



Using the Command-Line Interface

The Cisco IOS command-line interface (CLI) is the primary user interface used for configuring, monitoring, and maintaining Cisco devices. This user interface allows you to directly and simply execute Cisco IOS commands, whether using a router console or terminal, or using remote access methods.

This chapter describes the basic features of the Cisco IOS CLI and how to use them. Topics covered include an introduction to Cisco IOS command modes, navigation and editing features, help features, and command history features.

Additional user interfaces include Setup mode (used for first-time startup), the Cisco Web Browser, and user menus configured by a system administrator. For information about Setup mode, see the [“Using AutoInstall and Setup”](#) chapter of this book. For information on issuing commands using the Cisco Web Browser, see the [“Using the Cisco Web Browser User Interface”](#) chapter of this book. For information on user menus, see the [“Managing Connections, Menus, and System Banners”](#) chapter of this book.

For a complete description of the user interface commands in this chapter, refer to the “Basic Command-Line Interface Commands” chapter of the Release 12.2 *Cisco IOS Configuration Fundamentals Command Reference*. To locate documentation of other commands that appear in this chapter, use the *Cisco IOS Command Reference Master Index* or search online.

This chapter contains the following sections:

- [Cisco IOS CLI Command Modes Overview](#)
- [Cisco IOS CLI Task List](#)
- [Using the Cisco IOS CLI Examples](#)

Cisco IOS CLI Command Modes Overview

To aid in the configuration of Cisco devices, the Cisco IOS command-line interface is divided into different command modes. Each command mode has its own set of commands available for the configuration, maintenance, and monitoring of router and network operations. The commands available to you at any given time depend on the mode you are in. Entering a question mark (?) at the system prompt (router prompt) allows you to obtain a list of commands available for each command mode.

The use of specific commands allows you to navigate from one command mode to another. The standard order that a user would access the modes is as follows: user EXEC mode; privileged EXEC mode; global configuration mode; specific configuration modes; configuration submodes; and configuration subsubmodes.

When you start a session on a router, you generally begin in *user EXEC mode*, which is one of two access levels of the EXEC mode. For security purposes, only a limited subset of EXEC commands are available in user EXEC mode. This level of access is reserved for tasks that do not change the configuration of the router, such as determining the router status.

In order to have access to all commands, you must enter *privileged EXEC mode*, which is the second level of access for the EXEC mode. Normally, you must enter a password to enter privileged EXEC mode. In privileged EXEC mode, you can enter any EXEC command, as the privileged EXEC mode is a superset of the user EXEC mode commands.

Most EXEC mode commands are one-time commands, such as **show** or **more** commands, which show the current configuration status, and **clear** commands, which clear counters or interfaces. EXEC mode commands are not saved across reboots of the router.

From privileged EXEC mode, you can enter *global configuration mode*. In this mode, you can enter commands that configure general system characteristics. You also can use global configuration mode to enter specific configuration modes. Configuration modes, including global configuration mode, allow you to make changes to the running configuration. If you later save the configuration, these commands are stored across router reboots.

From global configuration mode you can enter a variety of protocol-specific or feature-specific configuration modes. The CLI hierarchy requires that you enter these specific configuration modes only through global configuration mode. As an example, this chapter describes *interface configuration mode*, a commonly used configuration mode.

From configuration modes, you can enter configuration submodes. Configuration submodes are used for the configuration of specific features within the scope of a given configuration mode. As an example, this chapter describes the *subinterface configuration mode*, a submode of the interface configuration mode.

ROM monitor mode is a separate mode used when the router cannot boot properly. If your system (router, switch, or access server) does not find a valid system image to load when it is booting, the system will enter ROM monitor mode. ROM monitor (ROMMON) mode can also be accessed by interrupting the boot sequence during startup.

The following sections contain detailed information on these command modes:

- [User EXEC Mode](#)
- [Privileged EXEC Mode](#)
- [Global Configuration Mode](#)
- [Interface Configuration Mode](#)
- [Subinterface Configuration Mode](#)
- [ROM Monitor Mode](#)

[Table 1](#) follows these sections and summarizes the main Cisco IOS command modes.

User EXEC Mode

Logging in to the router places you in user EXEC command mode (unless the system is configured to take you immediately to privileged EXEC mode). Typically, log-in will require a user name and a password. You may try three times to enter a password before the connection attempt is refused.

**Note**

For information on setting the password, see the “Configuring Passwords and Privileges” chapter in the Release 12.2 *Cisco IOS Security Configuration Guide* and the “Using AutoInstall and Setup” chapter in this document.

The EXEC commands available at the user level are a subset of those available at the privileged level. In general, the user EXEC commands allow you to connect to remote devices, change terminal line settings on a temporary basis, perform basic tests, and list system information.

To list the available user EXEC commands, use the following command:

Command	Purpose
Router> ?	Lists the user EXEC commands.

The user EXEC mode prompt consists of the host name of the device followed by an angle bracket (>), as shown in the following example:

```
Router>
```

The default host name is generally `Router`, unless it has been changed during initial configuration using the **setup** EXEC command. You also change the host name using the **hostname** global configuration command.

**Note**

Examples in Cisco IOS documentation assume the use of the default name of “Router.” Different devices (for example, access servers) may use a different default name. If the routing device (router, access server, or switch) has been named with the **hostname** command, that name will appear as the prompt instead of the default name.

To list the commands available in user EXEC mode, enter a question mark (?) as shown in the following example:

```
Router> ?
Exec commands:
<1-99>          Session number to resume
connect         Open a terminal connection
disconnect     Disconnect an existing telnet session
enable         Turn on privileged commands
exit           Exit from the EXEC
help           Description of the interactive help system
lat            Open a lat connection
lock           Lock the terminal
login          Log in as a particular user
logout         Exit from the EXEC
menu           Start a menu-based user interface
mbranch        Trace multicast route for branch of tree
mrbranch       Trace reverse multicast route to branch of tree
mtrace         Trace multicast route to group
name-connection Name an existing telnet connection
pad            Open a X.29 PAD connection
ping           Send echo messages
resume         Resume an active telnet connection
show           Show running system information
systat         Display information about terminal lines
telnet         Open a telnet connection
terminal       Set terminal line parameters
tn3270         Open a tn3270 connection
```

```

trace          Trace route to destination
where         List active telnet connections
x3           Set X.3 parameters on PAD
xremote      Enter XRemote mode

```

The list of commands will vary depending on the software feature set and router platform you are using.



Note

You can enter commands in uppercase, lowercase, or mixed case. Only passwords are case sensitive. However, Cisco IOS documentation convention is to always present commands in lowercase.

Privileged EXEC Mode

Because many privileged EXEC mode commands set operating parameters, privileged-level access should be password protected to prevent unauthorized use. The privileged EXEC command set includes those commands contained in user EXEC mode. Privilege EXEC mode also provides access to configuration modes through the **configure** command, and includes advanced testing commands, such as **debug**.

The privileged EXEC mode prompt consists of the host name of the device followed by a pound sign (#), as shown in the following example:

```
Router#
```

To access privileged EXEC mode, use the following command:

Command	Purpose
Router> enable	Enables privileged EXEC mode. After issuing the enable command, the system will prompt you for a password.

