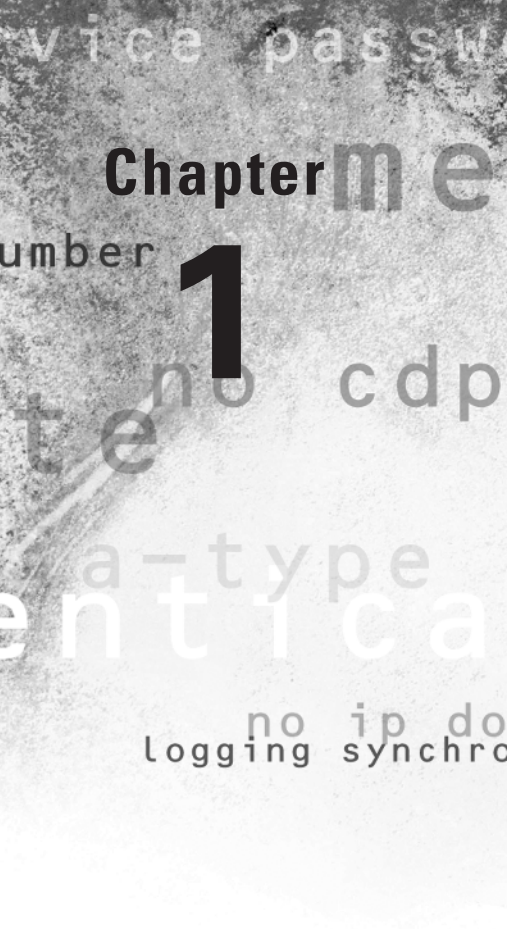


Basic IOS Commands

This book starts by introducing you to the Cisco Internetwork Operating System (IOS). The IOS is what runs Cisco routers as well as some Cisco switches, and it's what allows you to configure the devices. You use the command-line interface (CLI) to configure a router, and that is what I'll show you in this chapter.

The Cisco router IOS software is responsible for the following important tasks:

- Carrying network protocols and functions
- Connecting high-speed traffic between devices
- Adding security to control access and stop unauthorized network use
- Providing scalability for ease of network growth and redundancy
- Supplying network reliability for connecting to network resources

You can access the Cisco IOS through the console port of a router, from a modem into the auxiliary (or aux) port, or even through Telnet and Secure Shell (SSH). Access to the IOS command line is called an *exec session*.

Once you have attached your console cable (this is a rolled cable, sometimes referred to as a *rollover cable*) to the router and have started your terminal software, you will be ready to power on the router. Assuming that this is a new router, it will have no configuration and thus will need to have, well, everything set up. In this chapter, first I'll cover the power-on process of the router, and then I'll introduce the setup script.

For up-to-the minute updates for this chapter, please see www.lammle.com

Booting the Router

The following messages appear when you first boot or reload a router:

```
System Bootstrap, Version 12.4(13r)T, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 2006 by cisco Systems, Inc.
Initializing memory for ECC
c2811 platform with 262144 Kbytes of main memory
Main memory is configured to 64 bit mode with ECC enabled
Upgrade ROMMON initialized
```

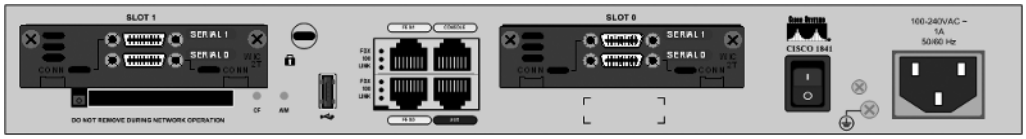
```
program load complete, entry point: 0x8000f000, size: 0xcb80
program load complete, entry point: 0x8000f000, size: 0xcb80
program load complete, entry point: 0x8000f000, size: 0x14b45f8
Self decompressing the image :
   ######################################################################
   ######################################### [OK]
 [some output cut]

Cisco IOS Software, 2800 Software (C2800NM-ADVSECURITYK9-M), Version
   12.4(12), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by Cisco Systems, Inc.
Compiled Fri 17-Nov-06 12:02 by prod_rel_team
Image text-base: 0x40093160, data-base: 0x41AA0000

[some output cut]

Cisco 2811 (revision 49.46) with 249856K/12288K bytes of memory.
Processor board ID FTX1049A1AB
2 FastEthernet interfaces
4 Serial(sync/async) interfaces
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity enabled.
239K bytes of non-volatile configuration memory.
62720K bytes of ATA CompactFlash (Read/Write)
```

Notice the following in the previous messages:

- The type of router (2811) and the amount of memory (262,144KB)
- The version of software the router is running (12.4, 13)
- The interfaces on the router (two Fast Ethernet and four serial)

Figure 1.1 shows a picture of an 1841 router, which is what is called an *integrated services router* (ISR), just like the 2811 router output shown earlier.

