the following commands must be configured on the branch office switches, in the listed configuration mode, to allow the engineer to telnet to the branch office switches? (Choose three answers.)

A. The **ip address** command in VLAN configuration mode

B. The **ip address** command in global configuration mode

C. The **ip default-gateway** command in VLAN configuration mode

D. The **ip default-gateway** command in global configuration mode

E. The **password** command in console line configuration mode

F. The **password** command in vty line configuration mode

6. Which of the following describes a way to disable IEEE standard autonegotiation on a 10/100 port on a Cisco switch?

A. Configure the **negotiate disable** interface subcommand

B. Configure the **no negotiate** interface subcommand

C. Configure the **speed 100** interface subcommand

D. Configure the **duplex half** interface subcommand

E. Configure the **duplex full** interface subcommand

F. Configure the **speed 100** and **duplex full** interface subcommands

7. In which of the following modes of the CLI could you configure the duplex setting for interface Fast Ethernet 0/5?

A. User mode

B. Enable mode

C. Global configuration mode

D. VLAN mode

E. Interface configuration mode