



CLI Tips, Techniques, and Shortcuts

This chapter describes techniques for using the command-line interface (CLI) of the Cisco IOS XR software.

Contents

The chapter contains the following sections:

- [CLI Tips and Shortcuts, page 4-1](#)
- [Displaying System Information with show Commands, page 4-5](#)
- [Wildcards, Templates, and Aliases, page 4-10](#)
- [Command History, page 4-17](#)
- [Key Combinations, page 4-18](#)



Note

Commands can be entered in uppercase, lowercase, or mixed case. Only passwords are case sensitive. However, the Cisco Systems documentation convention presents commands in lowercase.

CLI Tips and Shortcuts

The following sections describe tips and shortcuts useful when using the CLI:

- [Entering Abbreviated Commands, page 4-2](#)
- [Using the Question Mark \(?\) to Display On-Screen Command Help, page 4-2](#)
- [Completing a Partial Command with the Tab Key, page 4-4](#)
- [Identifying Command Syntax Errors, page 4-4](#)
- [Using the no Form of a Command, page 4-4](#)
- [Editing Command Lines that Wrap, page 4-5](#)

Entering Abbreviated Commands

You can abbreviate commands and keywords to the number of characters that allow a unique abbreviation. For example, the **configure** command can be abbreviated as **confi** because the abbreviated form of the command is unique. The router accepts and executes the abbreviated command.

Using the Question Mark (?) to Display On-Screen Command Help

Use the question mark (?) to learn what commands are available and the correct syntax for a command. [Table 4-1](#) summarizes the options for on-screen help.



Tip

The space (or lack of a space) before the question mark (?) is significant. If you include a space before the question mark, the system displays all available options for a command or CLI mode. If you do not include a space, the system displays a list of commands that begin with a particular character string.

Table 4-1 On-Screen Help Commands

Command	Description
<i>partial-command?</i>	<p>Enter a question mark (?) at the end of a partial command to list the commands that begin with those characters.</p> <pre>RP/0/RP0/CPU0:router# co?</pre> <p>configure copy</p> <p>Note Do not include a space between the command and question mark.</p>
<i>?</i>	Lists all commands available for a particular command mode.
<i>command ?</i>	<p>Include a space before the question mark (?) to list the keywords and arguments that belong to a command.</p> <pre>RP/0/RP0/CPU0:router# configure ?</pre> <pre> exclusive Configure exclusively from this terminal memory Configure from NV memory network Configure from TFTP network host overwrite-network Overwrite NV memory from TFTP network host terminal Configure from the terminal <cr> </pre> <p>Note For most commands, the <cr> symbol indicates that you can execute the command with the syntax already entered. For the preceding example, press Return to enter global configuration mode.</p>
<i>command keyword ?</i>	<p>Enter a question mark (?) after the keyword to list the next available syntax option for the command.</p> <pre>RP/0/RP0/CPU0:router# show aaa ?</pre> <pre> taskgroup Show all the local taskgroups configured in the system userdb Show all local users with the usergroups each belong to usergroup Show all the local usergroups configured in the system </pre> <p>Note Include a space between the keyword and question mark.</p>

The following example shows how to add an entry to access list 99. The added entry denies access to all hosts on subnet 172.0.0.0 and ignores bits for IPv4 addresses that start within the range of 0 to 255. The following steps provide an example of on-screen command help:

- Step 1** Enter the **access-list** command, followed by a space and a question mark, to list the available options for the command:

```
RP/0/RP0/CPU0:router(config)# access-list ?

<1-199>      IP Access list
<1300-2699> IP Access list
```



Note The number ranges (within the angle brackets) are inclusive ranges.

