



CHAPTER 1

Command-Line Interfaces

This chapter describes the command-line interfaces (CLI) available on the Catalyst 6500 series switches and contains these sections:

- [Switch CLI, page 1-1](#)
- [ROM Monitor CLI, page 1-16](#)

For information regarding the ATM CLI and commands, refer to the *ATM Software Configuration Guide and Command Reference—Catalyst 5000 Family and 6500 Series Switches* publication.

For information regarding the IDSM CLI and commands, refer to the *Catalyst 6500 Series Intrusion Detection System Module Installation and Configuration Note* publication.

For definitions of terms and acronyms listed in this publication, see [Appendix A, “Acronyms.”](#)

Switch CLI

Catalyst 6500 series switches are multimodule systems. The commands that you enter from the CLI can apply to the entire system or to a specific module, port, or VLAN.

You can configure and maintain the Catalyst 6500 series switches by entering commands from the switch CLI. The CLI is a basic command-line interpreter similar to the UNIX C shell. Using the CLI **session** command, you can access the router configuration software and perform tasks such as history substitution and alias creation.

Accessing the Switch CLI

You can access the switch CLI from a console terminal connected to an EIA/TIA-232 port or through a Telnet session. The CLI allows fixed band rates. Telnet sessions disconnect automatically after remaining idle for a user-defined time period.



Note

EIA/TIA-232 was known as RS-232 before its acceptance as a standard by the Electronic Industries Alliance and Telecommunications Industry Association.

Accessing the Switch CLI via the Console Port (EIA/TIA-232)

To access the switch through the console (EIA/TIA-232) port, perform these steps:

-
- Step 1** From the Cisco Systems Console prompt, press **Return**.
 - Step 2** At the prompt, enter the system password. The Console> prompt appears, indicating that you have accessed the CLI in normal mode.
 - Step 3** Enter the necessary commands to complete your desired tasks.
 - Step 4** When finished, exit the session by entering the **quit** command.
-

After connecting through the console port, you see this display:

```
Cisco Systems Console
Enter password:
Console> <password>
Console>
```

Accessing the Switch CLI via Telnet

To access the switch through a Telnet session, you must first set the IP address for the switch. You can open multiple sessions to the switch via Telnet.

To access the switch from a remote host with Telnet, perform these steps:

-
- Step 1** From the remote host, enter the **telnet** command and the host name or IP address of the switch that you want to access.
 - Step 2** At the prompt, enter the password for the CLI. If no password has been configured, press **Return**.
 - Step 3** Enter the necessary commands to complete your desired tasks.
 - Step 4** When finished, exit the Telnet session by entering the **quit** command.
-

After connecting through a Telnet session, you see this display:

```
host% telnet cat6000-1.cisco.com
Trying 172.16.44.30 ...
Connected to cat6000-1.
```

Operating the Switch CLI

This section describes command modes and functions that allow you to operate the switch CLI.

Accessing the Command Modes

The CLI has two modes of operation: normal and privileged. Both are password-protected. Use normal-mode commands for everyday system monitoring. Use privileged commands for system configuration and basic troubleshooting.

After you log in, the system enters normal mode, which gives you access to normal-mode commands only. You can enter privileged mode by entering the **enable** command followed by the enable password. Privileged mode is indicated by the word “enable” in the system prompt. To return to normal mode, enter the **disable** command at the prompt.

The following example shows how to enter privileged mode:

```
Console> enable
Enter password: <password>
Console> (enable)
```

Using Command-Line Processing

Switch commands are not case sensitive. You can abbreviate commands and parameters as long as they contain enough letters to be different from any other currently available commands or parameters. You can scroll through the last 20 commands stored in the history buffer and enter or edit the command at the prompt. (See [Table 1-1](#).)

