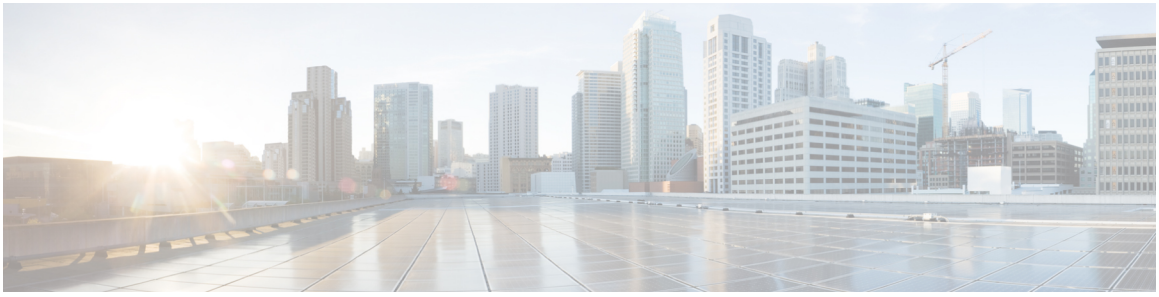
System Management Configuration Guide, Cisco IOS XE 17.x

First Published: 2023-08-21

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883



CONTENTS

Preface

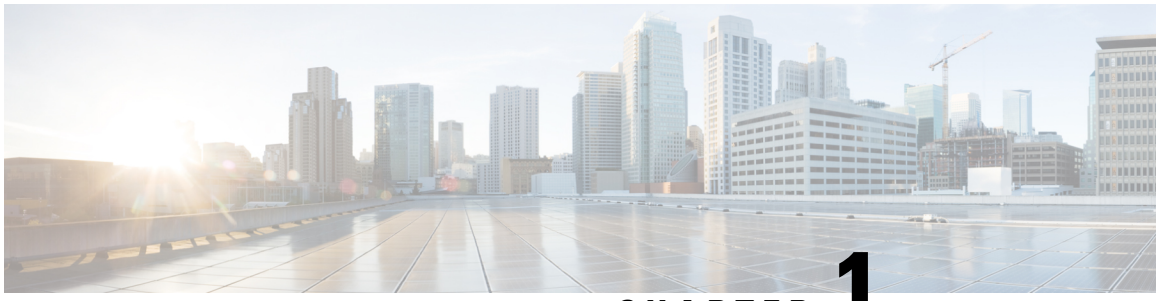
This preface describes the audience, organization, and conventions of this document. It also provides information on how to obtain other documentation.

This preface includes the following sections:

Reference Preface Map here

PART I

Basic System Management



CHAPTER 1

Performing Basic System Management

This module describes the basic tasks that you can perform to manage the general system features of the Cisco IOS software--those features that are generally not specific to a particular protocol.

Information About Performing Basic System Management

System Name

The system name, also called the hostname, is used to uniquely identify the system in your network. The system name is displayed at the CLI prompt. If no name is configured, the system default name is Router.

Command Aliases

Command aliases allow you to configure alternative syntax for commands. You may want to create aliases for commonly used or complex commands. For example, you could assign the alias **save config** to the **copy running-config startup-config** command to reduce the amount of typing you have to perform, or if your users might find the **save config** command easier to remember. Use word substitutions or abbreviations to tailor the command syntax for you and your user community.

Remember that any aliases you configure will be effective only on your system, and that the original command syntax will appear in the configuration file.

Minor Services

Minor services are small servers that run on your routing device and are useful for basic system testing and for providing basic network functions. Minor services are useful for testing connections from another host on the network.

Cisco small servers are conceptually equivalent to daemons.

Small servers provided by Cisco IOS software-based devices include TCP, UDP, HTTP, Bootstrap Protocol (BOOTP), and Finger. For information about the HTTP server, see the "Using the Cisco Web Browser User Interface" chapter in the Cisco IOS Configuration Fundamentals Configuration Guide.

The TCP small server provides the following minor services:

- Chargen--Generates a stream of ASCII data. To test this service, issue the **telnet** *a.b.c.d* **chargen**command from a remote host.
- Daytime--Returns the system date and time if you have configured Network Time Protocol (NTP) or set the date and time manually. To test this service, issue the **telnet** *a.b.c.d* **daytime**command from a remote host.
- Discard--Discards whatever you type. To test this service, issue the **telnet** *a.b.c.d* **discard**command from a remote host.
- Echo--Echoes back whatever you type. To test this service, issue the **telnet** *a.b.c.d* **echo**command from a remote host.

The UDP small server provides the following minor services:

- Chargen--Discards the datagram that you send and responds with a 72-character string of ASCII characters terminated with a CR+LF (carriage return and line feed).
- Discard--Discards the datagram you send.
- Echo--Echoes the payload of the datagram that you send.

Minor services are disabled by default.

Caution Enabling minor services creates the potential for certain types of denial-of-service (DoS) attacks, such as the UDP diagnostic port attack. Therefore, any network device that has UDP, TCP, BOOTP, or Finger services should be protected by a firewall or have the minor services disabled. For information on preventing UDP diagnostic port attacks, see the white paper titled Defining Strategies to Protect Against UDP Diagnostic Port Denial of Service Attacks a vailable on Cisco.com.

BOOTP Server

You can enable or disable an async line Bootstrap Protocol (BOOTP) service on your routing device. This small server is enabled by default. Due to security considerations, this service should be disabled if you are not using it.

Because DHCP is based on the BOOTP, both of these service share the well-known UDP server port 67 (per the Internet standards and RFCs). For more information about DHCP configuration in the Cisco IOS software, see the Cisco IOS IP Addressing Configuration Guide. For more information about BOOTP, see RFC 951. Interoperation between BOOTP and DHCP is defined in RFC 1534. DHCP is defined in RFC 2131.

Finger Protocol

The Finger protocol allows users throughout the network to get a list of the users currently using a particular routing device. The information displayed includes the processes running on the system, the line number, connection name, idle time, and terminal location. This information is provided through the Cisco IOS software **show users** EXEC command.

Hidden Telnet Addresses

You can hide addresses while attempting to establish a Telnet session. The hide feature suppresses the display of the address and continues to display all other messages that normally would be displayed during a connection attempt, such as detailed error messages if the connection fails.

EXEC Startup Delay

To delay the startup of the EXEC process on noisy lines until the line has been idle for 3 seconds, use the **service exec-wait** command in global configuration mode.

This command is useful on noisy modem lines or when a modem attached to the line is configured to ignore Microcom Networking Protocol (MNP) or V.42 negotiations, and when MNP or V.42 modems are dialing in. In these cases, noise or MNP/V.42 packets might be interpreted as usernames and passwords, causing authentication failure before the user can type a username or password. This command is not useful on nonmodem lines or lines without some kind of login configured.

Idle Telnet Connections

Normally, data sent to noncurrent Telnet connections is accepted and discarded. When the **service telnet-zero-idle** command is enabled and a session is suspended (that is, some other connection is made active), the TCP window is set to zero. This action prevents the remote host from sending any more data until the connection is resumed. Use this command when all messages sent by the host must be seen by the users and the users are likely to use multiple sessions. Do not use this command if your host will eventually time out and log out a TCP user whose window is zero.

Interval for Load Data

You can change the period of time over which a set of data is used for computing load statistics. Decisions, such as dial backup, depend on these statistics. If you decrease the load interval, the average statistics are computed over a shorter period of time and are more responsive to bursts of traffic.

Number of TCP Transactions

When you are using a standard TCP implementation to send keystrokes between machines, TCP tends to send one packet for each keystroke typed, which can use up the bandwidth and contribute to the congestion on larger networks.

John Nagle's algorithm (RFC 896) helps alleviate the small-packet problem in TCP. The first character typed after the connection establishment is sent in a single packet, but TCP holds any additional characters that are typed until the receiver acknowledges the previous packet. Then the second, larger packet is sent, and the additional typed characters are saved until the acknowledgment comes back. The effect is to accumulate characters into larger chunks, and pace their transmission to the network at a rate matching the round-trip time of the given connection. This method is usually preferable for all TCP-based traffic.

By default, the Nagle algorithm is not enabled.

Switching and Scheduling Priorities

The normal operation of the network server allows the switching operations to use as much of the central processor as required. If the network is running unusually heavy loads that do not allow the processor the time to handle the routing protocols, you may need to give priority to the system process scheduler.

System Buffer Size

You can adjust the initial buffer pool settings and limits at which temporary buffers are created and destroyed.

During normal system operation, there are two sets of buffer pools: public and interface. They behave as follows:

- The buffers in the public pools grow and shrink based upon demand. Some public pools are temporary and are created and destroyed as needed. Other public pools are permanently allocated and cannot be destroyed. Public buffer pools are labeled as small, middle, big, very big, large, and huge.
- Interface pools are static--that is, they are all permanent. One interface pool exists for each interface. For example, a Cisco 4000 1E 4T configuration has one Ethernet buffer pool and four serial buffer pools.

The server has one pool of queueing elements and six public pools of packet buffers of different sizes. For each pool, the server keeps count of the number of outstanding buffers, the number of buffers in the free list, and the maximum number of buffers allowed in the free list.

How to Perform Basic System Management

Setting Basic System Parameters

To set basic system parameters perform the following steps. You can perform these steps based on the customization requirements of your system.