Note that privileged EXEC mode is sometimes referred to as “enable mode,” because the **enable** command is used to enter the mode.

If a password has been configured on the system, you will be prompted to enter it before being allowed access to privileged EXEC mode. The password is not displayed on the screen and is case sensitive. If an enable password has not been set, privileged EXEC mode can be accessed only from the router console (terminal connected to the console port). The system administrator uses the **enable secret** or **enable password** global configuration commands to set the password that restricts access to privileged mode. For information on setting the passwords, see the “Configuring Passwords and Privileges” chapter in the Release 12.2 *Cisco IOS Security Configuration Guide*.

To return to user EXEC mode, use the following command:

Command	Purpose
Router# disable	Exits from privileged EXEC mode to user EXEC mode.

The following example shows the process of accessing privileged EXEC mode:

```

Router> enable
Password:<letmein>
Router#

```

Note that the password will not be displayed as you type, but is shown here for illustrational purposes. To list the commands available in privileged EXEC mode, issue the `?` command at the prompt. From privileged EXEC mode you can access global configuration mode, which is described in the following section.



Note

Because the privileged EXEC command set contains all of the commands available in user EXEC mode, some commands can be entered in either mode. In Cisco IOS documentation, commands that can be entered in either user EXEC mode or privileged EXEC mode are referred to as EXEC mode commands. If user or privileged is not specified in the documentation, assume that you can enter the referenced commands in either mode.

Global Configuration Mode

Global configuration mode is used to configure your system either globally or by entering specific configuration modes to configure specific elements, such as interfaces or protocols. Global configuration mode commands apply to features that affect the system as a whole. Use the **configure terminal** privileged EXEC command to enter global configuration mode.

To access global configuration mode, use the following command in privileged EXEC mode:

Command	Purpose
Router# configure terminal	From privileged EXEC mode, enters global configuration mode.

The following example shows the process of entering global configuration mode from privileged EXEC mode:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

Note that the system prompt changes to indicate that you are now in global configuration mode. The prompt for global configuration mode consists of the host-name of the device followed by `(config)` and the pound sign (`#`). To list the commands available in privileged EXEC mode, issue the `?` command at the prompt.

Commands entered in global configuration mode update the running configuration file as soon as they are entered. In other words, changes to the configuration take effect each time you press the Enter or Return key at the end of a valid command. However, these changes are not saved into the startup configuration file until you issue the **copy running-config startup-config** EXEC mode command. This behavior is explained in more detail later in this document.

As shown in the example above, the system dialogue prompts you to end your configuration session (exit configuration mode) by pressing the Control (Ctrl) and “z” keys simultaneously. When you press these keys, `^Z` is printed to the screen. Pressing the Ctrl-Z key combination is equivalent to entering the **end** command. The **end** command is used to indicate to the system that you are done with the current configuration session.



Tips

Pressing Ctrl-Z or entering the **end** command will always take you back to EXEC mode, regardless of which configuration mode you are in.

To exit global configuration command mode and return to privileged EXEC mode, use one of the following commands:

Command	Purpose
Router(config)# end or Router(config)# ^Z	Ends the current configuration session and returns to privileged EXEC mode.
Router(config)# exit	Exits the current command mode and returns to the preceding mode. For example, exits from global configuration mode to privileged EXEC mode.

From global configuration mode, you can enter a number of protocol-specific, platform-specific, and feature-specific configuration modes. For a complete list of configuration modes, see the “[Cisco IOS Command Modes](#)” appendix in this book. This appendix provides references to the appropriate documentation module for information about the specific configuration mode.

Interface configuration mode, described in the following section, is an example of a configuration mode you can enter from global configuration mode.

Interface Configuration Mode

One example of a specific configuration mode you enter from global configuration mode is interface configuration mode.

Many features are enabled on a per-interface basis. Interface configuration commands modify the operation of an interface such as an Ethernet, FDDI, or serial port. Interface configuration commands always follow an **interface** global configuration command, which defines the interface type.

For details on interface configuration commands that affect general interface parameters, such as bandwidth or clock rate, refer to the Release 12.2 *Cisco IOS Interface Configuration Guide*. For protocol-specific commands, refer to the appropriate Cisco IOS software command reference.

To access and list the interface configuration commands, use the following command:

Command	Purpose
Router(config)# interface <i>type number</i>	Specifies the interface to be configured, and enters interface configuration mode.

In the following example, the user enters interface configuration mode for serial interface 0. The new prompt, *hostname(config-if)#*, indicates interface configuration mode.

```
Router(config)# interface serial 0
Router(config-if)#
```

To exit interface configuration mode and return to global configuration mode, enter the **exit** command.

Configuration submodes are configuration modes entered from other configuration modes (besides global configuration mode). Configuration submodes are for the configuration of specific elements within the configuration mode. For a complete list of configuration submodes, see the “[Cisco IOS Command Modes](#)” appendix in this book. One example of a configuration submode is subinterface configuration mode, described in the following section.

Subinterface Configuration Mode

From interface configuration mode, you can enter subinterface configuration mode. Subinterface configuration mode is a submode of interface configuration mode. In subinterface configuration mode you can configure multiple virtual interfaces (called subinterfaces) on a single physical interface. Subinterfaces appear to be distinct physical interfaces to the various protocols. For example, Frame Relay networks provide multiple point-to-point links called permanent virtual circuits (PVCs). PVCs can be grouped under separate subinterfaces that in turn are configured on a single physical interface. From a bridging spanning-tree viewpoint, each subinterface is a separate bridge port, and a frame arriving on one subinterface can be sent out on another subinterface.

Subinterfaces also allow multiple encapsulations for a protocol on a single interface. For example, a router or access server can receive an ARPA-framed IPX packet and forward the packet back out the same physical interface as a SNAP-framed IPX packet.

For detailed information on how to configure subinterfaces, refer to the appropriate documentation module for a specific protocol in the Cisco IOS software documentation set.

To access subinterface configuration mode, use the following command in interface configuration mode:

Command	Purpose
Router(config-if)# interface <i>type number</i>	Specifies the virtual interface to be configured and enters subinterface configuration mode.

In the following example, a subinterface is configured for serial line 2, which is configured for Frame Relay encapsulation. The subinterface is identified as “2.1” to indicate that it is subinterface 1 of serial interface 2. The new prompt *hostname(config-subif)#* indicates subinterface configuration mode. The subinterface can be configured to support one or more Frame Relay PVCs.

```
Router(config)# interface serial 2
Router(config-if)# encapsulation frame-relay
Router(config-if)# interface serial 2.1
Router(config-subif)#
```

To exit subinterface configuration mode and return to interface configuration mode, use the **exit** command. To end your configuration session and return to privileged EXEC mode, press Ctrl-Z or enter the **end** command.

ROM Monitor Mode

ROM monitor mode (ROMMON) runs from a specialized software image, and is used to manually locate a valid system software image from which to boot the system (ROM monitor mode is also sometimes called “boot mode”).

If your system (router, switch, or access server) does not find a valid system image to load, the system will enter ROM monitor mode. ROM monitor mode can also be accessed by interrupting the boot sequence during startup. From ROM monitor mode, you can boot the device or perform diagnostic tests.