Table 1-1 Command-Line Processing Keystroke

Keystroke	Function
Ctrl-A	Jumps to the first character of the command line.
Ctrl-B or the left arrow key	Moves the cursor back one character.
Ctrl-C	Escapes and terminates prompts and tasks.
Ctrl-D	Deletes the character at the cursor.
Ctrl-E	Jumps to the end of the current command line.
Ctrl-F or the right arrow key ¹	Moves the cursor forward one character.
Ctrl-K	Deletes from the cursor to the end of the command line.
Ctrl-L; Ctrl-R	Repeats current command line on a new line.
Ctrl-N or the down arrow key ¹	Enters next command line in the history buffer.
Ctrl-P or the up arrow key ¹	Enters previous command line in the history buffer.
Ctrl-U; Ctrl-X	Deletes from the cursor to the beginning of the command line.
Ctrl-W	Deletes last word typed.

Table 1-1 Command-Line Processing Keystroke (continued)

Keystroke	Function
Esc B	Moves the cursor back one word.
Esc D	Deletes from the cursor to the end of the word.
Esc F	Moves the cursor forward one word.
Delete key or Backspace key	Erases a mistake when entering a command; reenter the command after using this key.

1. The arrow keys function only on ANSI-compatible terminals such as VT100s.

Using the Command-Line Editing Features

Catalyst 6500 series switch software includes an enhanced editing mode that provides a set of editing key functions similar to those of the Emacs editor. You can enter commands in uppercase, lowercase, or a mix of both. Only passwords are case sensitive. You can abbreviate commands and keywords to the number of characters that allow a unique abbreviation.

For example, you can abbreviate the **show** command to **sh**. After entering the command at the system prompt, press **Return** to execute the command.

Moving Around on the Command Line

Perform one of these tasks to move the cursor around on the command line for corrections or changes:

Task	Keystrokes
Move the cursor back one character.	Press Ctrl-B or press the left arrow key ¹ .
Move the cursor forward one character.	Press Ctrl-F or press the right arrow key ¹ .
Move the cursor to the beginning of the command line.	Press Ctrl-A .
Move the cursor to the end of the command line.	Press Ctrl-E .
Move the cursor back one word.	Press Esc B .
Move the cursor forward one word.	Press Esc F .

1. The arrow keys function only on ANSI-compatible terminals such as VT100s.

Pasting in Buffer Entries

The system provides a buffer that contains the last ten items you deleted. You can recall these items and paste them in the command line by performing this task:

Task	Keystrokes
Recall the most recent entry in the buffer.	Press Ctrl-Y .
Recall the next buffer entry.	Press Esc Y .

The buffer contains only the last ten items you have deleted or cut. If you press **Esc Y** more than ten times, you cycle back to the first buffer entry.

Editing Command Lines That Wrap

The new editing command set provides a wraparound 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, perform this task:

Task	Keystrokes
Return to the beginning of a command line to verify that you have entered a lengthy command correctly.	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 ¹ .

1. The arrow keys function only on ANSI-compatible terminals such as VT100s.

Use line wrapping with the command history feature to recall and modify previous complex command entries. See the [“Using History Substitution” section on page 1-7](#) for information about recalling previous command entries.

The system assumes your terminal screen is 80 columns wide. If your screen has a different width, enter the **terminal width** command to tell the router the correct width of your screen.

Deleting Entries

Perform one of these tasks to delete command entries if you make a mistake or change your mind:

Task	Keystrokes
Erase the character to the left of the cursor.	Press the Delete or Backspace key.
Delete the character at the cursor.	Press Ctrl-D .
Delete from the cursor to the end of the command line.	Press Ctrl-K .
Delete from the cursor to the beginning of the command line.	Press Ctrl-U or Ctrl-X .
Delete the word to the left of the cursor.	Press Ctrl-W .
Delete from the cursor to the end of the word.	Press Esc D .

Scrolling Down a Line or a Screen

When you use the help facility to list the commands in a particular mode, the list is often longer than the terminal screen can display. In such cases, a ---More--- prompt is displayed at the bottom of the screen. To view the next line or screen, perform these tasks:

Task	Keystrokes
Scroll down one line.	Press the Return key.
Scroll down one screen.	Press the Spacebar .