SUMMARY STEPS

1. **hostname** *name*
2. **prompt** *string*
3. **alias** *mode alias-name alias-command-line*
4. **service tcp-small-servers**
5. **service udp-small-servers**
6. **no ip bootp server**
7. **ip finger**
8. **ip finger rfc-compliant**
9. **service hide-telnet-address**
10. **line** *line-number*
11. exit
12. **exit**
13. **busy-message** *hostname message*
14. **service exec-wait**

15. **service telnet-zero-idle**
16. **load-interval** *seconds*
17. **service nagle**
18. **scheduler interval** *milliseconds*
19. **scheduler allocate** [*network-microseconds process-microseconds*]
20. **scheduler process-watchdog** {**hang** | **normal** | **reload** | **terminate**}
21. **buffers** {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number*
22. exit
23. **show aliases** [*mode*]
24. **show buffers**

DETAILED STEPS

Step 1 **hostname** *name*

Use the **hostname** *name* command to perform the basic system management task of assigning a name for your device.

Example:

```
Router(config)# hostname host1
```

Step 2 **prompt** *string*

or

no service prompt config

By default, the CLI prompt consists of the system name followed by an angle bracket (>) for user EXEC mode or a pound sign (#) for privileged EXEC mode. Use the the **prompt** *string* or the **no service prompt config** command to customize the CLI prompt for your system.

Example:

```
Router(config)# prompt Router123
```

or

Example:

```
Router(config)# no service prompt config
```

Step 3 **alias** *mode alias-name alias-command-line*

Use the **alias** *mode alias-name alias-command-line* command to create a command alias.

Example:

```
Router(config)# alias exec save config copy running-config startup-config
```

Step 4 **service tcp-small-servers**

Use the **service tcp-small-servers** command to enable minor TCP services such as chargen, daytime, discard, and echo.

Note The **no** form of the **service tcp-small-servers** command will appear in the configuration file when these basic services are disabled.

Example:

```
Router(config)# service tcp-small-servers
```

Step 5 **service udp-small-servers**

Use the **service udp-small-servers** command to enable minor UDP services such as chargen, daytime, discard, and echo.

Note The **no** form of the **service udp-small-servers** command will appear in the configuration file when these basic services are disabled.

Example:

```
Router(config)# service udp-small-servers
```

Step 6 **no ip bootp server**

Use the **no ip bootp server** command to disable the BOOTP server on your platform.

Example:

```
Router(config)# no ip bootp server
```

Step 7 **ip finger**

Use the **ip finger** command to enable a Cisco device to respond to Finger (port 79) requests. When the **ip finger** command is configured, the router will respond to a **telnet** *a.b.c.d* **finger** command from a remote host by immediately displaying the output of the **show users**command and then closing the connection.

Example:

```
Router(config)# ip finger
```

Step 8 **ip finger rfc-compliant**

Use the **ip finger rfc-compliant** command to configure the finger protocol to be compliant with RFC 1288. The **ip finger rfc-compliant** command should not be configured for devices with more than 20 simultaneous users. When the **ip finger rfc-compliant**command is configured, the router will wait for input before displaying any information. The remote user can then press the Return key to display the output of the **show users** command, or enter **/W** to display the output of the **show users wide** command. After this information is displayed, the connection is closed.

Example:

```
Router(config)# ip finger rfc-compliant
```

Step 9 **service hide-telnet-address**

Use the **service hide-telnet-address** command to configure the router to suppress Telnet addresses.

Example:

```
Router(config)# service hide-telnet-address
```

Step 10 **line** *line-number*

Use the line command to enter line configuration mode.

Example:

```
Router(config)# line 1
```

Step 11 exit

Use the **exit** command to exit line configuration mode and return to global configuration mode.

Example:

```
Router(config-line)# exit
```

Step 12 **exit**

Use the **exit** command to exit line configuration mode and return to global configuration mode.

Example:

```
Router(config-line)# exit
```

Step 13 **busy-message** *hostname message*

Use the **busy-message** command with the **service hide-telnet-address**command to customize the information displayed during Telnet connection attempts. If the connection attempt fails, the router suppresses the address and displays the message specified with the **busy-message** command.

Example:

```
Router(config)# busy-message host1 message1
```

Step 14 **service exec-wait**

Use the **service exec-wait**command to delay the startup of the EXEC process on noisy lines until the line has been idle for 3 seconds.

Example:

```
Router(config)# service exec-wait
```

Step 15 **service telnet-zero-idle**

Use the **service telnet-zero-idle** command to configure the Cisco IOS software to set the TCP window to zero (0) when the Telnet connection is idle.

Example:

```
Router(config)# service telnet-zero-idle
```

Step 16 **load-interval** *seconds*

Use the **load-interval** *seconds* command to change the length of time for which a set of data is used to compute load statistics.

Example:

```
Router(config)# load-interval 100
```

Step 17 **service nagle**

Use the **service nagle** command to enable the Nagle algorithm and thereby reduce the number of TCP transactions.

Example:

```
Router(config)# load-interval 100
```

Step 18 **scheduler interval** *milliseconds*

Use the **scheduler interval** *milliseconds* command to define the maximum amount of time that can elapse without running the lowest-priority system processes.

Example:

```
Router(config)# scheduler interval 100
```

Step 19 **scheduler allocate** [*network-microseconds process-microseconds*]

Use the **scheduler allocate** command to change the amount of time that the CPU spends on fast-switching and process-level operations on the Cisco 7200 series and Cisco 7500 series routers.

Caution Cisco recommends that you do not change the default values of the **scheduler allocate** command.

Example:

```
Router(config)# scheduler allocate 5000 200
```

Step 20 **scheduler process-watchdog** {**hang** | **normal** | **reload** | **terminate**}

Use the **scheduler process-watchdog** {**hang** | **normal** | **reload** | **terminate**} command to configure the characteristics for a looping process.

Example:

```
Router(config)# scheduler process-watchdog hang
```

Step 21 **buffers** {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number*

Use the **buffers** {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number* command to adjust the system buffer size.

Example:

```
Router(config)# buffers small permanent 10
```

Caution Cisco does not recommend that you adjust these parameters. Improper settings can adversely impact the system performance.

Step 22 exit

Use the **exit** command to exit global configuration mode and return to privileged EXEC mode.

Example:

```
Router(config)# exit
```

Step 23 **show aliases** [*mode*]

Use the **show aliases** [*mode*] command to display a list of command aliases currently configured on your system, and the original command syntax for those aliases.

Example:

```
Router# show aliases exec
```

Step 24 **show buffers**

Use the **show buffers**command to display buffer information. For more information about this command, see the Cisco IOS Configuration Fundamentals Command Reference.

Example:

```
Router# show buffers
Buffer elements:
     1119 in free list (1119 max allowed)
     641606 hits, 0 misses, 619 created
Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
     48 in free list (20 min, 150 max allowed)
     2976557 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Middle buffers, 600 bytes (total 25, permanent 25, peak 37 @ 2w0d):
     25 in free list (10 min, 150 max allowed)
     445110 hits, 4 misses, 12 trims, 12 created
     0 failures (0 no memory)
Big buffers, 1536 bytes (total 50, permanent 50):
     50 in free list (5 min, 150 max allowed)
     58004 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
VeryBig buffers, 4520 bytes (total 10, permanent 10):
     10 in free list (0 min, 100 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Large buffers, 5024 bytes (total 0, permanent 0):
     0 in free list (0 min, 10 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Huge buffers, 18024 bytes (total 0, permanent 0):
     0 in free list (0 min, 4 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Interface buffer pools:
Syslog ED Pool buffers, 600 bytes (total 282, permanent 282):
     257 in free list (282 min, 282 max allowed)
     32 hits, 0 misses
IPC buffers, 4096 bytes (total 2, permanent 2):
     1 in free list (1 min, 8 max allowed)
     1 hits, 0 fallbacks, 0 trims, 0 created
     0 failures (0 no memory)
Header pools:
Header buffers, 0 bytes (total 511, permanent 256, peak 511 @ 2w0d):
     255 in free list (256 min, 1024 max allowed)
     171 hits, 85 misses, 0 trims, 255 created
     0 failures (0 no memory)
     256 max cache size, 256 in cache
     0 hits in cache, 0 misses in cache
Particle Clones:
     1024 clones, 0 hits, 0 misses
Public particle pools:
F/S buffers, 128 bytes (total 512, permanent 512):
     0 in free list (0 min, 512 max allowed)
```

```
     512 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
     512 max cache size, 512 in cache
     0 hits in cache, 0 misses in cache
Normal buffers, 512 bytes (total 2048, permanent 2048):
     2048 in free list (1024 min, 4096 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Private particle pools:
HQF buffers, 0 bytes (total 2000, permanent 2000):
     2000 in free list (500 min, 2000 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Serial2/0 buffers, 512 bytes (total 256, permanent 256):
     0 in free list (0 min, 256 max allowed)
     256 hits, 0 fallbacks
     256 max cache size, 132 in cache
     124 hits in cache, 0 misses in cache
     10 buffer threshold, 0 threshold transitions
Serial2/1 buffers, 512 bytes (total 256, permanent 256):
     0 in free list (0 min, 256 max allowed)
     256 hits, 0 fallbacks
     256 max cache size, 132 in cache
     124 hits in cache, 0 misses in cache
     10 buffer threshold, 0 threshold transitions
```

Configuration Examples for Performing Basic System Management

There are no configuration examples for the Performing Basic System Management feature.