On most systems you can enter ROM monitor mode by entering the **reload EXEC** command and then issuing the Break command during the first 60 seconds of startup. The Break command is issued by pressing the Break key on your keyboard or by using the Break key-combination (the default Break key-combination is Ctrl-C).

**Note**

You must have a console connection to the router to perform this procedure, as Telnet connections will be lost when the system reboots.

To access ROM monitor mode from EXEC mode, perform the following steps:

-
- Step 1** Enter the **reload** command in EXEC mode. After issuing this command and responding to the system prompts as necessary, the system will begin reloading the system software image.
- Step 2** Issue the Break command during the first 60 seconds of system startup. The break command is issued using the Break key or Break key-combination. (The default Break key combination is Ctrl-C, but this may be configured differently on your system.) Issuing the break command interrupts the boot sequence and brings you into ROM monitor mode.
-

Another method for entering ROM monitor mode is to set the configuration register so that the router automatically enters ROM monitor mode when it boots. For information about setting the configuration register value, see the “Rebooting” chapter in this book.

ROM monitor mode uses an angle bracket (>) as the command line prompt. On some Cisco devices the default ROM monitor prompt is `rommon >`. A list of ROM monitor commands is displayed when you enter the `?` command or **help** command. The following example shows how this list of commands may appear:

```
User break detected at location 0x8162ac6\@
rommon 1 > ?
alias                set and display aliases command
boot                 boot up an external process
break                set/show/clear the breakpoint
confreg              configuration register utility
cont                 continue executing a downloaded image
context              display the context of a loaded image
cpu_card_type        display CPU card type
dev                  list the device table
dir                  list files in file system
dis                  disassemble instruction stream
frame                print out a selected stack frame
help                 monitor builtin command help
history              monitor command history
meminfo              main memory information
repeat               repeat a monitor command
reset                system reset
set                  show all monitor variables
stack                produce a stack trace
sync                 write monitor environment to NVRAM
sysret               print out info from last system return
unalias              unset an alias
unset                unset a monitor variable
rommon 2>
```

The list of available commands will vary depending on the software image and platform you are using. Some versions of ROMMON will display a list of commands in a pre-aliased format such as the following:

```
> ?
$ state              Toggle cache state (? for help)
B [filename] [TFTP Server IP address | TFTP Server Name]
                    Load and execute system image from ROM or from TFTP server
C [address]          Continue execution [optional address]
D /S M L V           Deposit value V of size S into location L with modifier M
```

```

E /S M L      Examine location L with size S with modifier M
G [address]   Begin execution
H             Help for commands
I             Initialize
K             Stack trace
L [filename] [TFTP Server IP address | TFTP Server Name]
              Load system image from ROM or from TFTP server, but do not
              begin execution
O             Show configuration register option settings
P             Set the break point
S             Single step next instruction
T function    Test device (? for help)
Deposit and Examine sizes may be B (byte), L (long) or S (short).
Modifiers may be R (register) or S (byte swap).
Register names are: D0-D7, A0-A6, SS, US, SR, and PC

```

To exit ROM monitor mode, use the **continue** command or **C** command alias; this will restart the booting process.

For more information on ROM monitor mode characteristics (including using aliases for commands) and using ROM monitor mode, see the “[Rebooting](#)” chapter in this document.

Summary of Main Cisco IOS Command Modes

[Table 1](#) summarizes the main command modes used in the Cisco IOS CLI. For a complete list of configuration modes, see the “[Cisco IOS Command Modes](#)” appendix in this book.

Table 1 Summary of the Main Cisco IOS Command Modes

Command Mode	Access Method	Prompt	Exit Method
User EXEC	Log in.	Router>	Use the logout command.
Privileged EXEC	From user EXEC mode, use the enable EXEC command.	Router#	To exit to user EXEC mode, use the disable command. To enter global configuration mode, use the configure terminal privileged EXEC command.
Global configuration	From privileged EXEC mode, use the configure terminal privileged EXEC command.	Router(config)#	To exit to privileged EXEC mode, use the end command or press Ctrl-Z . To enter interface configuration mode, use the interface configuration command.
Interface configuration	From global configuration mode, enter by specifying an interface with an interface command.	Router(config-if)#	To exit to global configuration mode, use the exit command. To exit to privileged EXEC mode, use the end command or press Ctrl-Z . To enter subinterface configuration mode, specify a subinterface with the interface command.

Table 1 Summary of the Main Cisco IOS Command Modes (continued)

Command Mode	Access Method	Prompt	Exit Method
Subinterface configuration	From interface configuration mode, specify a subinterface with an interface command. (The availability of this mode is dependent on your platform.)	Router(config-subif)#	To exit to global configuration mode, use the exit command. To exit to privileged EXEC mode, use the end command or press Ctrl-Z .
ROM monitor	From privileged EXEC mode, use the reload EXEC command. Press the Break key during the first 60 seconds while the system is booting.	> or boot> or rommon >	If you entered ROM monitor mode by interrupting the loading process, you can exit ROM monitor and resume loading by using the continue command or the C command alias.

Cisco IOS CLI Task List

To familiarize yourself with the features of the Cisco IOS CLI, perform any of the tasks described in the following sections:

- [Getting Context-Sensitive Help](#)
- [Using the no and default Forms of Commands](#)
- [Using Command History](#)
- [Using CLI Editing Features and Shortcuts](#)
- [Searching and Filtering CLI Output](#)

Getting Context-Sensitive Help

Entering a question mark (?) at the system prompt displays a list of commands available for each command mode. You also can get a list of the arguments and keywords available for any command with the context-sensitive help feature.

To get help specific to a command mode, a command name, a keyword, or an argument, use any of the following commands:

Command	Purpose
(prompt)# help	Displays a brief description of the help system.
(prompt)# <i>abbreviated-command-entry?</i>	Lists commands in the current mode that begin with a particular character string.
(prompt)# <i>abbreviated-command-entry</i> <Tab>	Completes a partial command name.
(prompt)# ?	Lists all commands available in the command mode.

Command	Purpose
<code>(prompt)# command ?</code>	Lists the available syntax options (arguments and keywords) for the command.
<code>(prompt)# command keyword ?</code>	Lists the next available syntax option for the command.

Note that the system prompt will vary depending on which configuration mode you are in.

When using context-sensitive help, the space (or lack of a space) before the question mark (?) is significant. To obtain a list of commands that begin with a particular character sequence, type in those characters followed immediately by the question mark (?). Do not include a space. This form of help is called *word help*, because it completes a word for you. For more information, see the “[Completing a Partial Command Name](#)” section later in this chapter.

To list keywords or arguments, enter a question mark (?) in place of a keyword or argument. Include a space before the ?. This form of help is called *command syntax help*, because it shows you which keywords or arguments are available based on the command, keywords, and arguments you already have entered.

You can abbreviate commands and keywords to the number of characters that allow a unique abbreviation. For example, you can abbreviate the **configure terminal** command to **confi t**. Because the abbreviated form of the command is unique, the router will accept the abbreviated form and execute the command.