- Step 2** Enter the access list number **99**, followed by a space and another question mark, to display the arguments that apply to the keyword and brief explanations:

```
RP/0/RP0/CPU0:router(config)# access-list 99 ?

<1-2147483647> Sequence number for this entry
deny           Specifies packets to reject
permit        Specifies packets to forward
remark        Comment for access list
<cr>
RP/0/RP0/CPU0:router(config)# access-list 99 deny ?

A.B.C.D Address to match
```

- Step 3** Generally, uppercase letters represent variables (arguments). Enter the IP address, followed by a space and a question mark (?), to list additional options:

```
RP/0/RP0/CPU0: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 mask is allowed. The mask is a method for matching IP addresses or ranges of IP addresses.

For example, a mask of 255.0.0.0 matches any number in the range from 0 to 255 that appears in the first octet of an IP address. Enter the mask, followed by a space and a question mark (?), to list further options.

```
RP/0/RP0/CPU0:router(config)# access-list 99 deny 172.31.134.0 255.0.0.0 ?
<cr>
```

- Step 4** The <cr> symbol by itself indicates that there are no more keywords or arguments. Press **Return** to execute the command:

```
RP/0/RP0/CPU0:router(config)# access-list 99 deny 172.31.134.0 255.0.0.0
```



Note The configuration does not become active until you enter the **commit** command to add the target configuration to the running configuration.

Completing a Partial Command with the Tab Key

If you cannot remember a complete command name or want to reduce the amount of typing you have to perform, enter the first few letters of the command, then press the Tab key. If only one command begins with that character string, the system completes the command for you. If the characters you entered indicate more than one command, the system beeps to indicate that the text string is not unique and the system provides a list of commands that match the text entered.

In the following example, the CLI recognizes **conf** as a unique string in EXEC mode and completes the command when Tab is pressed:

```
RP/0/RP0/CPU0:router# conf<Tab>
RP/0/RP0/CPU0:router# configure
```

The CLI displays the full command name. You must then press **Return** to execute the command. This feature allows you to modify or reject the suggested command.

In the next example, the CLI recognizes two commands that match the text entered:

```
RP/0/RP1/CPU0:router#co<Tab>
configure copy
RP/0/RP1/CPU0:router#con<Tab>
RP/0/RP1/CPU0:router#configure
```



Tip

If your keyboard does not have a Tab key, press Ctrl-I instead.

Identifying Command Syntax Errors

If an incorrect command is entered, an error message is returned with the caret (^) at the point of the error. In the following example, the caret appears where the character was typed incorrectly in the command:

```
RP/0/0/CPU0:router# configure termiMal
                                     ^
% Invalid input detected at '^' marker.
```



Note

The percent sign (%) indicates the line in which the error message occurred.

To display the correct command syntax, enter the “?” after the command:

```
RP/0/0/CPU0:router# configure ?
exclusive  Configure exclusively from this terminal
terminal   Configure from the terminal
<cr>
```

Using the no Form of a Command

Almost every configuration command has a **no** form. Depending on the command, the **no** form may enable or disable a feature. For example, when configuring an interface, the **no shutdown** command brings up the interface, and the **shutdown** command shuts down the interface. The **route ipv4** command creates a static route, and the **no route ipv4** command deletes a route when entered with the same parameters as an existing route.

The Cisco IOS XR software command reference publications provide the complete syntax for the configuration commands and describe what the **no** form of a command does. See the “[Related Documents](#)” section on page xv for more information.

Editing Command Lines that Wrap

The CLI 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. The first ten characters of the line are not shown, but it is possible to scroll back and check the syntax at the beginning of the command. To scroll back, press Ctrl-B or the left arrow key repeatedly, or press Ctrl-A to return directly to the beginning of the line.

In the following example, the **ipv4 access-list** command entry is too long to display on one line. When the cursor reaches the end of the line, the line is shifted to the left and redisplayed. The dollar sign (\$) after the command prompt indicates that the line has been scrolled to the left and the beginning of the command is hidden.

```
RP/0/0/CPU0:router(config)# $s-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.135.0
```

In the next example, Ctrl-A is used to display the beginning of the command line, and the dollar sign at the end of the command line shows the command has been scrolled to the right and the end of the command is hidden.

```
RP/0/0/CPU0:router(config)# ipv4 access-list 101 permit tcp 172.31.134.5 255.255.255.0 17$
```

In the next example, the right arrow key has been used to scroll to the right. Notice that dollar sign symbols appear at both ends of the line, which indicates that command information is hidden from the beginning and end of the command.