Additional References

Related Documents

Related Topic	Document Title
Network Management commands	*Cisco IOS Network Management Command Reference*
Cisco IOS fundamental configuration commands	*Cisco IOS Configuration Fundamentals Command Reference*
Cisco IOS fundamental configurations	*Cisco IOS Configuration Fundamentals Configuration Guide*
Preventing UDP diagnostic port attacks	Defining Strategies to Protect Against UDP Diagnostic Port Denial of Service Attacks
DHCP configuration	*Cisco IOS IP Addressing Configuration Guide*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 896	*Congestion Control in IP/TCP Internetworks*
RFC 951	*Algorithms for Synchronizing Network Clocks*
RFC 1288	*The Finger User Information Protocol*
RFC 1534	*Interoperation Between DHCP and BOOTP*
RFC 2131	*Dynamic Host Configuration Protocol*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

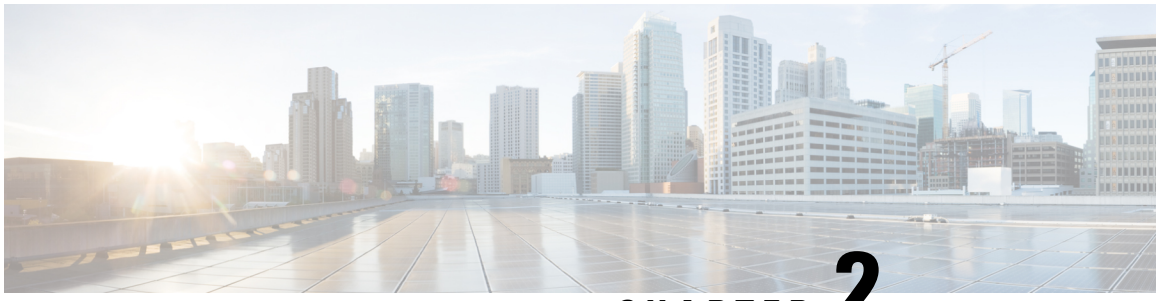
Feature Information for Performing Basic System Management

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for Performing Basic System Management

Feature Name	Releases	Feature Information
Performing Basic System Management		This module describes the basic tasks to manage the general system features of the Cisco IOS software.



CHAPTER 2

Memory Threshold Notifications

The Memory Threshold Notifications feature allows you to reserve memory for critical notifications and to configure a router to issue notifications when available memory falls below a specified threshold.

Information About Memory Threshold Notifications

The Memory Threshold Notifications feature provides two ways to mitigate low-memory conditions on a router: notifications can be sent to indicate that free memory has fallen below a configured threshold, and memory can be reserved to ensure that sufficient memory is available to issue critical notifications. To implement the Memory Threshold Notifications feature, you should understand the following concepts:

Memory Threshold Notifications

The Memory Threshold Notifications feature allows you to reserve memory for critical notifications and to configure a router to issue notifications when available memory falls below a specified threshold.

Memory Reservation

Memory reservation for critical operations ensures that management processes, such as event logging, continue to function even when router memory is exhausted.

How to Define Memory Threshold Notifications

Setting a Low Free Memory Threshold

Perform this task to set a low free memory threshold.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **memory free low-watermark** [**processor** *threshold*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**memory free low-watermark** [**processor** *threshold* **Example:** `Router(config)# memory free low-watermark processor 20000`	Specifies a threshold in kilobytes of free processor memory. To view acceptable values for the memory threshold, enter the following command: • **memory free low-watermark processor ?**

Reserving Memory for Critical Notifications

When a router is overloaded by processes, the amount of available memory might fall to levels insufficient for it to issue critical notifications. Perform this task to reserve a region of memory to be used by the router for the issuing of critical notifications.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **memory reserve critical** *kilobytes*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**memory reserve critical** *kilobytes* **Example:** `Router(config)# memory reserve critical 1000`	Reserves the specified amount of memory in kilobytes so that the router can issue critical notifications. • The amount of memory reserved for critical notifications cannot exceed 25 percent of total available memory.

Configuration Examples for Memory Threshold Notifications

Setting a Low Free Memory Threshold Examples

Threshold for Free Processor Memory

The following example shows how to specify a threshold of 20000 KB of free processor memory before the router issues notifications:

```
Router(config)# memory free low-watermark processor 20000
```

If available free memory falls below the specified threshold, the router sends a notification message like this one:

```
000029: *Aug 12 22:31:19.559: %SYS-4-FREEMEMLOW: Free Memory has dropped below 20000k
Pool: Processor  Free: 66814056  freemem_lwm: 204800000
```

Once available free memory rises to above 5 percent of the threshold, the router sends a notification message like this one:

```
000032: *Aug 12 22:33:29.411: %SYS-5-FREEMEMRECOVER: Free Memory has recovered 20000k
Pool: Processor  Free: 66813960  freemem_lwm: 0
```

Reserving Memory for Critical Notifications Example

The following example shows how to reserve 1000 KB of memory for critical notifications:

```
Router# memory reserved critical 1000
```

Note The amount of memory reserved for critical notifications cannot exceed 25 percent of total available memory.

Additional References

For additional information related to the CPU Thresholding Notification feature, refer to the following references:

Related Documents

Related Topic	Document Title
SNMP traps	*Configuration Fundamentals Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
CISCO-PROCESS-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

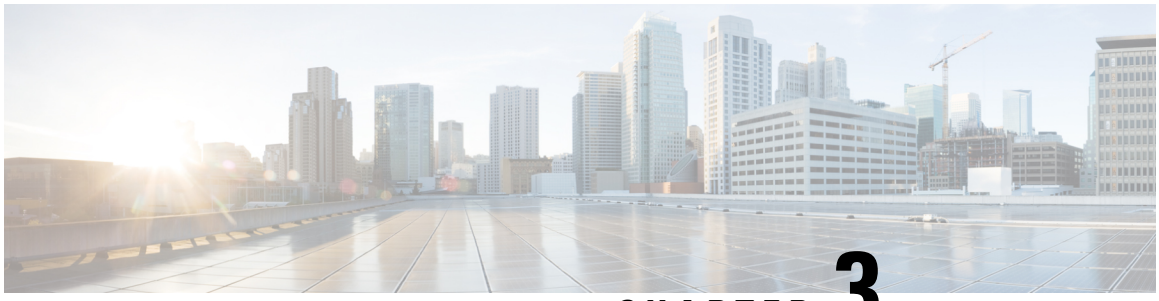
Feature Information for Memory Threshold Notifications

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for Memory Threshold Notifications

Feature Name	Releases	Feature Information
Memory Threshold Notifications	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.



CHAPTER 3

NTPv4 MIB

The NTPv4 MIB feature introduces the Network Time Protocol Version 4 (NTPv4) MIB in Cisco software. It defines data objects that represent the current status of NTP entities. These data objects are accessed using the Simple Network Management Protocol (SNMP) and are used to monitor and manage local NTP entities.

This module describes the NTPv4 MIB.

Information About the NTPv4 MIB

NTPv4 MIB

The Network Time Protocol Version 4 (NTPv4) MIB feature, which is based on RFC 5907, defines data objects that represent the current status of NTP entities. These data objects are accessed using the Simple Network Management Protocol (SNMP) and are used to monitor and manage local NTP entities.

The data objects contain the following information about the NTP entities:

- Connectivity to the upstream NTP servers and to hardware reference clocks.
- Product
- Vendor
- Version

By using the information contained in the data objects, you can detect failures before the overall time synchronization of the network is impacted.

The following object groups that are addressed in RFC 5907 are supported in the NTPv4 MIB:

- ntpAssociation
- ntpEntInfo

- ntpEntStatus

The following object groups that are addressed in RFC 5907 are not supported in the NTPv4 MIB:

- ntpEntControl
- ntpEntNotifObjects

How to Verify the NTPv4 MIB

No special configuration is needed for this feature. This feature is enabled by default.

Verifying NTPv4 MIB

To verify information about the NTPv4 MIB, perform any or all of the following optional commands in any order.

SUMMARY STEPS

1. **show ntp associations** [**detail**]
2. **show ntp status**
3. **show ntp info**
4. **show ntp packets**

DETAILED STEPS

Step 1 **show ntp associations** [**detail**]

Example:

```
Device> show ntp associations detail
```

(Optional) Displays detailed status of NTP associations.

Step 2 **show ntp status**

Example:

```
Device> show ntp status
```

(Optional) Displays the status of NTP.

Step 3 **show ntp info**

Example:

```
Device> show ntp info
```

(Optional) Displays information about NTP entities.

Step 4 **show ntp packets**

Example:

```
Device> show ntp packets
```

(Optional) Displays information about NTP packets.