Entering the **help** command (available in any command mode) will provide the following description of the help system:

```
Router# help
Help may be requested at any point in a command by entering
a question mark '?'. If nothing matches, the help list will
be empty and you must back up until entering a '?' shows the
available options.
Two styles of help are provided:
1. Full help is available when you are ready to enter a
   command argument (e.g. 'show ?') and describes each possible
   argument.
2. Partial help is provided when an abbreviated argument is entered
   and you want to know what arguments match the input
   (e.g. 'show pr?'.)
```

As described in the **help** command output, you can use the question mark (?) to complete a partial command name (partial help), or to obtain a list of arguments or keywords that will complete the current command.

The following example illustrates how the context-sensitive help feature enables you to create an access list from configuration mode.

Enter the letters **co** at the system prompt followed by a question mark (?). Do not leave a space between the last letter and the question mark. The system provides the commands that begin with **co**.

```
Router# co?
configure connect copy
```

Enter the **configure** command followed by a space and a question mark to list the keywords for the command and a brief explanation:

```
Router# configure ?
memory      Configure from NV memory
network     Configure from a TFTP network host
overwrite-network Overwrite NV memory from TFTP network host
terminal    Configure from the terminal
```

```
<cr>
```

The `<cr>` symbol (“cr” stands for carriage return) appears in the list to indicate that one of your options is to press the Return or Enter key to execute the command, without adding any additional keywords. In this example, the output indicates that your options for the `configure` command are **configure memory** (configure from NVRAM), **configure network** (configure from a file on the network), **configure overwrite-network** (configure from a file on the network and replace the file in NVRAM), or **configure terminal** (configure manually from the terminal connection). For most commands, the `<cr>` symbol is used to indicate that you can execute the command with the syntax you have already entered. However, the `configure` command is a special case, as the CLI will prompt you for the missing syntax:

```
Router# configure
Configuring from terminal, memory, or network [terminal]? terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

The default response for the `?` prompt is indicated in the CLI output by a bracketed option at the end of the line. In the preceding example, pressing the Enter (or Return) key is equivalent to typing in the word “terminal.”

To skip the prompting, enter the **configure terminal** command to enter global configuration mode:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

The CLI provides error isolation in the form of an error indicator, a caret symbol (^). The ^ symbol appears at the point in the command string where the user has entered incorrect or unrecognized command syntax. For example, the caret symbol in the following output shows the letter that was mistyped in the command:

```
Router# configure termiMal
      ^
% Invalid input detected at '^' marker.

Router#
```

Note that an error message (indicated by the % symbol) is printed to the screen to alert you to the error marker.

Enter the **access-list** command followed by a space and a question mark to list the available options for the command:

```
Router(config)# access-list ?
<1-99>      IP standard access list
<100-199>   IP extended access list
<1000-1099> IPX SAP access list
<1100-1199> Extended 48-bit MAC address access list
<200-299>   Protocol type-code access list
<300-399>   DECnet access list
<400-499>   XNS standard access list
<500-599>   XNS extended access list
<600-699>   Appletalk access list
<700-799>   48-bit MAC address access list
<800-899>   IPX standard access list
<900-999>   IPX extended access list
```

The two numbers within the angle brackets represent an inclusive range. Enter the access list number **99** and then enter another question mark to see the arguments that apply to the keyword and brief explanations:

```
Router(config)# access-list 99 ?
  deny    Specify packets to reject
  permit  Specify packets to forward
```

Enter the **deny** argument followed by a question mark (?) to list additional options:

```
Router(config)# access-list 99 deny ?
  A.B.C.D  Address to match
```

Generally, uppercase letters represent variables (arguments). Enter the IP address followed by a question mark (?) to list additional options:

```
Router(config)# access-list 99 deny 172.31.134.0 ?
  A.B.C.D  Mask of bits to ignore
  <cr>
```

In this output, A.B.C.D indicates that use of a wildcard mask is allowed. The wildcard mask is a method for matching IP addresses or ranges of IP addresses. For example, a wildcard mask of 0.0.0.255 matches any number in the range from 0 to 255 that appears in the fourth octet of an IP address.

Enter the wildcard mask followed by a question mark (?) to list further options.

```
Router(config)# access-list 99 deny 172.31.134.0 0.0.0.255 ?
<cr>
```

The <cr> symbol by itself indicates there are no more keywords or arguments. Press Enter (or Return) to execute the command.

```
Router(config)# access-list 99 deny 172.31.134.0 0.0.0.255
```

The system adds an entry to access list 99 that denies access to all hosts on subnet 172.31.134.0, while ignoring bits for IP addresses that end in 0 to 255.

Displaying All User EXEC Commands

To configure the current session to display the full set of user EXEC commands, use the following command in EXEC mode (user EXEC or privileged EXEC):

Command	Purpose
Router# terminal full-help	Configures this session to provide help for the full set of user-level commands.

The system administrator can also configure the system to always display full help for connections made to a particular line using the **full-help** line configuration command.

The **full-help** and **terminal full-help** commands enable the displaying of all help messages available in user EXEC mode when the **show ?** command is executed.

The following example is output for the **show ?** command with **terminal full-help** disabled and then enabled:

```
Router> terminal no full-help
Router> show ?

bootflash  Boot Flash information
calendar   Display the hardware calendar
clock      Display the system clock
context    Show context information
dialer     Dialer parameters and statistics
```

```

history      Display the session command history
hosts       IP domain-name, lookup style, nameservers, and host table
isdn        ISDN information
kerberos    Show Kerberos Values
modemcap    Show Modem Capabilities database
ppp         PPP parameters and statistics
rmon        rmon statistics
sessions    Information about Telnet connections
snmp        snmp statistics
terminal    Display terminal configuration parameters
users       Display information about terminal lines
version     System hardware and software status

```