```
RP/0/0/CPU0:router(config)# $ccess-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.$
```

By default, the Cisco IOS XR software uses a terminal screen 80 columns wide. To adjust for a different screen width, use the **terminal width** command in EXEC mode.

Use line wrapping with the command history feature to recall and modify previous complex command entries.

Displaying System Information with show Commands

The **show** commands display information about the system and its configuration. The following sections describe some common **show** commands and provide techniques to manage the output from those commands:

- [Common show Commands, page 4-6](#)
- [Browsing Display Output when the --More-- Prompt Appears, page 4-6](#)
- [Halting the Display of Screen Output, page 4-7](#)
- [Redirecting Output to a File, page 4-7](#)
- [Narrowing Output from Large Configurations, page 4-8](#)
- [Filtering show Command Output, page 4-9](#)

Common show Commands

Some of the most common **show** commands are described in [Table 4-2](#).

Table 4-2 Common show Commands in Cisco IOS XR Software

Command	Description	Command Mode
show version	Displays system information.	EXEC mode
show configuration	Displays the uncommitted configuration changes made during a configuration session.	EXEC or any configuration mode
show running-config [<i>command</i>]	Displays the current running configuration.	EXEC or any configuration mode
show tech-support	Collects a large amount of system information for troubleshooting. You can provide this output to technical support representatives when reporting a problem.	EXEC mode
show platform	Displays information about the router.	EXEC mode
show environment [all fans leds power-supply table temperatures voltages I]	Displays hardware information for the system, including fans, LEDs, power supply voltage and current, and temperatures.	EXEC mode

For more information on the use of these commands, see the “[Related Documents](#)” section on page xv.

Browsing Display Output when the --More-- Prompt Appears

When command output requires more than one screen, such as for the **?**, **show**, or **more** command, the output is presented one screen at a time, and a **--More--** prompt is displayed at the bottom of the screen.

To display additional command output, do one of the following:

- Press **Return** to display the next line.
- Press the space bar to display the next screen of output.

The following example shows one screen of data and the **--More--** prompt:

```
RP/0/RP0/CPU0:router# show ?
aaa                Show AAA configuration and operational data
adjacency          Adjacency information
aliases            Display alias commands
alphadisplay       Shows the message being displayed on the alpha display
aps                SONENT APS information
arm                IP ARM information
arp                ARP table
as-path-access-list List AS path access lists
asic-errors        ASIC error information
atc                Attractor Cache related
auto-rp            Auto-RP Commands
bgp                BGP show commands
buffer-manager     Show all buffer manager memory related information
bundle             Show hardware related information for Bundles.
calendar           Display the system calendar
```

```

cdp                CDP information
cef                Cisco Express Forwarding
cetftp            HFR control plane ethernet TFTP server
checkpoint        Show checkpoint services
cinetd            cinetd daemon
clns              Display CLNS related information
clock             Display the system clock
commit            Show commit information
--More--

```



Tips

If you do not see the `--More--` prompt, try entering a value for the screen length with the **terminal length** command in EXEC mode. Command output is not paused if the **length** value is set to zero. The following example shows how to set the terminal length:

```
RP/0/RP1/CPU0:router# terminal length 20
```

For information on searching or filtering CLI output, see the [“Filtering show Command Output”](#) section on page 4-9.

Halting the Display of Screen Output

To interrupt screen output and terminate a display, press Ctrl-C, as shown in the following example:

```
RP/0/RP0/CPU0:router# show running-config
<Ctrl-C>
```

Redirecting Output to a File

By default, CLI command output is displayed on screen. CLI command output can be redirected to a user-specified file by entering a filename and location after the **show** command syntax. The following command syntax is used to redirect output to a file:

```
show command | file filename
```

This feature enables you to save any **show** command output in a file for further analysis and reference. When you choose to redirect command output, consider the following guidelines:

- If the full path of the file is not specified, the default directory for your account is used. You should always save your target configuration files to this location.
- If the saved output is to be used as a configuration file, the filename should end with the `.cfg` suffix for easy identification. This suffix is not required, but can help locate target configuration files.

Example: `myconfig.cfg`

In the following example, a target configuration file is saved to the default user directory:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# show configure | file disk0:myconfig.cfg
RP/0/RP0/CPU0:router(config)# abort
RP/0/RP0/CPU0:router#
```

Narrowing Output from Large Configurations

Displaying a large running configuration can produce thousands of lines of output. To limit the output of a **show** command to only the items you want to view, enter an additional argument at the end of the **show running-config** command. You can also use a wildcard to display all instances for a feature.

Limiting show Command Output to a Specific Feature or Interface

Entering keywords and arguments in the **show** command limits the **show** output to a specific feature or interface.

In the following example, only information about the IP route configuration is displayed:

```
RP/0/RP0/CPU0:router# show running-config route ipv4

route ipv4 0.0.0.0/0 10.21.0.1
route ipv4 0.0.0.0/0 pos0/1/0/1 10.21.0.1
```

In the following example, the configuration for a specific interface is displayed:

```
RP/0/RP0/CPU0:router# show running-config interface POS 0/1/0/1

interface pos0/1/0/1
ipv4 address 10.21.54.31 255.255.0.0
ip proxy-arp disable!
```

Using Wildcards to Display All Instances of an Interface

To display the configuration for all instances, enter the asterisk (*) wildcard character.



Note

See the [“Using Wildcards to Identify Interfaces in show Commands”](#) section on page 4-11 for more information.

In the following example, a configuration for all Packet-over-SONET (PoS) interfaces is displayed:

```
RP/0/RP1/CPU0:router# show running-config interface pos *

interface POS0/1/0/0
ipv4 address 10.2.3.4 255.255.255.0
pos
  crc 32
  !
  shutdown
  keepalive disable
  !
interface POS0/1/0/1
ipv4 address 10.2.3.5 255.255.255.0
pos
  crc 32
  !
  shutdown
  keepalive disable
  !
```

```

interface POS0/1/0/2
  ipv4 address 10.2.3.6 255.255.255.0
  pos
  crc 32
  !
  shutdown
  keepalive disable
  !
interface POS0/1/0/3
  ipv4 address 10.2.3.7 255.255.255.0
  pos
  crc 32
  !
  shutdown
  keepalive disable
  !

--More--

```

Filtering show Command Output

Output from the **show** commands can generate a large amount of data. To display only a subset of information, enter the “pipe” character (**|**) followed by a keyword (**begin**, **include**, or **exclude**) and a regular expression. [Table 4-3](#) shows the filtering options for the **show** command.

Table 4-3 *show Command Filter Options*

Command	Description
show command begin regular-expression	Begins unfiltered output of the show command with the first line that contains the regular expression.
show command exclude regular-expression	Displays output lines that do not contain the regular expression.
show command include regular-expression	Displays output lines that contain the regular expression.
show command file disk0:myconfigfile	Writes the output lines that contain the regular expression to the specified file on the specified device.

In the following example, the **show interface** command includes only lines in which the expression “protocol” appears:

```
RP/0/RP0/CPU0:router# show interface | include protocol
```

```

Null0 is up, line protocol is up
0 drops for unrecognized upper-level protocol
POS0/2/0/0 is administratively down, line protocol is administratively down
0 drops for unrecognized upper-level protocol
POS0/2/0/1 is administratively down, line protocol is administratively down
0 drops for unrecognized upper-level protocol
POS0/2/0/2 is administratively down, line protocol is administratively down
0 drops for unrecognized upper-level protocol
POS0/2/0/3 is administratively down, line protocol is administratively down
0 drops for unrecognized upper-level protocol
MgmtEthernet0/RP0/CPU0/0 is administratively down, line protocol is administratively
down
MgmtEthernet0/RP0/CPU0/0 is administratively down, line protocol is administratively
down
0 drops for unrecognized upper-level protocol

```

**Note**

Filtering is available for submodes, complete commands, and anywhere that `<cr>` appears in the “?” output.