Configuration Examples for NTPv4 MIB

Example: Verifying the NTP4 MIB

Sample Output for the show ntp associations Command

```
Device> show ntp associations detail

172.31.32.2 configured, ipv4, our_master, sane, valid, stratum 1
ref ID .LOCL., time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
our mode active, peer mode passive, our poll intvl 16, peer poll intvl 16
root delay 0.00 msec, root disp 0.00, reach 377, sync dist 16.05
delay 0.00 msec, offset 0.0000 msec, dispersion 8.01, jitter 0.5 msec
precision 2**7, version 4
assoc ID 1, assoc name 192.0.2.1,
assoc in packets 60, assoc out packets 60, assoc error packets 0
org time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
rec time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
xmt time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =     7.81    8.05    8.29    8.53    8.77    9.01    9.25    9.49
minpoll = 4, maxpoll = 4

192.168.13.33 configured, ipv6, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
our mode client, peer mode unspec, our poll intvl 1024, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 15951.96
delay 0.00 msec, offset 0.0000 msec, dispersion 15937.50,  jitter 1000.45 msec
precision 2**7, version 4
assoc ID 2, assoc name myserver
assoc in packets 0, assoc out packets 0, assoc error packets 0
org time D2351E93.2235F124 (05:56:35.133 IST Tue Oct 4 2011)
rec time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
xmt time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
```

Sample Output for the show ntp status Command

```
Device> show ntp status

Clock is synchronized, stratum 2, reference assoc id 1, reference is 192.0.2.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**7
reference time is D2352258.243DDF14 (06:12:40.141 IST Tue Oct 4 2011)
clock offset is 0.0000 msec, root delay is 0.00 msec, time resolution 1000 (1 msec),
root dispersion is 15.91 msec, peer dispersion is 8.01 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 16, last update was 6 sec ago.
system uptime (00:00:00.000) UTC,
```

```
system time is D2352258.243DDF14 (06:12:40.141 IST Tue Oct 4 2011)
leap time is D2352258.243DDF14 (24:00:00.000 IST Tue Dec 31 2011)
leap direction is 1
```

Sample Output for the show ntp info Command

```
Device> show ntp info

Ntp Software Name: Example
NtP Software Version: ntp-1.1
Ntp Software Vendor: Example
Ntp System Type: Example_System
```

Sample Output for the show ntp packets Command

```
Device> show ntp packets

Ntp In packets: 100
Ntp Out packets: 110
Ntp bad version packets: 4
Ntp protocol error packets: 0
```

Additional References

Related Documents

Related Topic	Document Title
Basic System Management commands	Basic System Management Command Reference
Basic System Management configuration tasks	"Setting Time and Calendar Services" module in the *Basic System Management Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 5907	*Definitions of Managed Objects for Network Time Protocol Version 4 (NTPv4)*

MIBs

MIB	MIBs Link
NTPv4-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

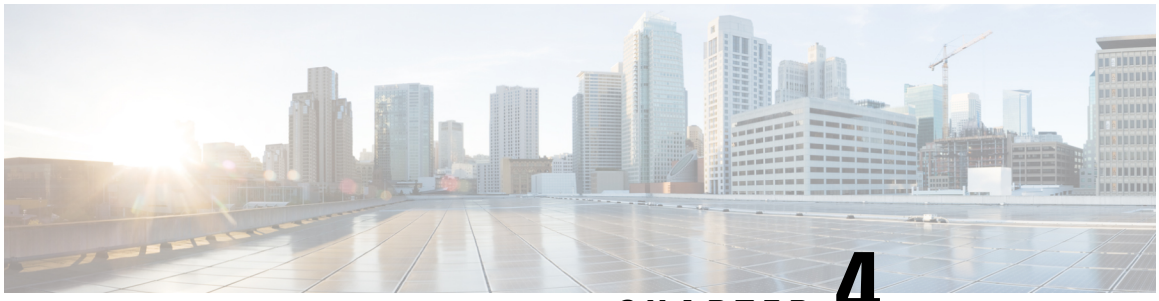
Feature Information for the NTPv4 MIB

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for the NTPv4 MIB

Feature Name	Releases	Feature Information
NTPv4 MIB		The NTPv4 MIB feature introduces the Network Time Protocol Version 4 (NTPv4) MIB in Cisco software. It defines data objects that represent the current status of NTP entities. These data objects are accessed using the Simple Network Management Protocol (SNMP) and are used to monitor and manage local NTP entities.



CHAPTER 4

Network Time Protocol

Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs on User Datagram Protocol (UDP), which in turn runs on IP. NTP Version 3 is documented in RFC 1305.

This module describes how to configure Network Time Protocol on Cisco devices.

Information About Network Time Protocol

Time and Calendar Services

The primary source for time data on your system is the software clock. This clock runs from the moment the system starts up and keeps track of the current date and time. The software clock can be set from a number of sources and in turn can be used to distribute the current time through various mechanisms to other systems. When a device with a hardware clock is initialized or rebooted, the software clock is initially set based on the time in the hardware clock. The software clock can then be updated from the following sources:

- Manual configuration (using the hardware clock)
- Network Time Protocol (NTP)
- Simple Network Time Protocol (SNTP)
- Virtual Integrated Network Service (VINES) Time Service

Because the software clock can be dynamically updated, it has the potential to be more accurate than the hardware clock.

The software clock can provide time to the following services:

- Access lists
- Logging and debugging messages
- NTP

- The hardware clock
- User **show** commands
- VINES Time Service

Note The software clock cannot provide time to the NTP or VINES Time Service if the clock was set using SNTP.

The software clock keeps track of time internally based on the Coordinated Universal Time (UTC), also known as Greenwich Mean Time (GMT). You can configure information about the local time zone and summer time (daylight saving time) so that time is displayed correctly relative to the local time zone.

The software clock keeps track of whether the time is authoritative (that is, whether it has been set by a time source considered to be authoritative). If it is not authoritative, the time will be available only for display purposes and will not be redistributed.

Network Time Protocol

Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs on UDP, which in turn runs on IP. NTP Version 3 (NTPv3) is documented in RFC 1305.

An NTP network usually gets its time from an authoritative time source such as a radio clock or an atomic clock attached to a time server. NTP then distributes this time across the network. NTP is extremely efficient; no more than one packet per minute is necessary to synchronize two machines to the accuracy of within a millisecond of one another.

NTP uses the concept of a stratum to describe how many NTP hops away a machine is from an authoritative time source. A stratum 1 time server typically has an authoritative time source (such as a radio or atomic clock or a Global Positioning System [GPS] time source) directly attached, a stratum 2 time server receives its time via NTP from a stratum 1 time server, and so on.

NTP has two ways to avoid synchronizing to a machine whose time may not be accurate. NTP does not synchronize to a machine that is not in turn synchronized with the NTP. NTP compares the time reported by several machines and does not synchronize to a machine whose time is significantly different from others, even if its stratum is lower. This strategy effectively builds a self-organizing tree of NTP servers.

Our implementation of NTP does not support stratum 1 service; that is, you cannot connect to a radio or atomic clock (for some specific platforms, however, you can connect to a GPS time-source device). We recommend that the time service you derive for your network from the public NTP servers that are available in the IP Internet.

If the network is isolated from the Internet, our implementation of NTP allows a machine to be configured so that it acts as though it is synchronized via NTP, when in fact the network has determined the time by using other means. Other machines can then synchronize to that machine via NTP.

A number of manufacturers include NTP software for their host systems and a publicly available version for systems running UNIX. This software also allows UNIX-derivative servers to acquire the time directly from an atomic clock, which would subsequently propagate time information along to Cisco devices.

The communication between machines running NTP (known as associations) are usually statically configured; each machine is given the IP address of all machines with which it should form associations. Accurate timekeeping is made possible through exchange of NTP messages between each pair of machines with an association.

However, in a LAN environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each machine can be configured to send or receive broadcast messages. However, the accuracy of timekeeping is marginally reduced because the information flow is only one way.

The time kept on a machine is a critical resource, so we strongly recommend that you use the security features of NTP to avoid the accidental or malicious setting of incorrect time. Two security mechanisms are available: an access-list-based restriction scheme and an encrypted authentication mechanism.

When multiple sources of time (VINES, hardware clock, manual configuration) are available, NTP is always considered to be more authoritative. NTP time overrides the time set by any other method.

NTP services are disabled on all interfaces by default.

For more information about NTP, see the following sections:

Poll-Based NTP Associations

Networking devices running NTP can be configured to operate in variety of association modes when synchronizing time with reference time sources. A networking device can obtain time information on a network in two ways—by polling host servers and by listening to NTP broadcasts. This section focuses on the poll-based association modes. Broadcast-based NTP associations are discussed in the *Broadcast-Based NTP Associations* section.

The following are the two most commonly used poll-based association modes:

- Client mode
- Symmetric active mode

The client and the symmetric active modes should be used when NTP is required to provide a high level of time accuracy and reliability.

When a networking device is operating in the client mode, it polls its assigned time-serving hosts for the current time. The networking device will then pick a host from among all the polled time servers to synchronize with. Because the relationship that is established in this case is a client-host relationship, the host will not capture or use any time information sent by the local client device. This mode is most suited for file-server and workstation clients that are not required to provide any form of time synchronization to other local clients. Use the **ntp server** command to individually specify the time server that you want your networking device to consider synchronizing with and to set your networking device to operate in the client mode.

When a networking device is operating in the symmetric active mode, it polls its assigned time-serving hosts for the current time and it responds to polls by its hosts. Because this is a peer-to-peer relationship, the host will also retain time-related information of the local networking device that it is communicating with. This mode should be used when a number of mutually redundant servers are interconnected via diverse network paths. Most stratum 1 and stratum 2 servers on the Internet adopt this form of network setup. Use the **ntp peer** command to individually specify the time serving hosts that you want your networking device to consider synchronizing with and to set your networking device to operate in the symmetric active mode.

The specific mode that you should set for each of your networking devices depends primarily on the role that you want them to assume as a timekeeping device (server or client) and the device's proximity to a stratum 1 timekeeping server.

A networking device engages in polling when it is operating as a client or a host in the client mode or when it is acting as a peer in the symmetric active mode. Although polling does not usually place a burden on memory and CPU resources such as bandwidth, an exceedingly large number of ongoing and simultaneous polls on a system can seriously impact the performance of a system or slow the performance of a given network.

To avoid having an excessive number of ongoing polls on a network, you should limit the number of direct, peer-to-peer or client-to-server associations. Instead, you should consider using NTP broadcasts to propagate time information within a localized network.

Broadcast-Based NTP Associations

Broadcast-based NTP associations should be used when time accuracy and reliability requirements are modest and if your network is localized and has more than 20 clients. Broadcast-based NTP associations are also recommended for use on networks that have limited bandwidth, system memory, or CPU resources.