**Note**

The ---More--- prompt is used for any output that has more lines than can be displayed on the terminal screen, including **show** command output.

Scrolling to Specified Text

If you enter */text* and press the **Return** key at the --More-- prompt, the display starts two lines above the line containing the *text* string. If the text string is not found, “Pattern Not Found” is displayed. You can also enter “**n**” at the --More-- prompt to search for the last entered *text* string. You can use this search method on all **show** commands that use the more buffer to display screen by screen output. The following is a list of **show** commands that do not use the more buffer and do not support this feature:

- **show cam**
- **show mls**
- **show tech-support**

Redisplaying the Current Command Line

If you enter a command and the system suddenly sends a message to your screen, you can recall your current command line entry. To do so, perform this task:

Task	Keystrokes
Redisplay the current command line.	Press Ctrl-L or Ctrl-R .

Transposing Mistyped Characters

If you mistype a command entry, you can transpose the mistyped characters by performing this task:

Task	Keystrokes
Transpose the character to the left of the cursor with the character located at the cursor.	Press Ctrl-T .

Controlling Capitalization

You can change words to uppercase or lowercase, or capitalize a set of letters, with simple keystroke sequences:

Task	Keystrokes
Capitalize at the cursor.	Press Esc C .
Change the word at the cursor to lowercase.	Press Esc L .
Capitalize letters from the cursor to the end of the word.	Press Esc U .

Designating a Keystroke as a Command Entry

You can use a particular keystroke as an executable command. Perform this task:

Task	Keystrokes
Insert a code to indicate to the system that the keystroke immediately following should be treated as a command entry, <i>not</i> an editing key.	Press Ctrl-V or Esc Q .

Using Command Aliases

Like regular commands, aliases are not case sensitive. However, unlike regular commands, some aliases cannot be abbreviated. See [Table 1-2](#) for a list of switch CLI aliases that cannot be abbreviated.

Table 1-2 Switch CLI Command Aliases

Alias	Command
batch	configure
di	show
earl	cam
exit	quit
logout	quit

Using History Substitution

Commands that you enter during each terminal session are stored in a history buffer, which stores the last 20 commands you entered during a terminal session. History substitution allows you to access these commands without retyping them by using special abbreviated commands. (See [Table 1-3](#).)

Table 1-3 History Substitution Commands

Command	Function
To repeat recent commands:	
!!	Repeat the most recent command.
!-nn	Repeat the <i>nn</i> th most recent command.
!n	Repeat command <i>n</i> .
!aaa	Repeat the command beginning with string <i>aaa</i> .

Table 1-3 History Substitution Commands (continued)

Command	Function
! ?aaa	Repeat the command containing the string aaa.
To modify and repeat the most recent command:	
^aaa^bbb	Replace string aaa with string bbb in the most recent command.
To add a string to the end of a previous command and repeat it:	
!!aaa	Add string aaa to the end of the most recent command.
!n aaa	Add string aaa to the end of command n.
!aaa bbb	Add string bbb to the end of the command beginning with string aaa.
!?aaa bbb	Add string bbb to the end of the command containing string aaa.

Accessing Command Help

To see a list of top-level commands and command categories, type **help** in normal or privileged mode. Context-sensitive help (usage and syntax information) for individual commands can be seen by appending **help** to any specific command. If you enter a command using the wrong number of arguments or inappropriate arguments, usage and syntax information for that command is displayed. Additionally, appending **help** to a command category displays a list of commands in that category.