FIGURE 1.1 A Cisco 1841 router



An 1841 router holds most of the same interfaces as a 2800 router, but it's smaller and less expensive.

Setup Mode

If the router has no initial configuration, you will be prompted to use setup mode to establish an initial configuration. You can also enter setup mode at any time from the command line by typing the command **setup** from something called *privileged mode*. Setup mode covers only some global commands and is generally just not helpful. Here is an example:

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

At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.

Basic management setup configures only enough connectivity
for management of the system, extended setup will ask you
to configure each interface on the system

Would you like to enter basic management setup? [yes/no]: y
Configuring global parameters:

  Enter host name [Router]:Ctrl+C
Configuration aborted, no changes made.
```

You can exit setup mode at any time by pressing Ctrl+C.

Router Configuration Modes

One key to navigating the CLI is to always be aware of which router configuration mode you are currently in (see Table 1.1). You can tell which configuration mode you are in by watching the CLI prompt.

TABLE 1.1 Router Configuration Modes

Mode	Definition	Example
User EXEC mode	Limited to basic monitoring commands	`Router>`
Privileged EXEC mode	Provides access to all other router commands	`Router#`

TABLE 1.1 Router Configuration Modes *(continued)*

Mode	Definition	Example
Global configuration mode	Commands that affect the entire system	`Router(config)#`
Specific configuration modes	Commands that affect interfaces, sub-interfaces, routing processes, or lines only	`Router(config-subif)#`

Once you understand the different modes, you will need to be able to move from one mode to another within the CLI. The commands in Table 1.2 allow you to navigate between the assorted CLI modes.

TABLE 1.2 Moving Between Modes

Command	Meaning
`Router>`**`enable`**	Changes from user EXEC to privileged EXEC mode
`Router#`**`disable`**	Changes to user EXEC from privileged EXEC mode
`Router#`**`config term`**	Changes to global configuration mode from privileged mode
`Router(config)#`**`exit`**	Exits from any configuration mode to privileged mode (Ctrl+Z is also commonly used)
`Router(config)#`**`interface <int>`**	Enters interface configuration mode from global configuration mode
`Router(config)#`**`interface <subint>`**	Enters subinterface configuration mode from global configuration mode
`Router(config)#`**`line <line>`**	Enters line configuration mode from global configuration mode
`Router(config)#`**`router eigrp 1`**	Enters router configuration mode from global configuration mode

Editing and Help Features

One difference between a good and a great CLI engineer is the ability to quickly edit the line being entered into the router. Great CLI engineers can quickly recall previously entered commands and modify them, which is often much quicker than reentering the entire command. Table 1.3 shows some of the editing commands most commonly used.

TABLE 1.3 CLI Editing Commands

Command	Meaning
Ctrl+P or Up arrow	Shows last command entered
Ctrl+N or Down arrow	Shows previous commands entered
`show history`	Shows last 10 commands entered by default
`show terminal`	Shows terminal configurations and history buffer size
`terminal history size`	Changes buffer size (the maximum is 256)
Ctrl+A	Moves your cursor to the beginning of the line
Ctrl+E	Moves your cursor to the end of the line
Esc+B	Moves back one word
Ctrl+B	Moves back one character
Ctrl+F	Moves forward one character
Esc+F	Moves forward one word
Ctrl+D	Deletes a single character
Backspace	Deletes a single character
Ctrl+R	Redisplays a line
Ctrl+U	Erases a line
Ctrl+W	Erases a word
Ctrl+Z	Ends configuration mode and returns to EXEC mode
Tab	Finishes typing a command for you

The CLI also provides extensive online help. Any great CLI engineer will have an excessively worn question-mark key on the keyboard! Table 1.4 shows some examples of using the online help.

TABLE 1.4 Online Help Commands

Command	Meaning
`Router#?`	Shows all available commands
`Router#c?`	Shows all available commands beginning with the letter *c*
`Router#clock ?`	Shows all available options for the `clock` command

Using the Question Mark

The only command is the question mark; however, it does make a difference where you use it. When entering complex IOS commands, it is common to use the question mark repeatedly while entering the command, as in the following example:

```
Router#clock ?
  read-calendar    Read the hardware calendar into the clock
  set              Set the time and date
  update-calendar  Update the hardware calendar from the clock
Router#clock set ?
  hh:mm:ss  Current Time
Router#clock set 11:15:11 ?
  <1-31>  Day of the month
  MONTH   Month of the year
Router#clock set 11:15:11 25 aug ?
  <1993-2035>  Year
Router#clock set 11:15:11 25 aug 2007 ?
  <cr>
Router#clock set 11:15:11 25 aug 2007
*Aug 25 11:15:11.000: %SYS-6-CLOCKUPDATE: System clock has been updated from
18:52:53 UTC Wed Feb 28 2007 to 11:15:11 UTC Sat Aug 25 2007, configured from
console by cisco on console.
```

Using the Pipe

The pipe (|) allows you to wade through all the configurations or other long outputs and get straight to your goods fast. Table 1.5 shows the pipe commands.

TABLE 1.5 Pipe Commands

Command	Meaning
`Router#sh running-config | ?`	Shows the options for the pipe command. These include the ability to begin, include, exclude, and so on.
`Router#sh run | begin interface`	Shows the running configuration, beginning at the interface configurations.
`Router#sh ip route | include 192.168.3.32`	Shows all entries in the IP routing table that include the IP address 192.168.3.32.

Here's an example of using the pipe command to view just interface information on a router:

```
Router#sh run | ?
  append    Append redirected output to URL (URLs supporting append operation
            only)
  begin     Begin with the line that matches
  exclude   Exclude lines that match
  include   Include lines that match
  redirect  Redirect output to URL
  section   Filter a section of output
  tee       Copy output to URL
!
Router#sh run | begin interface
interface FastEthernet0/0
 description Sales VLAN
 ip address 10.10.10.1 255.255.255.248
 duplex auto
 speed auto
!
```

Configuring a Router

In the following sections, I'll introduce the commands used to do basic router configuration. You'll use these commands (or should use them) on every router you configure.

Hostnames

You can set the identity of the router with the `hostname` command. This is only locally significant, which means it has no bearing on how the router performs name lookups or how the router works on the internetwork. Table 1.6 shows the command for setting a router's hostname.

TABLE 1.6 Setting a Router Hostname

Command	Meaning
`Router(config)#hostname` *name*	Sets the hostname of this router

Here's an example of setting a hostname on a router:

```
Router#config t
Router(config)#hostname Todd
Todd(config)#
```

Banners

You can create a banner to give anyone who shows up on the router exactly the information you want them to have. Make sure you're familiar with these four available banner types: exec process creation banner, incoming terminal line banner, login banner, and message of the day banner (all illustrated in Table 1.7).

TABLE 1.7 Setting a Banner

Command	Meaning
`Router(config)#banner motd #`	Enters a banner MOTD message and ends the message with the # character
`Router(config)#banner exec #`	Enters a banner exec message and ends the message with the # character
`Router(config)#banner incoming #`	Enters a banner incoming message and ends the message with the # character
`Router(config)#banner login #`	Enters a banner login message and ends the message with the # character

The following describes the various banners:

MOTD banner The MOTD banner will be displayed whenever anyone attaches to the router, regardless of how they access the router.

Exec banner You can configure a line activation (exec) banner to be displayed when an EXEC process (such as a line activation or incoming connection to a VTY line) is created. By simply starting a user exec session through a console port, you'll activate the exec banner.

Incoming banner You can configure a banner to be displayed on terminals connected to reverse Telnet lines. This banner is useful for providing instructions to users who use reverse Telnet.

Login banner You can configure a login banner to be displayed on all connected terminals. This banner is displayed after the MOTD banner but before the login prompts. The login banner can't be disabled on a per-line basis, so to globally disable it, you have to delete it with the `no banner login` command.

Passwords

You can use five passwords to secure your Cisco routers: console, auxiliary, Telnet (VTY), enable password, and enable secret. However, other commands are necessary to complete the password configurations on a router or switch, as shown in Table 1.8.

TABLE 1.8 Setting Passwords

Command	Meaning
`Todd(config)#enable password todd`	Sets the enable password to Todd.
`Todd(config)#enable secret todd`	Sets the enable secret password to Todd. Supersedes the enable password.
`Todd(config)#line` *line*	Changes to line mode to configure the console, aux, and VTY (Telnet).
`Todd(config-line)#password password`	The line password for aux, console, and VTY (Telnet) are all set in line configuration mode.
`Todd(config-line)#login`	When a line is configured to use a password, the login command must be set to prompt for login.
`Todd(config)#service password-encryption`	Encrypts the passwords in the clear-text configuration file (both running-config and startup-config).

Setting Passwords

Here's an example of setting all your passwords and then encrypting them in the plain configuration file:

```
Todd#config t
Todd(config)#line aux ?
  <0-0>  First Line number
Todd(config)#line aux 0
Todd(config-line)#password aux
Todd(config-line)#login
Todd(config-line)#line con 0
Todd(config-line)#password console
Todd(config-line)#login
Todd(config-line)#exit
Todd(config)#line vty 0 ?
  <1-1180>  Last Line number
  <cr>
Todd(config)#line vty 0 1180
Todd(config-line)#password telnet
Todd(config-line)#login
Todd(config)#enable password todd
Todd(config)#enable secret globalnet
Todd(config)#service password-encryption
Todd(config)#do show run
Building configuration...
[outut cut]
!
enable secret 5 $1$S0a2$rLcwXpbme3zIrl2.TS1kX.
enable password 7 010709005F
!
[output cut]
!
line con 0
 exec-timeout 1 40
 password 7 050809013243420C
 logging synchronous
 login
line aux 0
 password 7 03054E13
 login
line vty 0 4
 password 7 105A0C150B1206
```

```
 login
line vty 5 1180
 password 7 0010160A0A5E1F
 login
!
 [output cut]
Todd(config)#no service password-encryption
```

Some other console helpful commands include the following. This sets the console timeout in second and minutes from 0-35791:

```
Todd(config-line)#exec-timeout ?
```

This redisplays interrupted console input:

```
Todd(config-line)#logging synchronous
```

Here's an example of setting the exec-timeout and logging synchronous commands:

```
Todd)#config t
Todd(config)#line con 0
Todd(config-line)#exec-timeout ?
  <0-35791>  Timeout in minutes
Todd(config-line)#exec-timeout 0 ?
  <0-2147483>  Timeout in seconds
  <cr>
Todd(config-line)#exec-timeout 0 100
Todd(config-line)#logging synchronous
```

Setting Up Secure Shell (SSH)

Instead of Telnet, you can use *Secure Shell*, which creates a more secure session than the Telnet application that uses an unencrypted data stream. SSH uses encrypted keys to send data so that your username and password are not sent in the clear. Table 1.9 lists the commands.

TABLE 1.9 SSH Commands

Command	Meaning
`ip domain-name Lammle.com`	Sets your domain name. You must set this.
`crypto key generate rsa general-keys modulus`	Sets the size of the key up to 2048.
`ip ssh time-out`	Sets the idle timeout up to 120 seconds.

TABLE 1.9 SSH Commands *(continued)*

Command	Meaning
`ip ssh authentication-retries`	Sets the max failed attempts up to 120.
`line vty first-line last_line`	Chooses your VTY lines to configure.
`transport input ssh telnet`	Tells the router to use SSH and then Telnet. You do not need the `telnet` command at the end of the line, but if you don't use it, only SSH will work on the router.

Here's an example of how you set up SSH on a router:

1. Set your hostname:

```
Router(config)#hostname Todd
```

2. Set the domain name (both the hostname and domain name are required for the encryption keys to be generated):

```
Todd(config)#ip domain-name Lammle.com
```

3. Generate the encryption keys for securing the session:

```
Todd(config)#crypto key generate rsa general-keys modulus ?
  <360-2048>  size of the key modulus [360-2048]
Todd(config)#crypto key generate rsa general-keys modulus 1024
The name for the keys will be: Todd.Lammle.com
% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
*June 24 19:25:30.035: %SSH-5-ENABLED: SSH 1.99 has been enabled
```

4. Set the maximum idle timer for an SSH session:

```
Todd(config)#ip ssh time-out ?
  <1-120>  SSH time-out interval (secs)
Todd(config)#ip ssh time-out 60
```

5. Set the maximum failed attempts for an SSH connection:

```
Todd(config)#ip ssh authentication-retries ?
  <0-5>  Number of authentication retries
Todd(config)#ip ssh authentication-retries 2
```

6. Connect to the VTY lines of the router:

```
Todd(config)#line vty 0 1180
```

7. Last, configure SSH and then Telnet as access protocols:

```
Todd(config-line)#transport input ssh telnet
```

If you do not use the keyword `telnet` at the end of the command string, then only SSH will work on the router. I suggest that you use just SSH if at all possible. Telnet is just too insecure for today's networks.

Router Interfaces

Interface configuration is one of the most important router configurations, because without interfaces, a router is pretty much a completely useless object. Plus, interface configurations must be totally precise to enable communication with other devices. Network layer addresses, media type, bandwidth, and other administrator commands are all used to configure an interface. Table 1.10 shows the commands.

TABLE 1.10 Router Interface Commands

Command	Meaning
`Todd(config)#interface fastethernet` *slot/port*	Enters interface configuration mode from global configuration mode.
`Todd(config)#interface serial` *slot/port*	Same as the previous command, except for serial interface 0/0/0 rather than Fast Ethernet interface.
`Todd(config-if)#shutdown`	Shuts down an interface.
`Todd(config-if)#no shutdown`	Enables an interface that is shut down.
`Todd(config-if)#ip address` *address mask*	Configures an interface with an IP address and a subnet mask.
`Todd(config-if)#ip address` *address mask* `secondary`	Adds a secondary IP address to an interface.
`Todd(config-if)#description description`	Adds a description to an interface.
`Todd(config-if)#clock rate rate`	Sets the clock rate of a serial interface in bits per second.
`Todd(config-if)#bandwidth rate`	Sets the bandwidth of an interface in kilobits per second.
`Todd#show interface interface`	Displays the status and configuration of the Fast Ethernet interface.
`Todd#clear counters interface`	Clears the display counters on the serial interface.

TABLE 1.10 Router Interface Commands *(continued)*

Command	Meaning
Todd#**sh ip interface interface**	Displays the layer-3 properties of an interface.
Todd#**sh ip int brief**	Displays the IP interfaces in a summarized table. This is one of the most useful show commands!

Let's take a look at setting both an IP address and a secondary IP address on an interface, and then we'll verify the configuration:

```
Todd(config)#interface fastEthernet 0/0
Todd(config)#interface serial 0/0/0
Todd(config-if)#shutdown
Jul 30 15:31:23.542: %LINK-5-CHANGED: Interface Serial0/0/0, changed state to
administratively down
Jul 30 15:31:24.542: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/
0, changed state to down
Todd(config-if)#no shutdown
Jul 30 15:31:27.566: %LINK-3-UPDOWN: Interface Serial0/0/0, changed state to up
Jul 30 15:31:28.566: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0,
changed state to up
Todd(config-if)#ip address 172.16.10.1 255.255.255.0
Todd(config-if)#ip address 172.16.20.1 255.255.255.0 ?
  secondary  Make this IP address a secondary address
  <cr>
Todd(config-if)#ip address 172.16.20.1 255.255.255.0 secondary
Todd(config-if)#description Serial link to Miami
Todd(config-if)#clock rate 1000000
Todd(config-if)#bandwidth 1000
Todd(config-if)#exit
Todd(config)#exit
Todd#show interface s0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Description: Serial link to Miami
  Internet address is 172.16.10.1/24
  MTU 1500 bytes, BW 1000 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:04, output 00:00:08, output hang never
```

```
  Last clearing of "show interface" counters 2d05h
  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
     34632 packets input, 2479012 bytes, 0 no buffer
     Received 34031 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     34372 packets output, 2303104 bytes, 0 underruns
     0 output errors, 0 collisions, 5 interface resets
     0 output buffer failures, 0 output buffers swapped out
     112 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