A networking device operating in the broadcast client mode does not engage in any polling. Instead, it listens for NTP broadcast packets that are transmitted by broadcast time servers. Consequently, time accuracy can be marginally reduced because time information flows only one way.

Use the **ntp broadcast client** command to set your networking device to listen for NTP broadcast packets propagated through a network. For broadcast client mode to work, the broadcast server and its clients must be located on the same subnet. You must enable the time server that transmits NTP broadcast packets on the interface of the given device by using the **ntp broadcast** command.

NTP Access Group

The access list-based restriction scheme allows you to grant or deny certain access privileges to an entire network, a subnet within a network, or a host within a subnet. To define an NTP access group, use the **ntp access-group** command in global configuration mode.

The access group options are scanned in the following order, from least restrictive to the most restrictive:

1. **ipv4**—Configures IPv4 access lists.
2. **ipv6**—Configures IPv6 access lists.
3. **peer**—Allows time requests and NTP control queries, and allows the system to synchronize itself to a system whose address passes the access list criteria.
4. **serve**—Allows time requests and NTP control queries, but does not allow the system to synchronize itself to a system whose address passes the access list criteria.
5. **serve-only**—Allows only time requests from a system whose address passes the access list criteria.
6. **query-only**—Allows only NTP control queries from a system whose address passes the access list criteria.

If the source IP address matches the access lists for more than one access type, the first type is granted access. If no access groups are specified, all access types are granted access to all systems. If any access groups are specified, only the specified access types will be granted access.

For details on NTP control queries, see RFC 1305 (NTP Version 3).

The encrypted NTP authentication scheme should be used when a reliable form of access control is required. Unlike the access list-based restriction scheme that is based on IP addresses, the encrypted authentication scheme uses authentication keys and an authentication process to determine if NTP synchronization packets sent by designated peers or servers on a local network are deemed as trusted before the time information that they carry along with them is accepted.

The authentication process begins from the moment an NTP packet is created. Cryptographic checksum keys are generated using the message digest algorithm 5 (MD5) and are embedded into the NTP synchronization packet that is sent to a receiving client. Once a packet is received by a client, its cryptographic checksum key is decrypted and checked against a list of trusted keys. If the packet contains a matching authentication key,

the time-stamp information that is contained within the packet is accepted by the receiving client. NTP synchronization packets that do not contain a matching authenticator key are ignored.

Note In large networks, where many trusted keys must be configured, the Range of Trusted Key Configuration feature enables configuring multiple keys simultaneously.

It is important to note that the encryption and decryption processes used in NTP authentication can be very CPU-intensive and can seriously degrade the accuracy of the time that is propagated within a network. If your network setup permits a more comprehensive model of access control, you should consider the use of the access list-based form of control.

After NTP authentication is properly configured, your networking device will synchronize with and provide synchronization only to trusted time sources.

NTP Services on a Specific Interface

Network Time Protocol (NTP) services are disabled on all interfaces by default. NTP is enabled globally when any NTP commands are entered. You can selectively prevent NTP packets from being received through a specific interface by using the **ntp disable** command in interface configuration mode.

Source IP Address for NTP Packets

When the system sends an NTP packet, the source IP address is normally set to the address of the interface through which the NTP packet is sent. Use the **ntp source** *interface* command in global configuration mode to configure a specific interface from which the IP source address will be taken.

This interface will be used for the source address for all packets sent to all destinations. If a source address is to be used for a specific association, use the **source** keyword in the **ntp peer** or **ntp server** command.

System as an Authoritative NTP Server

Use the **ntp** command in global configuration mode if you want the system to be an authoritative NTP server, even if the system is not synchronized to an outside time source.

Note Use the **ntp primary** command with caution. It is very easy to override valid time sources using this command, especially if a low stratum number is configured. Configuring multiple machines in the same network with the **ntp primary** command can cause instability in timekeeping if the machines do not agree on the time.

Orphan Mode

The NTP subnet is sometimes isolated from local reference clocks or Internet clock servers. During this period of isolation, the subnet servers and clients are synchronized to a common time scale. The local clock driver simulates a UTC source to provide a common time scale. A server connected to the driver directly or indirectly synchronizes the other hosts in the subnet.

Using a local clock driver may sometimes result in irrecoverable failures of the subnet, and maintaining redundancy using multiple servers is not feasible. The Orphan Mode feature, which does not have any such disadvantages, eliminates the need for a local clock driver. The Orphan Mode feature provides a single

simulated UTC source with multiple servers and a seamless switching mechanism as servers recover from a failure.

In private networks, one or multiple core servers operating at the lowest stratum is normally included. You must configure each of these servers as backups for other servers using symmetric or broadcast modes. Even if one core server reaches a UTC source, the entire subnet synchronizes to the simulating server. If none of the servers reach a UTC source, one of the servers, which is known as the orphan parent, can simulate a UTC source, and serve as the simulated UTC source for all the other hosts, known as orphan children, in the subnet.

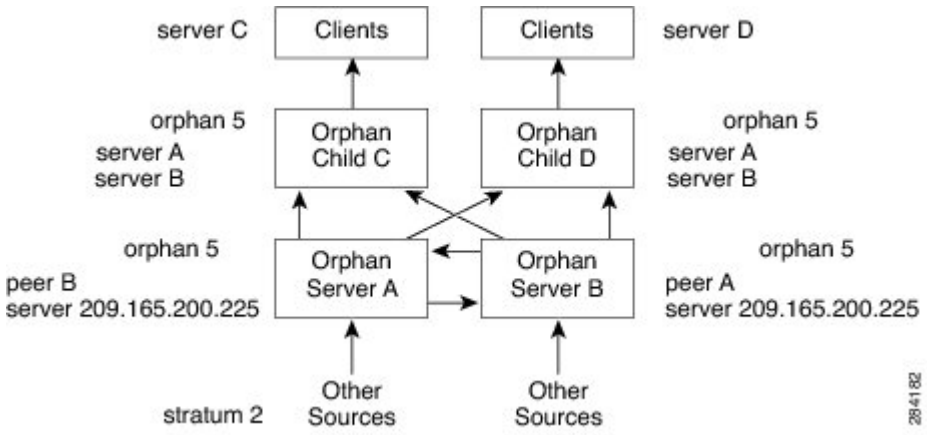
Use the **ntp orphan** *stratum* command to enable a host for orphan mode, where *stratum* is a stratum value less than 16 and greater than any stratum value that occurs in the configured Internet time servers. However, you must provide sufficient stratums so that every subnet host dependent on the orphan children has a stratum value less than 16. If no associations for other servers or reference clocks are configured, you must set the orphan stratum value to 1.

An orphan parent operating at stratum 1 with no sources displays the reference ID LOOP. An orphan parent not operating at stratum 1 displays the UNIX loopback address 127.0.0.1. Ordinary NTP clients use a selection metric based on delay and dispersion, whereas orphan children use a metric computed from the IP address of each core server in the subnet. Each orphan child selects the orphan parent with the smallest metric as the root server.

A server that loses all sources, continuously synchronizes the local clock driver with other servers, thus backing up the server. Enable orphan mode only in core servers and orphan children.

The following figure illustrates how orphan mode is set up, and a peer network configuration, where two primary or secondary (stratum 2) servers are configured with reference clocks or public Internet primary servers, with each using symmetric modes.

Figure 1: Orphan Mode Setup



Prerequisites for Orphan Mode

To ensure smooth function of the orphan mode, you must configure each core server with available sources to operate at the same stratum. Configure the **ntp orphan** command in all the core servers and the orphan children. Configure each orphan child with all root servers.

Simple Network Time Protocol

Simple Network Time Protocol (SNTP) is a simplified, client-only version of NTP. SNTP can receive only the time from NTP servers; it cannot be used to provide time services to other systems.

SNTP typically provides time within 100 milliseconds of the accurate time, but it does not provide the complex filtering and statistical mechanisms of NTP. In addition, SNTP does not authenticate traffic, although you can configure extended access lists to provide some protection. An SNTP client is more vulnerable to servers that have unexpected behavior than an NTP client, and should be used only in situations where strong authentication is not required.

You can configure SNTP to request and accept packets from configured servers or to accept NTP broadcast packets from any source. When multiple sources are sending NTP packets, the server with the best stratum is selected. (See the *Network Time Protocol* section on page 3 for a description of strata.) If multiple servers are at the same stratum, a configured server is preferred over a broadcast server. If multiple servers pass both tests, the first one to send a time packet is selected. SNTP will choose a new server only if it stops receiving packets from the currently selected server, or if a better server (according to the criteria described) is discovered.

VINES Time Service

Time service is available when Banyan VINES is configured. This protocol is a standard part of VINES. The Cisco implementation allows the VINES time service to be used in two ways. First, if the system has learned the time from some other source, it can act as a VINES time server and provide time to other machines running VINES. Second, it can use the VINES time service to set the software clock if no other form of time service is available.

Note Support for Banyan VINES and Xerox Network Systems (XNS) is not available in all releases.

Hardware Clock

Some devices contain a battery-powered hardware clock that tracks the date and time across system restarts and power outages. The hardware clock is always used to initialize the software clock when the system is restarted.

Note Within the CLI command syntax, the hardware clock is referred to as the system calendar.

If no other source is available, the hardware clock can be considered as an authoritative source of time and be redistributed via NTP. If NTP is running, the hardware clock can be updated periodically from NTP, compensating for the inherent drift, which is the consistent gain or loss of time at a certain rate if the hardware clock is left to run.