Top-Level Commands and Command Categories

In normal mode, use the **help** command to display a list of top-level commands and command categories, as follows:

```

Console> help
Commands:
-----
cd                Set default flash device
dir              Show list of files on flash device
enable          Enable privileged mode
help            Show this help screen
history         Show contents of history substitution buffer
l2trace         Layer2 trace between hosts
ping           Send echo packets to hosts
pwd            Show default flash device
quit          Exit from the Admin session
session       Tunnel to ATM or Router module
set           Set commands, use 'set help' for more info
show         Show commands, use 'show help' for more info
traceroute   Trace the route to a host
verify       Verify checksum of file on flash device
wait         Wait for x seconds
whichboot    Which file booted
Console>

```

In privileged mode, enter the **help** command to display a list of top-level commands and command categories, as follows:

```

Console> (enable) help
Commands:
-----

```

```

cd                Set default flash device
clear             Clear, use 'clear help' for more info
commit           Commit ACL to hardware and NVRAM
configure        Configure system from network
copy             Copy files between TFTP/RCP/module/flash devices
delete           Delete a file on flash device
dir              Show list of files on flash device
disable          Disable privileged mode
disconnect       Disconnect user session
download         Download code to a processor
enable           Enable privileged mode
format           Format a flash device
help             Show this help screen
history          Show contents of history substitution buffer
l2trace          Layer2 trace between hosts
ping             Send echo packets to hosts
pwd              Show default flash device
quit             Exit from the Admin session
reconfirm        Reconfirm VMPS
reload           Force software reload to linecard
reset            Reset system or module
rollback         Rollback changes made to ACL in editbuffer
session          Tunnel to ATM or Router module
set              Set commands, use 'set help' for more info
show             Show commands, use 'show help' for more info
slip             Attach/detach Serial Line IP interface
squeeze          Reclaim space used by deleted files
switch           Switch to standby <clock|supervisor>
telnet           Telnet to a remote host
test             Test command, use 'test help' for more info
undelete         Undelete a file on flash device
upload           Upload code from a processor
verify           Verify checksum of file on flash device
wait             Wait for x seconds
whichboot        Which file booted
write            Write system configuration to terminal/network
Console> (enable)

```

Command Categories

On some commands (such as **clear**, **set**, and **show**), typing **help** after the command provides a list of commands in that category. For example, this display shows a partial list of commands for the **clear** category:

```
Console> (enable) clear help
```

```
Clear commands:
```

```

-----
clear alias          Clear aliases of commands
clear arp            Clear ARP table entries
clear banner         Clear Message Of The Day banner
clear boot           Clear booting environment variable
clear cam            Clear CAM table entries
clear channel        Clear PAgP statistical information
.
.
.

```

Context-Sensitive Help

Usage and syntax information for individual commands can be seen by appending **help** to any specific command. For example, the following display shows usage and syntax information for the **set length** command:

```
Console> set length help
Usage: set length <screenlength> [default]
      (screenlength = 5..512, 0 to disable 'more' feature)
Console>
```

Designating Modules, Ports, and VLANs

The Catalyst 6500 series modules (module slots), ports, and VLANs are numbered starting with 1. The supervisor engine module is module 1, residing in the top slot. On each module, port 1 is the leftmost port. To reference a specific port on a specific module, the command syntax is *mod/port*. For example, **3/1** denotes module 3, port 1. In some commands, such as **set trunk**, **set cam**, and **set vlan**, you can enter lists of ports and VLANs.

You can designate ports by entering the module and port number pairs, separated by commas. To specify a range of ports, use a dash (-) between the module number and port number pairs. Dashes take precedence over commas. The following examples show several ways of designating ports:

Example 1: **2/1,2/3** denotes module 2, port 1 and module 2, port 3.

Example 2: **2/1-12** denotes module 2, ports 1 through 12.

Example 3: **2/1-2/12** also denotes module 2, ports 1 through 12.

Each VLAN is designated by a single number. You can specify lists of VLANs the same way you do for ports. Individual VLANs are separated by commas (,); ranges are separated by dashes (-). In the following example, VLANs 1 through 10 and VLAN 1000 are specified:

```
1-10,1000
```

Designating MAC Addresses, IP and IPX Addresses, and IP Aliases

Some commands require a MAC address that you must designate in a standard format. The MAC address format must be six hexadecimal numbers separated by hyphens, as shown in this example:

```
00-00-0c-24-d2-fe
```

Some commands require an IP address. The IP address format is 32 bits, written as four octets separated by periods (dotted decimal format). IP addresses are made up of a network section, an optional subnet section, and a host section, as shown in this example:

```
126.2.54.1
```

If DNS is configured properly on the switch, you can use IP host names instead of IP addresses. For information on configuring DNS, refer to the *Catalyst 6500 Series Switch Software Configuration Guide*.