Adding a Filter at the --More-- Prompt

You can specify a filter at the `--More--` prompt of a **show** command output by entering a forward slash (/) followed by a regular expression. The filter remains active until the command output finishes or is interrupted (using Ctrl-Z or Ctrl-C). The following rules apply to this technique:

- If a filter is specified at the original command or previous `--More--` prompt, a second filter cannot be applied.
- The use of the keyword **begin** does not constitute a filter.
- The minus sign (-) preceding a regular expression displays output lines that do not contain the regular expression.
- The plus sign (+) preceding a regular expression displays output lines that contain the regular expression.

In the following example, the user adds a filter at the `--More--` prompt to show only the lines in the remaining output that contain the regular expression “ip.”

```
RP/0/RP0/CPU0:router# show configuration running | begin line

Building configuration...
line console
  exec-timeout 120 120
!
logging trap
--More--
/ip
filtering...
ip route 0.0.0.0 255.255.0.0 pos0/2/0/0
interface pos0/2/0/0
  ip address 172.19.73.215 255.255.0.0
end
```

**Tip**

On most systems, Ctrl-Z can be entered at any time to interrupt the output and return to EXEC mode.

For more information, see [Appendix D, “Understanding Regular Expressions, Special Characters, and Patterns.”](#)

Wildcards, Templates, and Aliases

This section contains the following topics:

- [Using Wildcards to Identify Interfaces in show Commands, page 4-11](#)
- [Creating Configuration Templates, page 4-12](#)
- [Aliases, page 4-16](#)
- [Keystrokes Used as Command Aliases, page 4-17](#)

Using Wildcards to Identify Interfaces in show Commands

Wildcards (*) identify a group of interfaces in **show** commands. [Table 4-4](#) provides examples of wildcard usage to identify a group of interfaces.

Table 4-4 Examples of Wildcard Usage

Wildcard Syntax	Description
*	Specifies all interfaces
pos*	Specifies all POS interfaces in the system
pos0/1/*	Specifies all POS interfaces in rack 0, slot 1
pos0/3/4.*	Specifies all subinterfaces for POS0/3/4



Note

The wildcard (*) must be the last character in the interface name.

Example

In the following example, the configuration for all POS interfaces in rack 0, slot 1 is displayed:

```
RP/0/RP1/CPU0:router# show running-config interface pos0/1/*

interface POS0/1/0/0
  ipv4 address 10.2.3.4 255.255.255.0
  pos
  crc 32
  !
  keepalive disable
interface POS0/1/0/1
  ipv4 address 10.2.3.5 255.255.255.0
  pos
  crc 32
  !
  keepalive disable
interface POS0/1/0/2
  ipv4 address 10.2.3.6 255.255.255.0
  pos
  crc 32
  !
  keepalive disable
interface POS0/1/0/3
  ipv4 address 10.2.3.7 255.255.255.0
  pos
  crc 32
  !
  keepalive disable

--More--
```

In the following example, the state of all POS interfaces is displayed:

```
RP/0/RP1/CPU0:router# show interfaces pos* brief
```

Intf Name	Intf State	LineP State	Encap Type	MTU (byte)	BW (Kbps)
PO0/1/0/0	up	up	HDLC	4474	2488320
PO0/1/0/1	up	up	HDLC	4474	2488320
PO0/1/0/2	up	up	HDLC	4474	2488320
PO0/1/0/3	up	up	HDLC	4474	2488320
PO0/1/0/4	up	up	HDLC	4474	2488320
PO0/1/0/5	up	up	HDLC	4474	2488320
PO0/1/0/6	up	up	HDLC	4474	2488320
PO0/1/0/7	up	up	HDLC	4474	2488320
PO0/1/0/8	up	up	HDLC	4474	2488320
PO0/1/0/9	up	up	HDLC	4474	2488320
PO0/1/0/10	up	up	HDLC	4474	2488320
PO0/1/0/11	up	up	HDLC	4474	2488320
PO0/1/0/12	up	up	HDLC	4474	2488320
PO0/1/0/13	up	up	HDLC	4474	2488320
PO0/1/0/14	up	up	HDLC	4474	2488320
PO0/1/0/15	up	up	HDLC	4474	2488320