You can configure a hardware clock (system calendar) on any device to be periodically updated from the software clock. We recommend that you use this configuration for any device using NTP, because the time and date on the software clock (set using NTP) will be more accurate than the hardware clock, because the time setting on the hardware clock has the potential to drift slightly over time.

Use the **ntp update-calendar** command in global configuration mode if a routing device is synchronized to an outside time source via NTP and you want the hardware clock to be synchronized to NTP time.

Time Ranges

The Cisco software allows implementation of features based on the time of day. The **time-range** global configuration command defines specific times of the day and week, which can then be referenced by a function, so that those time restrictions are imposed on the function itself.

Depending on your release, IP and Internetwork Packet Exchange (IPX) extended access lists are the only functions that can use time ranges. The time range allows the network administrator to define when the permit or deny statements in the access list are in effect. Prior to the introduction of this feature, access list statements were always in effect once they were applied. Both named and numbered access lists can reference a time range.

Note The time range relies on the system's software clock. For the time range feature to work the way you intend, you need a reliable clock source. We recommend that you use NTP to synchronize the system's software clock.

Benefits of time ranges include the following:

- The network administrator has more control over permitting or denying a user access to resources. These resources could be an application (identified by an IP address/mask pair and a port number), policy routing, or an on-demand link (identified as interesting traffic to the dialer).
- Network administrators can set a time-based security policy, including the following:
  - Perimeter security using the Cisco Firewall feature set or access lists.
  - Data confidentiality with Cisco Encryption Technology or IP security.
- Policy-based routing and queueing functions are enhanced.
- When provider access rates vary by time of day, traffic can be rerouted automatically and cost-effectively.
- Service providers can dynamically change a committed access rate (CAR) configuration to support the quality of service (QoS) service level agreements (SLAs) that are negotiated for certain times of the day.

Network administrators can control logging messages. Access list entries can log traffic at certain times of the day, but not constantly. Therefore, administrators can deny access without the need to analyze the many logs generated during peak hours.

How to Configure Network Time Protocol

Configuring NTP

Restrictions for Network Time Protocol

The Network Time Protocol (NTP) package contains a vulnerability that could allow an unauthenticated, remote attacker to cause a denial of service (DoS) condition. NTP versions 4.2.4p7 and earlier are vulnerable.

The vulnerability is due to an error in handling of certain malformed messages. An unauthenticated, remote attacker could send a malicious NTP packet with a spoofed source IP address to a vulnerable host. The host that processes the packet sends a response packet back to the transmitter. This action could start a loop of

messages between the two hosts that could cause both the hosts to consume excessive CPU resources, use up the disk space by writing messages to log files, and consume the network bandwidth. All of these could cause a DoS condition on the affected hosts.

For more information, see the Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability web page.

Cisco software releases that support NTPv4 are not affected. All other versions of Cisco software are affected.

To display whether a device is configured with NTP, use the **show running-config | include ntp** command. If the output returns any of the following commands, then that device is vulnerable to the attack:

- **ntp broadcast client**
- **ntp primary**
- **ntp multicast client**
- **ntp peer**
- **ntp server**

For more information on understanding Cisco software releases, see the White Paper: Cisco IOS and NX-OS Software Reference Guide.

There are no workarounds for this vulnerability other than disabling NTP on the device. Only packets destined for any configured IP address on the device can exploit this vulnerability. Transit traffic will not exploit this vulnerability.

Depending on your release, your feature will process NTP mode 7 packets and will display the message "NTP: Receive: dropping message: Received NTP private mode 7 packet " if debugs for NTP are enabled. Configure the **ntp allow mode private** command to process NTP mode 7 packets. This command is disabled by default.

Note NTP peer authentication is not a workaround and is a vulnerable configuration.

NTP services are disabled on all interfaces by default.

Networking devices running NTP can be configured to operate in a variety of association modes when synchronizing time with reference time sources. A networking device can obtain time information on a network in two ways: by polling host servers and by listening to NTP broadcasts.

Line Aux 0 option is disabled by default.

When you configure both IP address and FQDN of the same NTP server in Cisco IOS XE, only the FQDN configuration is displayed in the **show running-config** command output after FQDN resolves to the same IP address.

Configuring Poll-Based NTP Associations

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp peer** *ip-address* [**normal-sync**] [**version** *number*] [**key** *key-id*] [**prefer**]
4. **ntp server** *ip-address* [**version** *number*] [**key** *key-id*] [**prefer**]

5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ntp peer** *ip-address* [**normal-sync**] [**version** *number*] [**key** *key-id*] [**prefer**] **Example:** `Device(config)# ntp peer 192.168.10.1 normal-sync version 2 prefer`	Forms a peer association with another system.
Step 4	**ntp server** *ip-address* [**version** *number*] [**key** *key-id*] [**prefer**] **Example:** `Device(config)# ntp server 192.168.10.1 version 2 prefer`	Forms a server association with another system.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Broadcast-Based NTP Associations

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ntp broadcast version** *number*
5. **ntp broadcast client**
6. **ntp broadcastdelay** *microseconds*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0`	Configures an interface and enters interface configuration mode.
Step 4	**ntp broadcast version** *number* **Example:** `Device(config-if)# ntp broadcast version 2`	Configures the specified interface to send NTP broadcast packets.
Step 5	**ntp broadcast client** **Example:** `Device(config-if)# ntp broadcast client`	Configures the specified interface to receive NTP broadcast packets.
Step 6	**ntp broadcastdelay** *microseconds* **Example:** `Device(config-if)# ntp broadcastdelay 100`	Adjusts the estimated round-trip delay for NTP broadcasts.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring NTP Authentication

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp authenticate**
4. **ntp authentication-key** *number* **md5** *key*
5. **ntp authentication-key** *number* **md5** *key*
6. **ntp authentication-key** *number* **md5** *key*

7. **ntp trusted-key** *key-number* [- *end-key*]
8. **ntp server** *ip-address* **key** *key-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ntp authenticate** **Example:** `Device(config)# ntp authenticate`	Enables the NTP Authentication feature.
Step 4	**ntp authentication-key** *number* **md5** *key* **Example:**	Defines authentication keys. • Each key has a key number, a type, and a value.
Step 5	**ntp authentication-key** *number* **md5** *key* **Example:**	Defines authentication keys. • Each key has a key number, a type, and a value.
Step 6	**ntp authentication-key** *number* **md5** *key* **Example:**	Defines authentication keys. • Each key has a key number, a type, and a value.
Step 7	**ntp trusted-key** *key-number* [- *end-key*] **Example:** `Device(config)# ntp trusted-key 1 - 3`	Defines trusted authentication keys. • If a key is trusted, this device will be ready to synchronize to a system that uses this key in its NTP packets.
Step 8	**ntp server** *ip-address* **key** *key-id* **Example:** `Device(config)# ntp server 172.16.22.44 key 2`	Allows the software clock to be synchronized by an NTP time server. **Note** When multiple NTP servers are configured and logging is enabled, clock synchronization lost messages are randomly seen on the device. To resolve this issue, configure the NTP server using **peer** keyword. `Device(config)# ntp server ip-address [version number] [key key-id] [prefer]`

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring an External Reference Clock

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line aux** *line-number*
4. **end**
5. **show ntp associations**
6. **show ntp status**
7. **debug ntp refclock**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line aux** *line-number* **Example:** `Device(config)# line aux 0`	Enters line configuration mode for the auxiliary port 0.
Step 4	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.
Step 5	**show ntp associations** **Example:** `Device# show ntp associations`	Displays the status of NTP associations, including the status of the GPS reference clock.

	Command or Action	Purpose
Step 6	**show ntp status** **Example:** `Device# show ntp status`	Displays the status of NTP.
Step 7	**debug ntp refclock** **Example:** `Device# debug ntp refclock`	Allows advanced monitoring of reference clock activities for the purposes of debugging.

Configuring Orphan Mode

To configure orphan mode, you would require at least two clients. The following task shows how to configure orphan mode on one client. Repeat the steps in the other client.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp server** *ip-address*
4. **ntp peer** *ip-address*
5. **ntp orphan** *stratum*
6. Repeat steps 1 to 5 on the other client.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ntp server** *ip-address* **Example:** `Router(config)# ntp server 10.1.1.1`	Forms a server association with another system.
Step 4	**ntp peer** *ip-address* **Example:** `Router(config)# ntp peer 172.16.0.1`	Forms a peer association with another system. **Note** Use an IP address that is different from the one you just configured, such as 172.16.0.2, while configuring the peer in the other client.

	Command or Action	Purpose
Step 5	**ntp orphan** *stratum* **Example:** `Router(config)# ntp orphan 4`	Enables orphan mode in the host.
Step 6	Repeat steps 1 to 5 on the other client.	

Configuring Rate-Limiting Delay

use the following example to configure rate-limiting delay to NTP mode-6 queries:

```
Router(config)#ntp allow mode control ?
  <0-15>  Rate limiting delay (s)
  <cr>    <cr>

Router(config)#ntp allow mode control 7
Router(config)#
```

Configuring SNTP

SNTP generally is supported on those platforms that do not provide support for NTP. SNTP is disabled by default. To configure SNTP, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sntp server** {*address* | *hostname*} [**version** *number*]
4. **sntp broadcast client**
5. **exit**
6. **show sntp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sntp server** {*address* | *hostname*} [**version** *number*] **Example:**	Configures SNTP to request NTP packets from an NTP server.