If the IP alias table is configured, you can use IP aliases in place of the dotted decimal IP address. This is true for most commands that use an IP address, except commands that define the IP address or IP alias.

When entering the IPX address syntax, use the following format:

- IPX net address—1..FFFFFFFE
- IPX node address—x.x.x where x is 0..FFFF
- IPX address—ipx_net.ipx_node (for example 3.0034.1245.AB45, A43.0000.0000.0001)

Using Command Completion Features

The command completion features consist of these functions:

- [Using Command Self-Repeat](#)
- [Using Keyword Lookup](#)
- [Using Partial Keyword Lookup](#)
- [Using Command Completion](#)

Using Command Self-Repeat

Use the command self-repeat function to display matches to all possible keywords if a string represents a unique match. If a unique match is not found, the longest matching string is provided. To display the matches, enter a space after the last parameter and enter ?. Once the matches are displayed, the system comes back to the prompt and displays the last command without the ?. In the following example, notice how the system repeats the command entered without the ?:

```
Console> (enable) set mls nde
disable          Disable multilayer switching data export filter
enable          Enable multilayer switching data export filter
engineer        Engineer setting of the export filter
flow            Setting multilayer switching export filter
<collector_ip>  IP address
Console> (enable) set mls nde
```

Using Keyword Lookup

Use the keyword-lookup function to display a list of valid keywords and arguments for a command. To display the matches, enter a space after the last parameter and enter ?. For example, five parameters are used by the **set mls** command. To see these parameters, enter **set mls ?** at the privileged prompt. In the following example, notice how the system repeats the command entered without the ?:

```
Console> (enable) set mls ?
agingtime       Set agingtime for MLS cache entry
exclude         Set MLS excluded protocol ports
flow            Set minimum flow mask
nde             Configure Netflow Data Export
statistics      Add protocols to protocol statistics list
Console> (enable) set mls
```

Using Partial Keyword Lookup

Use the partial keyword-lookup function to display a list of commands that begin with a specific set of characters. To display the matches, enter ? immediately after the last parameter. For example, enter **co?** at the privileged prompt to display a list of commands that start with **co**. The system displays all commands that begin with **co** and repeats the command entered without the ?:

```
Console> (enable) co?
commit          Commit ACL to hardware and NVRAM
configure       Configure system from network
copy            Copy files between TFTP/RCP/module/flash devices
Console> (enable) CO
```

Using Command Completion

Use the command completion function to complete a command or keyword. When you enter a unique partial character string and press **Tab**, the system completes the command or keyword on the command line. For example, if you enter **co** at the privileged prompt and press **Tab**, the system completes the command as **configure** because it is the only command that matches the criteria.

If no completion can be done, no action is carried out and the system returns to the prompt and the last command. The cursor appears immediately after the keyword, allowing you to enter additional information.

Using the CLI String Search

The pattern in the command output is referred to as a string. The CLI string search feature allows you to search or filter any **show** or **more** command output and allows you to search and filter at --More-- prompts. This feature is useful when you need to sort through large amounts of output or if you want to exclude output that you do not need to see.

With the search function, you can begin unfiltered output at the first line that contains a regular expression you specify. You can then specify a maximum of one filter per command or start a new search from the --More-- prompt.

A regular expression is a pattern (a phrase, number, or more complex pattern) that software uses to match against **show** or **more** command output. Regular expressions are case sensitive and allow for complex matching requirements. Examples of simple regular expressions are `Serial`, `misses`, and `138`. Examples of complex regular expressions are `00210...`, `(is)`, and `[Oo]utput`.