Todd#clear counters s0/0/0
Clear "show interface" counters on this interface [confirm][enter]
Jul 30 15:39:02.818: %CLEAR-5-COUNTERS: Clear counter on interface Serial0/0/0
by console
Todd#sh ip interface s0/0/0
Serial0/0/0 is up, line protocol is up
  Internet address is 172.16.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Secondary address 172.16.20.1/24
  Secondary address 172.16.30.1/24
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is enabled
  Local Proxy ARP is disabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is enabled
  IP Flow switching is disabled
  IP CEF switching is enabled
  IP CEF Fast switching turbo vector
```

```
  [output cut]
Todd#sh ip int brief
Interface         IP-Address      OK? Method Status        Protocol
FastEthernet0/0   192.168.21.1    YES manual up              up
FastEthernet0/1   unassigned      YES unset  administratively down down
Serial0/0/0       172.16.10.1     YES manual up              up
Serial0/0/1       unassigned      YES unset  administratively down down
Todd#
Viewing, Saving, and Erasing Configurations
```

Once you have gone to all the work of creating a configuration, you will need to know how to save it, and maybe even delete it. Table 1.11 shows the commands used to manipulate configurations.

TABLE 1.11 Commands to Manipulate Configurations

Command	Meaning
`Todd#copy running-config startup-config`	Saves the running configuration to NVRAM
`Todd#show running-config`	Shows the running configuration on the terminal
`Todd#show startup-config`	Shows the start-up configuration (in NVRAM) on the terminal
`Todd#erase startup-config`	Erases the configuration stored in NVRAM

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

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

Todd#show running-config
Building configuration...

Current configuration : 3343 bytes
!
version 12.4
[output cut]
```

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

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

Todd#erase startup-config
Erasing the nvram filesystem will remove all configuration files!
    Continue? [confirm][enter]
[OK]
Erase of nvram: complete
Todd#
*Feb 28 23:51:21.179: %SYS-7-NV_BLOCK_INIT: Initialized the geometry of nvram
Todd#sh startup-config
startup-config is not present
Todd#reload
Proceed with reload? [confirm]System configuration has been modified.
    Save? [yes/no]: n
```

Using the *show* Command

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

Table 1.12 shows some basic command you might use every day in a production environment.

TABLE 1.12 Everyday Commands

Command	Meaning
`show running config`	This shows the configuration that router is using.
`show startup-config`	This shows the configuration that the router will use when the router is reload.
`show interface`	This shows the status of all interfaces.

TABLE 1.12 Everyday Commands *(continued)*

Command	Meaning
`show ip interface`	This shows the default IP configuration on all interfaces.
`show ip interface brief`	This command provides a quick overview of the router's interfaces, including the logical address and status.
`show protocols`	This shows the status of layers 1 and 2 of each interface as well as the IP addresses used.
`show controllers`	This shows whether an interface is a DCE or DTE interface.

The `show running-config` command is important and could very well be one of the most used commands on a Cisco router. The `show running-config` command shows the configuration that the router is running. The `show startup-config` shows the backup config, or the configuration that will be used the next time the router is booted.

The `show interface` command is important, and that's what I'll discuss in this section. Here's what it looks like:

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

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

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

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

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

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

To enable the interface, use the command `no shutdown` from interface configuration mode:

```
Router#config t
Router(config)#int s0/0/0
Router(config-if)#no shutdown
```

You can get a nice brief overview of the interfaces with the `show interface brief` command:

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

Todd#sh protocols
Global values:
  Internet Protocol routing is enabled
FastEthernet0/0 is up, line protocol is up
  Internet address is 192.168.21.1/24
FastEthernet0/1 is administratively down, line protocol is down
Serial0/0/0 is up, line protocol is up
  Internet address is 172.16.10.1/24
Serial0/0/1 is administratively down, line protocol is down
Todd#
```

The `show controllers` command displays information about the physical interface itself. It'll also give you the type of serial cable plugged into a serial port. Usually, this will be only a DTE cable that plugs into a type of data service unit (DSU).

```
Router#sh controllers serial 0/0
HD unit 0, idb = 0x1229E4, driver structure at 0x127E70
buffer size 1524 HD unit 0, V.35 DTE cable
cpb = 0xE2, eda = 0x4140, cda = 0x4000

Router#sh controllers serial 0/1
HD unit 1, idb = 0x12C174, driver structure at 0x131600
buffer size 1524 HD unit 1, V.35 DCE cable
cpb = 0xE3, eda = 0x2940, cda = 0x2800
```