	Command or Action	Purpose
	`Device(config)# sntp server 192.168.2.1 version 2`	• Enter the **sntp server** command once for each NTP server. The NTP servers must be configured to respond to the SNTP messages from the device.
Step 4	**sntp broadcast client** **Example:** `Device(config)# sntp broadcast client`	Configures SNTP to accept NTP packets from any NTP broadcast server. **Note** If you enter both the **sntp server** command and the **sntp broadcast client** command, the device will accept time from a broadcast server but will prefer time from a configured server, assuming that the strata are equal.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show sntp** **Example:** `Device# show sntp`	Displays information about SNTP.

Configuring VINES Time Service

Time service is available when Banyan VINES is configured. This protocol is a standard part of VINES. Perform the following task to configure VINES Time Service.

Note Depending on your release, the Banyan VINES and XNS is available in the Cisco software. The **vines time set-system** and **vines time use-system** commands are not available in some releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vines time use-system**
4. **vines time set-system**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vines time use-system** **Example:** `Device(config)# vines time use-system`	Distributes the system software clock time to other VINES systems.
Step 4	**vines time set-system** **Example:** `Device(config)# vines time set-system`	Sets the software clock system time and date as derived from VINES time services.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring the Time and Date

If no other source of time is available, you can manually configure the current time and date after the system is restarted. The time will remain accurate until the next system restart. We recommend that you use manual configuration only as a last resort.

If you have an outside source to which the device can synchronize, you need not manually set the software clock. Perform the following task to configure the time and date manually.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clock timezone** *zone hours-offset* [*minutes-offset*]
4. **clock summer-time** *zone* **recurring** [*week day month hh*:*mm week day month hh*:*mm* [*offset*]]
5. **clock summer-time** *zone* **date** *date month year hh*:*mm date month year hh*:*mm* [*offset*]
6. **exit**
7. **clock set** *hh*:*mm*:*ss date month year*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clock timezone** *zone hours-offset* [*minutes-offset*] **Example:** `Device(config)# clock timezone PST 2 30`	Configures the time zone used by the Cisco software. **Note** The *minutes-offset* argument of the **clock timezone** command is available for those cases where a local time zone is a percentage of an hour different from UTC/GMT. For example, the time zone for some sections of Atlantic Canada (AST) is UTC –3.5. In this case, the necessary command would be **clock timezone AST –3 30**.
Step 4	**clock summer-time** *zone* **recurring** [*week day month hh*:*mm week day month hh*:*mm* [*offset*]] **Example:** `Device(config)# clock summer-time PST recurring 1 monday january 12:12 4 Tuesday december 12:12 120`	Configures summer time (daylight saving time) in areas where it starts and ends on a particular day of the week each year.
Step 5	**clock summer-time** *zone* **date** *date month year hh*:*mm date month year hh*:*mm* [*offset*] **Example:** `Device(config)# clock summer-time PST date 1 january 1999 12:12 4 december 2001 12:12 120`	Configures a specific summer time start and end date. • The *offset* argument is used to indicate the number of minutes to add to the clock during summer time.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 7	**clock set** *hh*:*mm*:*ss date month year* **Example:** `Device# clock set 12:12:12 1 january 2011`	Sets the software clock. • Use this command if no other time sources are available. The time specified in this command is relative to the configured time zone. **Note** Generally, if the system is synchronized by a valid outside timing mechanism, such as an NTP or VINES clock source, or if you have a device with a hardware clock, you need not set the software clock.

Setting the Hardware Clock

Most Cisco devices have a separate hardware-based clock in addition to the software-based clock. The hardware clock is a chip with a rechargeable backup battery that can retain the time and date information across reboots of the device.

To maintain the most accurate time update from an authoritative time source on the network, the software clock should receive time updates from an authoritative time on the network. The hardware clock should in turn be updated at regular intervals from the software clock while the system is running.

The hardware clock (system calendar) maintains time separately from the software clock. The hardware clock continues to run when the system is restarted or when the power is turned off. Typically, the hardware clock needs to be manually set only once, when the system is installed.

You should avoid setting the hardware clock if you have access to a reliable external time source. Time synchronization should instead be established using NTP.

Perform the following task to set the hardware clock.

Before you begin

Note Depending on your release, NTP runs within IOS daemon (IOSd), which updates the time on the Linux kernel. As the Linux kernel updates the hardware clock every 11 minutes, NTP does not interact with the hardware clock directly. So, the calendar-related commands are not required.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clock calendar-valid**
4. **exit**
5. **clock read-calendar**
6. **clock update-calendar**
7. **show calendar**
8. **show clock** [**detail**]
9. **show ntp associations** [**detail**]
10. **show ntp status**
11. **show sntp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clock calendar-valid** **Example:** `Device(config)# clock calendar-valid`	Enables the device to act as a valid time source to which network peers can synchronize. • By default, the time maintained on the software clock is not considered to be reliable and will not be synchronized with NTP or VINES time service. To set the hardware clock as a valid time source, use this command.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**clock read-calendar** **Example:** `Device# clock read-calendar`	Sets the software clock to the new hardware clock setting.
Step 6	**clock update-calendar** **Example:** `Device# clock update-calendar`	Updates the hardware clock with a new software clock setting.
Step 7	**show calendar** **Example:** `Device# show calendar`	Displays the current hardware clock time.
Step 8	**show clock** [**detail**] **Example:** `Device# show clock detail`	Displays the current software clock time.
Step 9	**show ntp associations** [**detail**] **Example:** `Device# show ntp associations detail`	Displays the status of NTP associations.
Step 10	**show ntp status** **Example:** `Device# show ntp status`	Displays the status of NTP.

	Command or Action	Purpose
Step 11	**show sntp** **Example:** `Device# show sntp`	Displays information about SNTP.

Configuring Time Ranges

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **time-range** *time-range-name*
4. Enter one of the following:
   - **absolute** [**start** *hh*:*mm date month year*] [**end** *hh*:*mm date month year*]
   - **periodic** *day-of-the-week hh*:*mm* **to** [*day-of-the-week*] *hh*:*mm*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**time-range** *time-range-name* **Example:** `Device(config)# time-range range1`	Assigns a name to the time range to be configured and enters time range configuration mode.
Step 4	Enter one of the following: • **absolute** [**start** *hh*:*mm date month year*] [**end** *hh*:*mm date month year*] • **periodic** *day-of-the-week hh*:*mm* **to** [*day-of-the-week*] *hh*:*mm* **Example:** `Device(config-time-range)# absolute start 12:12 30 January 1999 end 12:12 30 December 2000`	Specifies when the time range will be in effect. • Use a combination of these commands; multiple **periodic** commands are allowed; only one **absolute** command is allowed.

	Command or Action	Purpose
	`Device(config-time-range)# periodic monday 12:12 to friday 12:12`	
Step 5	**end** **Example:** `Device(config-time-range)# end`	Exits time range configuration mode and returns to privileged EXEC mode.

Verifying Network Time Protocol

SUMMARY STEPS

1. **show calendar**
2. **show clock** [**detail**]
3. **show ntp associations detail**
4. **show ntp status**
5. **show running-config| i-ntp**

DETAILED STEPS

Step 1 **show calendar**

This command displays the current hardware clock time. The following is sample output from this command.

Example:

```
Device# show calendar

18:34:29 UTC Tue Jan 4 2011
```

Step 2 **show clock** [**detail**]

This command displays the current software clock time. The following is sample output from this command.

Example:

```
Device# show clock detail

*18:38:21.655 UTC Tue Jan 4 2011
Time source is hardware calendar
```

Step 3 **show ntp associations detail**

This command displays the status of NTP associations. The following is sample output from this command.

Example:

```
Device# show ntp associations detail

192.168.10.1 configured, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
```

```
our mode active, peer mode unspec, our poll intvl 64, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 15940.56
delay 0.00 msec, offset 0.0000 msec, dispersion 15937.50
precision 2**24, version 4
org time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
rec time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
xmt time D0CDE881.9A6A9005 (18:42:09.603 UTC Tue Jan 4 2011)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
192.168.45.1 configured, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
our mode client, peer mode unspec, our poll intvl 64, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 16003.08
delay 0.00 msec, offset 0.0000 msec, dispersion 16000.00
precision 2**24, version 4
org time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
rec time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
xmt time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
```

Step 4 **show ntp status**

This command displays the status of NTP. The following is sample output from this command.

Example:

```
Device# show ntp status

Clock is synchronized, stratum 8, reference is 127.127.1.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
reference time is D25AF07C.4B439650 (15:26:04.294 PDT Tue Oct 21 2011)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 2.31 msec, peer dispersion is 1.20 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 16, last update was 10 sec ago.
```

Step 5 **show running-config| i-ntp**

This command displays the configured rate-limiting delay. The following is sample output from this command, with rate-limiting delay set to 7 seconds:

Example:

```
device# show running-config | i ntp
ntp allow mode control 7

device# show running-config all | i ntp mode
ntp allow mode control 7
```

Configuration Examples for Network Time Protocol

Example: Configuring Network Time Protocol

In the following example, a device with a hardware clock that has server associations with two other systems sends broadcast NTP packets, periodically updates the hardware clock, and redistributes time into VINES:

```
clock timezone PST -8
clock summer-time PDT recurring

ntp server 192.168.13.57
ntp server 192.168.11.58
interface GigabitEthernet 0/0
 ntp broadcast
vines time use-system
```

In the following example, a device with a hardware clock has no outside time source, so it uses the hardware clock as an authoritative time source and distributes the time via NTP broadcast packets:

```
clock timezone MET 2
clock calendar-valid
ntp master
interface vlan 3
 ntp broadcast
```

The following example shows Line Aux 0 option is disabled by default.

```
config-register 0x0
reload
rommon 1 > set
rommon 2 >  AUX_PORT=1
rommon 3 > SYNC
rommon 4 > reset
rommon 1 > set
rommon 2 > confreg 0x2102
rommon 3 > reset
```