```
Router> terminal full-help
```

```
Router> show ?
```

```

access-expression  List access expression
access-lists       List access lists
aliases            Display alias commands
apollo             Apollo network information
appletalk          AppleTalk information
arp                ARP table
async              Information on terminal lines used as router interfaces
bootflash          Boot Flash information
bridge             Bridge Forwarding/Filtering Database [verbose]
bsc                BSC interface information
bstun              BSTUN interface information
buffers            Buffer pool statistics
calendar           Display the hardware calendar
cdp                CDP information
clns               CLNS network information
clock              Display the system clock
cls                DLC user information
cmns               Connection-Mode networking services (CMNS) information
.
.
.
x25                X.25 information
xns                XNS information
xremote            XRemote statistics

```

Using the no and default Forms of Commands

Almost every configuration command has a **no** form. In general, use the **no** form to disable a feature or function. Use the command without the **no** keyword to reenable a disabled feature or to enable a feature that is disabled by default. For example, IP routing is enabled by default. To disable IP routing, use the **no ip routing** form of the **ip routing** command. To reenable it, use the plain **ip routing** form. The Cisco IOS software command reference publications describe the function of the **no** form of the command whenever a **no** form is available.

Many CLI commands also have a **default** form. By issuing the command **default command-name**, you can configure the command to its default setting. The Cisco IOS software command reference publications describe the function of the **default** form of the command when the **default** form performs a different function than the plain and **no** forms of the command. To see what default commands are available on your system, enter **default ?** in the appropriate command mode.

Using Command History

The Cisco IOS CLI provides a history or record of commands that you have entered. This feature is particularly useful for recalling long or complex commands or entries, including access lists. To use the command history feature, perform any of the tasks described in the following sections:

- [Setting the Command History Buffer Size](#)
- [Recalling Commands](#)
- [Disabling the Command History Feature](#)

Setting the Command History Buffer Size

By default, the system records ten command lines in its history buffer. To set the number of command lines that the system will record during the current terminal session, use the following command in EXEC mode:

Command	Purpose
Router# terminal history [<i>size number-of-lines</i>]	Enables the command history feature for the current terminal session.

The **terminal no history size** command resets the number of lines saved in the history buffer to the default of ten lines.

To configure the number of command lines the system will record for all sessions on a particular line, use the following command in line configuration mode:

Command	Purpose
Router(config-line)# history [<i>size number-of-lines</i>]	Enables the command history feature.

Recalling Commands

To recall commands from the history buffer, use one of the following commands or key combinations:

Command or Key Combination	Purpose
Ctrl-P or the Up Arrow key. ¹	Recalls commands in the history buffer, beginning with the most recent command. Repeat the key sequence to recall successively older commands.
Ctrl-N or the Down Arrow key. ¹	Returns to more recent commands in the history buffer after recalling commands with Ctrl-P or the Up Arrow key. Repeat the key sequence to recall successively more recent commands.
Router> show history	While in EXEC mode, lists the last several commands entered.

1. The arrow keys function only on ANSI-compatible terminals .

Disabling the Command History Feature

The command history feature is automatically enabled. To disable it during the current terminal session, use the following command in EXEC mode:

Command	Purpose
Router> <code>terminal no history</code>	Disables command history for the current session.

To configure a specific line so that the command history feature is disabled, use the following command in line configuration mode:

Command	Purpose
Router(config-line)# <code>no history</code>	Disables command history for the line.

Using CLI Editing Features and Shortcuts

A variety of shortcuts and editing features are enabled for the Cisco IOS CLI. The following subsections describe these features:

- [Moving the Cursor on the Command Line](#)
- [Completing a Partial Command Name](#)
- [Recalling Deleted Entries](#)
- [Editing Command Lines that Wrap](#)
- [Deleting Entries](#)
- [Continuing Output at the --More-- Prompt](#)
- [Redisplaying the Current Command Line](#)
- [Transposing Mistyped Characters](#)
- [Controlling Capitalization](#)
- [Designating a Keystroke as a Command Entry](#)
- [Disabling and Reenabling Editing Features](#)

Moving the Cursor on the Command Line

[Table 2](#) shows the key combinations or sequences you can use to move the cursor around on the command line to make corrections or changes. Ctrl indicates the Control key, which must be pressed simultaneously with its associated letter key. Esc indicates the Escape key, which must be pressed first, followed by its associated letter key. Keys are not case sensitive. Many letters used for CLI navigation and editing were chosen to provide an easy way of remembering their functions. In [Table 2](#) characters are bolded in the “Function Summary” column to indicate the relation between the letter used and the function.

Table 2 Key Combinations Used to Move the Cursor

Keystrokes	Function Summary	Function Details
Left Arrow or Ctrl-B	Back character	Moves the cursor one character to the left. When you enter a command that extends beyond a single line, you can press the Left Arrow or Ctrl-B keys repeatedly to scroll back toward the system prompt and verify the beginning of the command entry, or you can press the Ctrl-A key combination.
Right Arrow or Ctrl-F	Forward character	Moves the cursor one character to the right.
Esc, B	Back word	Moves the cursor back one word.
Esc, F	Forward word	Moves the cursor forward one word.
Ctrl-A	Beginning of line	Moves the cursor to the beginning of the line.
Ctrl-E	End of line	Moves the cursor to the end of the command line.

Completing a Partial Command Name

If you cannot remember a complete command name, or if you want to reduce the amount of typing you have to perform, enter the first few letters of the command, then press the Tab key. The command line parser will complete the command if the string entered is unique to the command mode. If your keyboard does not have a Tab key, press **Ctrl-I** instead.

The CLI will recognize a command once you have entered enough characters to make the command unique. For example, if you enter **conf** in privileged EXEC mode, the CLI will be able to associate your entry with the **configure** command, because only the **configure** command begins with **conf**.

In the following example the CLI recognizes the unique string for privileged EXEC mode of **conf** when the Tab key is pressed:

```
Router# conf<Tab>
Router# configure
```

When you use the command completion feature the CLI displays the full command name. The command is not executed until you use the Return or Enter key. This way you can modify the command if the full command was not what you intended by the abbreviation. If you enter a set of characters that could indicate more than one command, the system beeps to indicate that the text string is not unique.

If the CLI can not complete the command, enter a question mark (?) to obtain a list of commands that begin with that set of characters. Do not leave a space between the last letter you enter and the question mark (?).

For example, entering **co?** will list all commands available in the current command mode:

```
Router# co?
configure connect copy
Router# co
```

Note that the characters you enter before the question mark are reprinted to the screen to allow you to complete the command entry.

Deleting Entries

Use any of the following keys or key combinations to delete command entries if you make a mistake or change your mind:

Keystrokes	Purpose
Delete or Backspace	Deletes the character to the left of the cursor.
Ctrl-D	Deletes the character at the cursor.
Ctrl-K	Deletes all characters from the cursor to the end of the command line.
Ctrl-U or Ctrl-X	Deletes all characters from the cursor to the beginning of the command line.
Ctrl-W	Deletes the word to the left of the cursor.
Esc, D	Deletes from the cursor to the end of the word.

Recalling Deleted Entries

The CLI stores commands or keywords that you delete in a history buffer. Only character strings that begin or end with a space are stored in the buffer; individual characters that you delete (using Backspace or Ctrl-D) are not stored. The buffer stores the last ten items that have been deleted using Ctrl-K, Ctrl-U, or Ctrl-X. To recall these items and paste them in the command line, use the following key combinations:

Keystrokes	Purpose
Ctrl-Y	Recalls the most recent entry in the buffer (press keys simultaneously).
Esc, Y	Recalls the previous entry in the history buffer (press keys sequentially).

Note that the Esc, Y key sequence will not function unless you press the Ctrl-Y key combination first. If you press Esc, Y more than ten times, you will cycle back to the most recent entry in the buffer.

Editing Command Lines that Wrap

The CLI provides a wrap-around feature for commands that extend beyond a single line on the screen. When the cursor reaches the right margin, the command line shifts ten spaces to the left. You cannot see the first ten characters of the line, but you can scroll back and check the syntax at the beginning of the command. To scroll back, press Ctrl-B or the left arrow key repeatedly until you scroll back to the beginning of the command entry, or press Ctrl-A to return directly to the beginning of the line.

In the following example, the **access-list** command entry extends beyond one line. When the cursor first reaches the end of the line, the line is shifted ten spaces to the left and redisplayed. The dollar sign (\$) indicates that the line has been scrolled to the left. Each time the cursor reaches the end of the line, the line is again shifted ten spaces to the left.