Creating Configuration Templates

Configuration templates allow you to create a name that represents a group of configuration commands. After a template is defined, it can be applied to interfaces by you or other users. As networks scale to large numbers of nodes and ports, the ability to configure multiple ports quickly using templates can greatly reduce the time it takes to configure interfaces.

The two primary steps in working with templates are creating templates and applying templates. The following procedure describes how to create a configuration template.

SUMMARY STEPS

1. **configure**
2. **template** *template-name* [*parameter*] [*config-commands*]
3. Enter the template definitions.
4. **end-template**
5. **show running-config template** *template-name*
6. Apply the template.
 - a. **configure**
 - b. **apply-template** *template-name* [*parameter*]
 - c. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: Router# configure	Enters global configuration mode.
Step 2	template <i>template-name</i> [<i>parameter</i>] [<i>config-commands</i>] Example: RP/0/RP0/CPU0:router(config)# template tmplt_1	Enters template configuration mode and creates a template. <ul style="list-style-type: none"> • <i>template-name</i>: Unique name for the template to be applied to the running configuration. • <i>parameter</i>: (Optional) Actual values of the variables specified in the template definition. Up to five parameters can be specified within parentheses. Templates can be created with or without parameters. • <i>config-commands</i>: (Optional) Global configuration commands to be added to the template definition. Any name in a command (such as the server name, group name, and so on) can be parameterized. This means that those parameters can be used in the template commands (starting with \$) and replaced with real arguments when applied. • To remove the template, use the no form of this command. Type the template command in global configuration mode.
Step 3	Enter the template definitions. Example: RP/0/RP0/CPU0:router(config-TPL)# interface pos0/2/0/4 RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.0.0.1 RP/0/RP0/CPU0:router(config-if)# exit	—
Step 4	end-template Example: RP/0/RP0/CPU0:router(config-TPL)# end-template	Ends the template definition session and exits template configuration mode. <ul style="list-style-type: none"> • When you end the template session, you are returned to global configuration mode.

	Command or Action	Purpose
Step 5	<p>show running-config template <i>template-name</i></p> <p>Example: RP/0/RP0/CPU0:router# show running-config template tmplt_1</p>	Displays the details of the template.
Step 6	<p>Apply the template:</p> <ol style="list-style-type: none"> configure apply-template <i>template-name</i> [<i>parameter</i>] show running-config <p>Example: RP/0/RP0/CPU0:router# configure RP/0/RP0/CPU0:router(config)# apply-template tmplt_1 RP/0/RP0/CPU0:router(config)# commit RP/0/RP0/CPU0:router(config)# show running-config ... template tmplt_1 interface POS0/2/0/4 ipv4 address 10.0.0.1 255.255.255.0 ! end-template .. interface POS0/2/0/4 ipv4 address 10.0.0.1 255.255.255.0 ! RP/0/RP0/CPU0:router(config)# exit</p>	<p>Applies a defined template and its parameters to the running configuration of the system.</p> <ul style="list-style-type: none"> Type this command in global configuration mode. Only one template can be applied at a time. If the same template is applied multiple times, the most recent application overwrites the previous ones. Provide the exact number of parameters for the template. Templates are applied as a “best effort” operation; only valid changes are committed. If any command in the template fails, that command is discarded. To remove the template, use the no form of this command.

Applying Configuration Templates

To apply a template, enter the **apply-template** *template-name* [*parameter*] command in global configuration mode.