You can perform three types of filtering:

- Use the **begin** keyword to begin output with the line that contains a specified regular expression.
- Use the **include** keyword to include output lines that contain a specified regular expression.
- Use the **exclude** keyword to exclude output lines that contain a specified regular expression.

You can then search this filtered output at the --More-- prompts.



Note

The CLI string search function does not allow you to search or filter backward through previous output; filtering cannot be specified using HTTP access to the CLI.

Regular Expressions

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. This section describes how to create both single-character patterns and multiple-character patterns and how to create 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 1-4](#) lists the keyboard characters with special meaning.

Table 1-4 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 word delimiter. All alphanumeric characters and the underscore itself (_) form a word.

To enter these special characters as single-character patterns, remove the special meaning by preceding each character with a backslash (\). These 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. One and 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 end points 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. To do so, enter the following:

```
[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. This example matches any letter except the ones listed:

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

This 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 in front of the keyboard characters that have special meaning when you want to remove their special meaning.

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. This 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. In the following expression

a\.

only the string `a.` matches this regular expression.

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. These examples are all valid regular expressions:

telebit 3107 v32bis

Multipliers

You can create more complex regular expressions to match multiple occurrences of a specified regular expression by using some special characters with your single- and multiple-character patterns. [Table 1-5](#) lists the special characters that specify “multiples” of a regular expression.

Table 1-5 Special Characters Used as Multipliers

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

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

a*

This pattern requires that at least one letter `a` in the string is matched:

a+

This pattern matches the string `bb` or `bab`:

ba?b

This 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, this 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 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 contains a specific pattern. You “anchor” these regular expressions to a portion of the string using the special characters shown in [Table 1-6](#).

Table 1-6 Special Characters Used for Anchoring

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

This regular expression matches a string only if the string starts with abcd:

^abcd

In contrast, this expression is in a range that matches any single letter, as long as it is not the letters a, b, c, or d:

[^abcd]

With this example, the regular expression matches a string that ends with .12:

\$.12

Contrast these anchoring characters with the special character underscore (_). The 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's 1300 can be preceded by or end with a space, brace, or comma. For example:

`{1300-` or `{1300:`

matches the regular expression, but 21300 and 13000 do not.

Using the underscore character, you can replace long regular expression lists, such as the following:

`^1300$ ^1300(space) (space)1300 {1300, ,1300, {1300} ,1300, (1300`

with

`_1300_`

ROM Monitor CLI

The ROM monitor is a ROM-based program that executes upon platform startup, reset, or when a fatal exception occurs.

Accessing the ROM Monitor CLI

The system enters ROM-monitor mode if the switch does not find a valid system image, if the NVRAM configuration is corrupted, or if the configuration register is set to enter ROM-monitor mode. From the ROM-monitor mode, you can load a system image manually from Flash memory, from a network server file, or from bootflash. You can also enter ROM-monitor mode by restarting the switch and pressing the **Break** key during the first 60 seconds of startup.



Note

Break is always enabled for 60 seconds after rebooting the system, regardless of whether Break is configured to be off by configuration register settings.

To connect through a terminal server, escape to the Telnet prompt, and enter the **send break** command to break back to the ROM-monitor mode.

Operating the ROM Monitor CLI

The ROM monitor commands are used to load and copy system images, microcode images, and configuration files. System images contain the system software. Microcode images contain microcode to be downloaded to various hardware devices. Configuration files contain commands to customize Catalyst 6500 series software.

The manual **boot** command has the following syntax:

**Note**

Enter the **copy** *file-id* {**tftp** | **flash** | *file-id*} command to obtain an image from the network.

- **boot**—Boot from ROM
- **boot** [-*xv*] [*device:*][*imagename*]—Boot from the local device. If you do not specify an image name, the system defaults to the first valid file in the device. The image name is case sensitive.

Once you are in ROM-monitor mode, the prompt changes to rommon 1>. While you are in ROM-monitor mode, each time you enter a command, the number in the prompt increments by one.