```
Router(config)# access-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.1
Router(config)# $ 101 permit tcp 172.31.134.5 255.255.255.0 172.31.135.0 255.25
Router(config)# $t tcp 172.31.134.5 255.255.255.0 172.31.135.0 255.255.255.0 eq
```

```
Router(config)# $31.134.5 255.255.255.0 172.31.135.0 255.255.255.0 eq 45
```

When you have completed the entry, press **Ctrl-A** to check the complete syntax before pressing the Return key to execute the command. The dollar sign (\$) appears at the end of the line to indicate that the line has been scrolled to the right:

```
Router(config)# access-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.1$
```

The Cisco IOS software assumes you have a terminal screen that is 80 columns wide. If you have a different screen-width, use the **terminal width EXEC** command to set the width of your terminal.

Use line wrapping in conjunction with the command history feature to recall and modify previous complex command entries. See the [“Recalling Commands”](#) section in this chapter for information about recalling previous command entries.

Continuing Output at the --More-- Prompt

When working with the Cisco IOS CLI, output often extends beyond the visible screen length. For cases where output continues beyond the bottom of the screen, such as with the output of many **?**, **show**, or **more** commands, the output is paused and a --More-- prompt is displayed at the bottom of the screen. To resume output, press the Return key to scroll down one line, or press the Spacebar to display the next full screen of output.



Tips

If output is pausing on your screen, but you do not see the --More-- prompt, try entering a smaller value for the screen length using the **length** line configuration command or the **terminal length EXEC** command. Command output will not be paused if the **length** value is set to zero.

For information about filtering output from the --More-- prompt, see the [“Searching and Filtering CLI Output”](#) section in this chapter.

Redisplaying the Current Command Line

If you are entering a command and the system suddenly sends a message to your screen, you can easily recall your current command line entry. To redisplay the current command line (refresh the screen), use either of the following key combinations:

Keystrokes	Purpose
Ctrl-L or Ctrl-R	Redisplays the current command line.

Transposing Mistyped Characters

If you have mistyped a command entry, you can transpose the mistyped characters. To transpose characters, use the following key combination:

Keystrokes	Purpose
Ctrl-T	Transposes the character to the left of the cursor with the character located at the cursor.

Controlling Capitalization

You can capitalize or lowercase words or capitalize a set of letters with simple key sequences. Note, however, that Cisco IOS commands are generally case-insensitive, and are typically all in lowercase. To change the capitalization of commands, use any of the following key sequences:

Keystrokes	Purpose
Esc, C	Capitalizes the letter at the cursor.
Esc, L	Changes the word at the cursor to lowercase.
Esc, U	Capitalizes letters from the cursor to the end of the word.

Designating a Keystroke as a Command Entry

You can configure the system to recognize particular keystroke (key combination or sequence) as command aliases. In other words, you can set a keystroke as a shortcut for executing a command. To enable the system to interpret a keystroke as a command, use the either of the following key combinations before entering the command sequence:

Keystrokes	Purpose
Ctrl-V or Esc, Q	Configures the system to accept the following keystroke as a user-configured command entry (rather than as an editing command).

Disabling and Reenabling Editing Features

The editing features described in the previous sections were introduced in Cisco IOS Release 9.21, and are automatically enabled on your system. However, there may be some unique situations that could warrant disabling these editing features. For example, you may have scripts that conflict with editing functionality. To globally disable editing features, use the following command in line configuration mode:

Command	Purpose
Router(config-line)# no editing	Disables CLI editing features for a particular line.

To disable the editing features for the current terminal session, use the following command in EXEC mode:

Command	Purpose
Router# terminal no editing	Disables CLI editing features for the local line.

To reenable the editing features for the current terminal session, use the following command in EXEC mode:

Command	Purpose
Router# <code>terminal editing</code>	Enables the CLI editing features for the current terminal session.

To reenable the editing features for a specific line, use the following command in line configuration mode:

Command	Purpose
Router(config-line)# <code>editing</code>	Enables the CLI editing features.

Searching and Filtering CLI Output

The Cisco IOS CLI provides ways of searching through large amounts of command output and filtering output to exclude information you do not need. These features are enabled for **show** and **more** commands, which generally display large amounts of data.



Note

Show and **more** commands are always entered in EXEC mode.

When output continues beyond what is displayed on your screen, the Cisco IOS CLI displays a --More-- prompt. Pressing Return displays the next line; pressing the Spacebar displays the next screen of output. The CLI String Search feature allows you to search or filter output from --More-- prompts.

Understanding Regular Expressions

A regular expression is a pattern (a phrase, number, or more complex pattern) the CLI String Search feature matches against **show** or **more** command output. Regular expressions are case sensitive and allow for complex matching requirements. Simple regular expressions include entries like `Serial`, `misses`, or `138`. Complex regular expressions include entries like `00210...`, `(is)`, or `[Oo]utput`.

A regular expression can be a single-character pattern or a multiple-character pattern. That is, a regular expression can be a single character that matches the same single character in the command output or multiple characters that match the same multiple characters in the command output. The pattern in the command output is referred to as a string. This section describes creating both single-character patterns and multiple-character patterns. It also discusses creating more complex regular expressions using multipliers, alternation, anchoring, and parentheses.

Single-Character Patterns

The simplest regular expression is a single character that matches the same single character in the command output. You can use any letter (A–Z, a–z) or digit (0–9) as a single-character pattern. You can also use other keyboard characters (such as ! or ~) as single-character patterns, but certain keyboard characters have special meaning when used in regular expressions. [Table 3](#) lists the keyboard characters that have special meaning.

Table 3 Characters with Special Meaning

Character	Special Meaning
.	Matches any single character, including white space.
*	Matches 0 or more sequences of the pattern.
+	Matches 1 or more sequences of the pattern.
?	Matches 0 or 1 occurrences of the pattern.
^	Matches the beginning of the string.
\$	Matches the end of the string.
_ (underscore)	Matches a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), the beginning of the string, the end of the string, or a space.

To use these special characters as single-character patterns, remove the special meaning by preceding each character with a backslash (\). The following examples are single-character patterns matching a dollar sign, an underscore, and a plus sign, respectively.

```
\$ \_ \+
```

You can specify a range of single-character patterns to match against command output. For example, you can create a regular expression that matches a string containing one of the following letters: a, e, i, o, or u. Only one of these characters must exist in the string for pattern matching to succeed. To specify a range of single-character patterns, enclose the single-character patterns in square brackets ([]). For example, **[aeiou]** matches any one of the five vowels of the lowercase alphabet, while **[abcdABCD]** matches any one of the first four letters of the lower- or uppercase alphabet.

You can simplify ranges by entering only the endpoints of the range separated by a dash (-). Simplify the previous range as follows:

```
[a-dA-D]
```

To add a dash as a single-character pattern in your range, include another dash and precede it with a backslash:

```
[a-dA-D\-]
```

You can also include a right square bracket (]) as a single-character pattern in your range, as shown here:

```
[a-dA-D\-\]]
```

The previous example matches any one of the first four letters of the lower- or uppercase alphabet, a dash, or a right square bracket.

You can reverse the matching of the range by including a caret (^) at the start of the range. The following example matches any letter except the ones listed.