The following command applies the template *tmplt_1*:

```
RP/0/RP0/CPU0:router(config)# apply-template tmplt_1
```

The following **apply** command applies the template *bar* with two arguments:

```
RP/0/RP0/CPU0:router(config)# apply-template bar (mibtwister %wd_default_mem.tcl)
```

To display the results of the previously applied templates, enter the **show running-config** command:

```
RP/0/RP0/CPU0:router# show running-config

hostname mibtwister #new hostname set by apply bar
template bar (abc cde )
hostname $abc
fault manager policy %wd_default_cpu.tcl system
fault manager policy $cde system
logging trap
logging trap informational
logging console debugging
logging history size 1
logging history warnings
logging monitor debugging
logging buffered 16384
end-template
template tmplt_1
interface preconfigure pos0/1/0/2
ipv4 address 10.2.3.4 255.255.255.0
!
end-template
.
.
.
fault manager policy %wd_default_cpu.tcl system
fault manager policy %wd_default_mem.tcl system # substituted
logging trap
logging trap informational
logging console debugging
logging history size 1
logging history warnings
logging monitor debugging
logging buffered 16384
.
.
.
interface preconfigure pos0/1/0/2 # set by apply tmplt_1
ipv4 address 10.2.3.4 255.255.255.0
!
.
.
.
```

Examples

In the following example, a simple template is defined. The template contents are then displayed with the **show running-config template** *template-name* command:

```
RP/0/RP0/CPU0:router(config)# template tmplt_1
RP/0/RP0/CPU0:router(config-TPL)# interface pos0/2/0/4
RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.2.3.4 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config-TPL)# end-template
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router# show running-config template tmplt_1
interface pos0/2/0/4
    ipv4 address 10.2.3.4 255.255.255.0      !
end-template

interface pos0/2/0/4
ipv4 address 10.2.3.4 255.255.0.0      !
```

In the following example, a template named *bar* with two parameters is defined:

```
RP/0/RP0/CPU0:router(config)# template bar (abc cde)
RP/0/RP0/CPU0:router(config-TPL)# hostname $abc
RP/0/RP0/CPU0:router(config-TPL)# fault manager policy %wd_default_cpu.tcl system
RP/0/RP0/CPU0:router(config-TPL)# fault manager policy $cde system
RP/0/RP0/CPU0:router(config-TPL)# logging trap
RP/0/RP0/CPU0:router(config-TPL)# logging trap informational
RP/0/RP0/CPU0:router(config-TPL)# logging console debugging
RP/0/RP0/CPU0:router(config-TPL)# logging history size 1
RP/0/RP0/CPU0:router(config-TPL)# logging history warnings
RP/0/RP0/CPU0:router(config-TPL)# logging monitor debugging
RP/0/RP0/CPU0:router(config-TPL)# logging buffered 16384
RP/0/RP0/CPU0:router(config-TPL)# end-template
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router#show running-config template bar
```

```
template bar (abc cde)
hostname $abc
fault manager policy %wd_default_cpu.tcl system
fault manager policy $cde system
logging trap
logging trap informational
logging console debugging
logging history size 1
logging history warnings
logging monitor debugging
logging buffered 16384
end-template
```



Note

Configuration commands in the template body can have variables beginning with the dollar sign (\$). When the template is applied, variables beginning with “\$” can be substituted by real arguments.

Aliases

Cisco IOS XR software lets you define command line aliases for any physical or logical entity in a router. After you define the alias, it can be used in the CLI to reference the real entity.

To create a command alias, enter the following command in global configuration mode:

```
alias alias-name command-syntax
```

Table 4-5 defines the **alias** command syntax.

Table 4-5 *alias Command Syntax*

Syntax	Specifies that the Alias Is Created for
<i>alias-name</i>	Name of the command alias. An alias name can be a single word or multiple words joined by a dash (-).
<i>command-syntax</i>	Original command syntax. Valid abbreviations of the original command syntax can be entered for the <i>command-syntax</i> argument.

To delete all aliases in a command mode or a specific alias, enter the **no** form of the **alias** command.