```
[^a-dqsv]
```

The following example matches anything except a right square bracket (]) or the letter d:

```
[^\]d]
```

Multiple-Character Patterns

When creating regular expressions, you can also specify a pattern containing multiple characters. You create multiple-character regular expressions by joining letters, digits, or keyboard characters that do not have special meaning. For example, `a4%` is a multiple-character regular expression. Put a backslash before the keyboard characters that have special meaning when you want to indicate that the character should be interpreted literally.

With multiple-character patterns, order is important. The regular expression `a4%` matches the character `a` followed by a `4` followed by a `%` sign. If the string does not have `a4%`, in that order, pattern matching fails. The multiple-character regular expression `a.` uses the special meaning of the period character to match the letter `a` followed by any single character. With this example, the strings `ab`, `a!`, or `a2` are all valid matches for the regular expression.

You can remove the special meaning of the period character by putting a backslash in front of it. For example, when the expression `a\.` is used in the command syntax, only the string `a.` will be matched.

You can create a multiple-character regular expression containing all letters, all digits, all keyboard characters, or a combination of letters, digits, and other keyboard characters. For example, `teletbit 3107 v32bis` is a valid regular expression.

Multipliers

You can create more complex regular expressions that instruct Cisco IOS software to match multiple occurrences of a specified regular expression. To do so, you use some special characters with your single-character and multiple-character patterns. Table 4 lists the special characters that specify “multiples” of a regular expression.

Table 4 Special Characters Used as Multipliers

Character	Description
*	Matches 0 or more single-character or multiple-character patterns.
+	Matches 1 or more single-character or multiple-character patterns.
?	Matches 0 or 1 occurrences of a single-character or multiple-character pattern.

The following example matches any number of occurrences of the letter `a`, including none:

`a*`

The following pattern requires that at least one letter `a` be in the string to be matched:

`a+`

The following pattern matches the string `bb` or `bab`:

`ba?b`

The following string matches any number of asterisks (`*`):

`**`

To use multipliers with multiple-character patterns, you enclose the pattern in parentheses. In the following example, the pattern matches any number of the multiple-character string `ab`:

`(ab)*`

As a more complex example, the following pattern matches one or more instances of alphanumeric pairs, but not none (that is, an empty string is not a match):

`([A-Za-z][0-9])+`

The order for matches using multipliers (*, +, or ?) is to put the longest construct first. Nested constructs are matched from outside to inside. Concatenated constructs are matched beginning at the left side of the construct. Thus, the regular expression matches A9b3, but not 9Ab3 because the letters are specified before the numbers.

Alternation

Alternation allows you to specify alternative patterns to match against a string. You separate the alternative patterns with a vertical bar (|). Exactly one of the alternatives can match the string. For example, the regular expression **codex|telebit** matches the string codex or the string telebit, but not both codex and telebit.

Anchoring

You can instruct Cisco IOS software to match a regular expression pattern against the beginning or the end of the string. That is, you can specify that the beginning or end of a string contain a specific pattern. You “anchor” these regular expressions to a portion of the string using the special characters shown in [Table 5](#).

Table 5 Special Characters Used for Anchoring

Character	Description
^	Matches the beginning of the string.
\$	Matches the end of the string.

For example, the regular expression **^con** matches any string that starts with con, and **\$sole** matches any string that ends with sole.

In addition to indicating the beginning of a string, the ^ symbol can be used to indicate the logical function “not” when used in a bracketed range. For example, the expression **[^abcd]** indicates a range that matches any single letter, as long as it is not the letters a, b, c, or d.

Contrast these anchoring characters with the special character underscore (_). Underscore matches the beginning of a string (^), the end of a string (\$), parentheses (()), space (), braces ({}), comma (,), or underscore (_). With the underscore character, you can specify that a pattern exist anywhere in the string. For example, **_1300_** matches any string that has 1300 somewhere in the string. The string 1300 can be preceded by or end with a space, brace, comma, or underscore. So, while **{1300_}** matches the regular expression **_1300_**, 21300 and 13000 do not.

Using the underscore character, you can replace long regular expression lists. For example, instead of specifying **^1300()1300\$ {1300, ,1300, {1300} ,1300, (1300** you can specify simply **_1300_**.

Parentheses for Recall

As shown in the “Multipliers” section, you use parentheses with multiple-character regular expressions to multiply the occurrence of a pattern. You can also use parentheses around a single- or multiple-character pattern to instruct the Cisco IOS software to remember a pattern for use elsewhere in the regular expression.

To create a regular expression that recalls a previous pattern, you use parentheses to indicate memory of a specific pattern and a backslash (\) followed by a digit to reuse the remembered pattern. The digit specifies the occurrence of a parentheses in the regular expression pattern. If you have more than one remembered pattern in your regular expression, then \1 indicates the first remembered pattern, and \2 indicates the second remembered pattern, and so on.

The following regular expression uses parentheses for recall:

a(.)bc(.)\1\2

This regular expression matches an a followed by any character (call it character no. 1), followed by bc followed by any character (character no. 2), followed by character no. 1 again, followed by character no. 2 again. So, the regular expression can match aZbcTZT. The software remembers that character no. 1 is Z and character no. 2 is T and then uses Z and T again later in the regular expression.

Searching and Filtering show Commands

To search **show** command output, use the following command in EXEC mode:

Command	Purpose
Router# show <i>any-command</i> begin <i>regular-expression</i>	Begins unfiltered output of the show command with the first line that contains the regular expression.



Note

Cisco IOS documentation generally uses the vertical bar to indicate a choice of syntax. However, to search the output of **show** and **more** commands, you will need to enter the pipe character (the vertical bar). In this section the pipe appears in bold (|) to indicate that you should enter this character.

To filter **show** command output, use one of the following commands in EXEC mode:

Command	Purpose
Router# show <i>any-command</i> exclude <i>regular-expression</i>	Displays output lines that do not contain the regular expression.
Router# show <i>any-command</i> include <i>regular-expression</i>	Displays output lines that contain the regular expression.

On most systems you can enter the Ctrl-Z key combination at any time to interrupt the output and return to EXEC mode. For example, you can enter the **show running-config | begin hostname** command to start the display of the running configuration file at the line containing the hostname setting, then use Ctrl-z when you get to the end of the information you are interested in.

Searching and Filtering more Commands

You can search **more** commands the same way you search **show** commands (**more** commands perform the same function as **show** commands). To search **more** command output, use the following command in EXEC mode:

Command	Purpose
Router# more <i>any-command</i> begin <i>regular-expression</i>	Begins unfiltered output of a more command with the first line that contains the regular expression.

You can filter **more** commands the same way you filter **show** commands. To filter **more** command output, use one of the following commands in EXEC mode:

Command	Purpose
Router# more <i>any-command</i> exclude <i>regular-expression</i>	Displays output lines that do not contain the regular expression.
Router# more <i>any-command</i> include <i>regular-expression</i>	Displays output lines that contain the regular expression.

Searching and Filtering from the --More-- Prompt

You can search output from --More-- prompts. To search **show** or **more** command output from a --More-- prompt, use the following command in EXEC mode:

Command	Purpose
-More- <i>/regular-expression</i>	Begins unfiltered output with the first line that contains the regular expression.

You can filter output from --More-- prompts. However, you can only specify one filter for each command. The filter remains until the **show** or **more** command output finishes or until you interrupt the output (using Ctrl-Z or Ctrl-6). Therefore, you cannot add a second filter at a --More-- prompt if you already specified a filter at the original command or at a previous --More-- prompt.



Note

Searching and filtering are different functions. You can search command output using the **begin** keyword and specify a filter at the --More-- prompt for the same command.

To filter **show** or **more** command output at a --More-- prompt, use one of the following commands in EXEC mode:

Command	Purpose
-More- <i>-regular-expression</i>	Displays output lines that do not contain the regular expression.
-More- <i>+regular-expression</i>	Displays output lines that contain the regular expression.

Using the Cisco IOS CLI Examples

The following sections provide examples of using the CLI:

- [Determining Command Syntax and Using Command History Example](#)
- [Searching and Filtering CLI Output Examples](#)

Determining Command Syntax and Using Command History Example

The CLI provides error isolation in the form of an error indicator, a caret symbol (^). The ^ symbol appears at the point in the command string where you have entered an incorrect command, keyword, or argument.

In the following example, suppose you want to set the clock. Use context-sensitive help to determine the correct command syntax for setting the clock.

```
Router# clock ?
  set Set the time and date
Router# clock
```

The help output shows that the **set** keyword is required. Determine the syntax for entering the time:

```
Router# clock set ?
hh:mm:ss Current time
Router# clock set
```

Enter the current time:

```
Router# clock set 13:32:00
% Incomplete command.
```

The system indicates that you need to provide additional arguments to complete the command. Press **Ctrl-P** or the Up Arrow to automatically repeat the previous command entry. Then add a space and question mark (?) to reveal the additional arguments:

```
Router# clock set 13:32:00 ?
<1-31> Day of the month
January Month of the year
February
March
April
May
June
July
August
September
October
November
December
```

Now you can complete the command entry:

```
Router# clock set 13:32:00 23 February 01
^
% Invalid input detected at '^' marker.
```

The caret symbol (^) and help response indicate an error at 01. To list the correct syntax, enter the command up to the point where the error occurred and then enter a question mark (?):

```
Router# clock set 13:32:00 23 February ?
<1993-2035> Year
Router# clock set 13:32:00 23 February
```

Enter the year using the correct syntax and press Enter or Return to execute the command:

```
Router# clock set 13:32:00 23 February 2001
```

Searching and Filtering CLI Output Examples

The following is partial sample output of the **more nvram:startup-config | begin EXEC** command that begins unfiltered output with the first line that contain the regular expression `ip` . At the `--More--` prompt, the user specifies a filter to exclude output lines that contain the regular expression `ip` .

```
Router# more nvram:startup-config | begin ip
ip subnet-zero
ip domain-name cisco.com
ip name-server 198.92.30.32
ip name-server 171.69.2.132
!
isdn switch-type primary-5ess
.
.
.
interface Ethernet1
ip address 5.5.5.99 255.255.255.0
--More--
-ip
filtering...
media-type 10BaseT
!
interface Serial0:23
encapsulation frame-relay
no keepalive
dialer string 4001
dialer-group 1
isdn switch-type primary-5ess
no fair-queue
```

The following is partial sample output of the **more nvram:startup-config | include** command. It only displays lines that contain the regular expression `ip` .

```
Router# more nvram:startup-config | include ip
ip subnet-zero
ip domain-name cisco.com
ip name-server 198.92.30.32
ip name-server 171.69.2.132
```

The following is partial sample output of the **more nvram:startup-config | exclude** command. It excludes lines that contain the regular expression `service`. At the `--More--` prompt, the user specifies a filter with the regular expression `Dialer1` . Specifying this filter resumes the output with the first line that contains `Dialer1` .

```
Router# more nvram:startup-config | exclude service
!
version 12.2
!
hostname router
!
boot system flash
no logging buffered
!
ip subnet-zero
ip domain-name cisco.com
.
.
.
--More--
/Dialer1
filtering...
```

```
interface Dialer1
  no ip address
  no ip directed-broadcast
  dialer in-band
  no cdp enable
```

The following is partial sample output of the **show interface EXEC** command with an output search specified. The use of the keywords **begin Ethernet** after the pipe begins unfiltered output with the first line that contains the regular expression `Ethernet` . At the `--More--` prompt, the user specifies a filter that displays only the lines that contain the regular expression `Serial` .

```
Router# show interface | begin Ethernet
Ethernet0 is up, line protocol is up
Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Description: ip address is 172.1.2.14 255.255.255.0
  Internet address is 172.1.2.14/24
.
.
.
    0 lost carrier, 0 no carrier
    0 output buffer failures, 0 output buffers swapped out
--More--
+Serial
filtering...
Serial1 is up, line protocol is up
Serial2 is up, line protocol is up
Serial3 is up, line protocol is down
Serial4 is down, line protocol is down
Serial5 is up, line protocol is up
Serial6 is up, line protocol is up
Serial7 is up, line protocol is up
```

The following is partial sample output of the **show buffers | exclude** command. It excludes lines that contain the regular expression `0 misses`. At the `--More--` prompt, the user specifies a search that continues the filtered output beginning with the first line that contains `Serial0`.

```
Router# show buffers | exclude 0 misses

Buffer elements:
  398 in free list (500 max allowed)
Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
  50 in free list (20 min, 150 max allowed)
  551 hits, 3 misses, 0 trims, 0 created
Big buffers, 1524 bytes (total 50, permanent 50):
  49 in free list (5 min, 150 max allowed)
Very Big buffers, 4520 bytes (total 10, permanent 10):
.
.
.
Huge buffers, 18024 bytes (total 0 permanent 0):
  0 in free list (0 min, 4 max allowed)
--More--
/Serial0
filtering...
Serial0 buffers, 1543 bytes (total 64, permanent 64):
  16 in free list (0 min, 64 max allowed)
  48 hits, 0 fallbacks
```

The following is partial sample output of the **show interface | include** command. The use of the **include (is)** keywords after the pipe (|) causes the command to display only lines that contain the regular expression (is). The parenthesis force the inclusion of the spaces before and after is. Use of the parenthesis ensures that only lines containing is with a space both before and after it will be included in the output (excluding from the search, for example, words like “disconnect”).

```
router# show interface | include ( is )
ATM0 is administratively down, line protocol is down
  Hardware is ATMizer BX-50
Dialer1 is up (spoofing), line protocol is up (spoofing)
  Hardware is Unknown
  DTR is pulsed for 1 seconds on reset
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Internet address is 172.21.53.199/24
Ethernet1 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.639c (bia 0060.837c.639c)
  Internet address is 5.5.5.99/24
Serial0:0 is down, line protocol is down
  Hardware is DSX1
.
.
.
--More--
```

At the --More-- prompt, the user specifies a search that continues the filtered output beginning with the first line that contains Serial0:13 :

```
/Serial0:13
filtering...
Serial0:13 is down, line protocol is down
  Hardware is DSX1
  Internet address is 11.0.0.2/8
    0 output errors, 0 collisions, 2 interface resets
  Timeslot(s) Used:14, Transmitter delay is 0 flag
```