In the following example, an alias named *my-cookie* is created for the Management Ethernet interface, and then the new alias is specified to enter interface configuration mode:

```
RP/0/0/CPU0:router(config)#alias my-cookie mgmtEth 0/0/CPU0/0
```

```
RP/0/0/CPU0:router(config) #interface my-cookie <cr>
RP/0/0/CPU0:router(config) #interface mgmtEth 0/0/CPU0/0
RP/0/0/CPU0:router(config-if) #
```

After you enter a command with an alias, the router displays the command you entered with the alias value so that you can verify that alias value.

Keystrokes Used as Command Aliases

The system can be configured to recognize particular keystrokes (key combination or sequence) as command aliases. In other words, a keystroke can be set as a shortcut for executing a command. To enable the system to interpret a keystroke as a command, use the Ctrl-V or Esc, Q key combinations before entering the command sequence.

Command History

The Cisco IOS XR software lets you display a history of the most recently entered and deleted commands. You can also redisplay the command line while a console message is being shown. The following sections describe the command history functionality:

- [Recalling Previously Entered Commands, page 4-17](#)
- [Recalling Deleted Entries, page 4-18](#)
- [Redisplaying the Command Line, page 4-18](#)

**Note**

To roll back to a previously committed configuration, see the [“Managing Configuration History and Rollback” section on page 3-3](#).

Recalling Previously Entered Commands

The Cisco IOS XR software records the ten most recent commands issued from the command line in its history buffer. This feature is particularly useful for recalling long or complex commands or entries, including access lists.

To recall commands from the history buffer, use one of the commands or key combinations listed in [Table 4-6](#).

Table 4-6 Command History

Command or Key Combination	Purpose
Ctrl-P or the up arrow key	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	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.

**Note**

Use the **terminal history** command to set the number of command line entries the system holds for recall during a terminal session.

Recalling Deleted Entries

The Cisco IOS XR CLI also stores deleted commands or keywords in a history buffer. The buffer stores the last ten items that have been deleted using Ctrl-K, Ctrl-U, or Ctrl-X. Individual characters deleted using Backspace or Ctrl-D are not stored.

[Table 4-7](#) identifies the keystroke combinations used to recall deleted entries to the command line.

Table 4-7 Keystroke Combinations to Recall Deleted Entries

Command or Key Combination	Recalls the
Ctrl-Y	Most recent entry in the buffer (press the keys simultaneously).
Esc, Y	Previous entry in the history buffer (press the keys sequentially).

**Note**

The Esc, Y key sequence does not function unless the Ctrl-Y key combination is pressed first. If the Esc, Y is pressed more than ten times, the history cycles back to the most recent entry in the buffer.

Redisplaying the Command Line

If the system sends a message to the screen while a command is being entered, the current command line entry can be redisplayed using the Ctrl-L or Ctrl-R key combination.

Key Combinations

The following sections provide information on key combinations:

- [Key Combinations to Move the Cursor](#), page 4-19
- [Keystrokes to Control Capitalization](#), page 4-19
- [Keystrokes to Delete CLI Entries](#), page 4-20

Key Combinations to Move the Cursor

Table 4-8 shows the key combinations or sequences you can use to move the cursor around on the command line to make corrections or changes. When you use cursor control keys, consider the following guidelines:

- 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.

Table 4-8 Key Combinations Used to Move the Cursor

Keystrokes	Function	Moves the Cursor
Left arrow or Ctrl-B	Back character	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	One character to the right.
Esc, B	Back word	Back one word.
Esc, F	Forward word	Forward one word.
Ctrl-A	Beginning of line	To the beginning of the line.
Ctrl-E	End of line	To the end of the command line.

Keystrokes to Control Capitalization

Letters can be capitalized or uncapitalized using simple key sequences. Table 4-9 describes the keystroke combinations used to control capitalization.



Note

Cisco IOS XR commands are generally case insensitive and typically all in lowercase.

Table 4-9 Keystrokes Used to Control Capitalization

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.

Keystrokes to Delete CLI Entries

Table 4-10 describes the keystrokes used to delete command line entries.

Table 4-10 *Keystrokes for Deleting Entries*

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

Transposing Mistyped Characters

To transpose mistyped characters, use the Ctrl-T key combination.