Additional References for Network Time Protocol

Related Documents

Related Topic	Document Title
Basic System Management commands	Basic System Management Command Reference
NTP4 in IPv6	*Cisco IOS Basic System Management Guide*
IP extended access lists	*Cisco IOS IP Addressing Configuration Guide*
IPX extended access lists	*Novell IPX Configuration Guide*
NTP package vulnerability	*Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability*

Related Topic	Document Title
Cisco IOS and NX-OS software releases	*'White Paper: Cisco IOS and NX-OS Software Reference Guide*

Standards and RFCs

Standard/RFCs	Title
RFC 1305	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

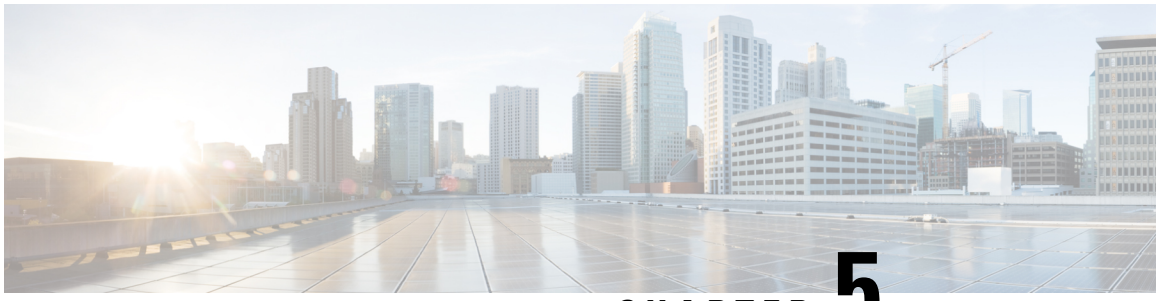
Feature Information for Network Time Protocol

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4: Feature Information for Network Time Protocol

Feature Name	Releases	Feature Information
Network Time Protocol	11.2(1) 12.2(28)SB 12.2(33)SRA 12.2(33)SXI 12.2(33)SXJ 12.2(50)SY 12.2(58)SE 15.0(1)M 15.1(2)S 15.1(2)SG Cisco IOS XE Release 3E	NTP is a protocol designed to time-synchronize a network of machines. NTP runs on UDP, which in turn runs on IP. NTP is documented in RFC 1305. The following commands were introduced or modified: **ntp access-group**, **ntp allow mode passive**, **ntp authenticate**, **ntp authentication-key**, **ntp broadcast, ntp broadcast client**, **ntp broadcastdelay**, **ntp clear drift**, **ntp clock-period**, **ntp disable**, **ntp logging**, **ntp primary**, **ntp max-associations**, **ntp multicast**, **ntp multicast client**, **ntp server**, **ntp source**, **ntp trusted-key** and **ntp update-calendar**.



CHAPTER 5

Simple Network Time Protocol

Simple Network Time Protocol (SNTP) is a simplified version of Network Time Protocol(NTP). This module describes how to configure Simple Network Time Protocol on Cisco devices.

Restrictions for Simple Network Time Protocol

- Simple Network Time Protocol (SNTP) and Network Time Protocol (NTP) cannot coexist on the same machine as they use the same port. This means that these two services cannot be configured on the system at the same time.
- Support for IPv6 addresses is available only if the image supports IPv6 addressing.

Information About Simple Network Time Protocol

Simple Network Time Protocol

Simple Network Time Protocol (SNTP) is a simplified, client-only version of NTP. SNTP can receive only the time from NTP servers; it cannot be used to provide time services to other systems.

SNTP typically provides time within 100 milliseconds of the accurate time, but it does not provide the complex filtering and statistical mechanisms of NTP. In addition, SNTP does not authenticate traffic, although you can configure extended access lists to provide some protection. An SNTP client is more vulnerable to servers that have unexpected behavior than an NTP client, and should be used only in situations where strong authentication is not required.

You can configure SNTP to request and accept packets from configured servers or to accept NTP broadcast packets from any source. When multiple sources are sending NTP packets, the server with the best stratum is selected. (See the *Network Time Protocol* section on page 3 for a description of strata.) If multiple servers

are at the same stratum, a configured server is preferred over a broadcast server. If multiple servers pass both tests, the first one to send a time packet is selected. SNTP will choose a new server only if it stops receiving packets from the currently selected server, or if a better server (according to the criteria described) is discovered.

How to Configure Simple Network Time Protocol

Configuring Simple Network Time Protocol (SNTP) Authentication

Simple Network Time Protocol (SNTP) is a simplified version of Network Time Protocol (SNTP). This module describes how to configure SNTP on Cisco devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sntp authenticate**
4. **sntp authentication-key** *number* **md5** *key*
5. **sntp trusted-key** *key-number* [- *end-key*]
6. **sntp server** *ip-address* **key** *key-id*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sntp authenticate** **Example:** `Device(config)# sntp authenticate`	Enables the SNTP Authentication feature.
Step 4	**sntp authentication-key** *number* **md5** *key* **Example:** `Device(config)# sntp authentication-key 1 md5 key1`	Defines authentication keys. • Each key has a key number, a type, and a value. • Repeat this step to define additional authentication keys.
Step 5	**sntp trusted-key** *key-number* [- *end-key*] **Example:**	Defines trusted authentication keys.

	Command or Action	Purpose
	`Device(config)# sntp trusted-key 1 - 3`	• If a key is trusted, this device will be ready to synchronize to a system that uses this key in its SNTP packets.
Step 6	**sntp server** *ip-address* **key** *key-id* **Example:** `Device(config)# sntp server 172.16.22.44 key 2`	Allows the software clock to be synchronized by an SNTP time server.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying and Troubleshooting Simple Network Time Protocol

To verify and troubleshoot Simple Network Time Protocol configuration, use the following commands.

•

SUMMARY STEPS

1. **enable**
2. **debug sntp packets** [**detail**]
3. **debug sntp select**
4. **show sntp**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

• Enter your password if prompted.

Step 2 **debug sntp packets** [**detail**]

Example:

```
Device> debug sntp packets
```

Displays the NTP packet sent and received along with the SNTP packet fields.

Step 3 **debug sntp select**

Example:

```
Device> debug sntp select
```

Displays the SNTP server selection for IPv4 and IPv6 servers.

Step 4 **show sntp**

Example:

```
Device# show sntp

SNTP server     Stratum   Version    Last Receive
172.168.10.1      16        1        never
Broadcast client mode is enabled.
Multicast client 224.0.1.1 is enabled.
```

Displays information about SNTP available in Cisco devices.

Configuration Examples for Simple Network Time Protocol

Example: Configuring Simple Network Time Protocol

```
clock timezone PST -8
clock summer-time PDT recurring
sntp update-calendar
sntp server 192.168.13.57
sntp server 192.168.11.58
interface Ethernet 0/0
 sntp broadcast
```

Additional References for Simple Network Time Protocol

Related Documents

Related Topic	Document Title
Basic System Management commands	Basic System Management Command Reference
NTP4 in IPv6	*Cisco IOS Basic System Management Guide*
IP extended access lists	*Cisco IOS IP Addressing Configuration Guide*
IPX extended access lists	*Novell IPX Configuration Guide*
NTP package vulnerability	*Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability*
Cisco IOS and NX-OS software releases	*White Paper: Cisco IOS and NX-OS Software Reference Guide*

Standards and RFCs

Standard/RFCs	Title
RFC 1305	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for the SNTP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

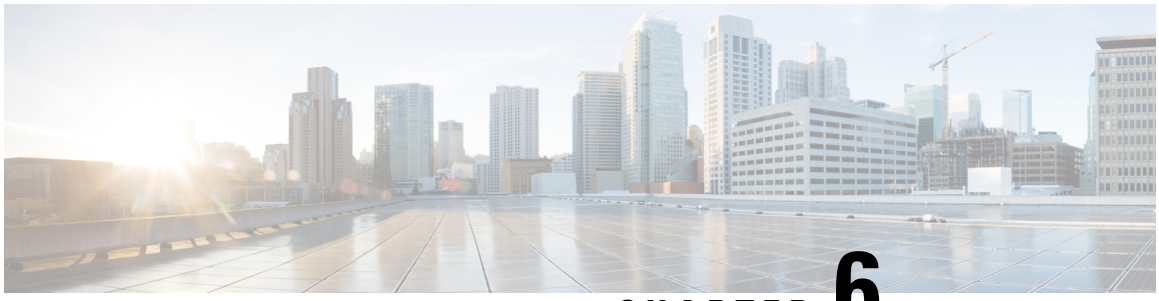
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5: Feature Information for the SNTPv4

Feature Name	Releases	Feature Information
Simple Network Time Protocol		Simple Network Time Protocol (SNTP) is a simplified version of Network Time Protocol(NTP). This module describes how to configure Simple Network Time Protocol on Cisco devices. The following commands were introduced or modified: **sntp server**, **sntp authenticate**, **sntp authentication-key**, **sntp multicast**, **sntp trusted-key**.

PART II

Configuration Fundamentals



CHAPTER 6

Using the Cisco IOS 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 Using Setup Mode to Configure a Cisco Networking Device and Using AutoInstall to Remotely Configure Cisco Networking Devices. For information on issuing commands using the Cisco Web Browser, see "Using the Cisco Web Browser User Interface". For information on user menus, see "Managing Connections, Menus, and System Banners".

For a complete description of the user interface commands in this chapter, see the *Cisco IOS Configuration Fundamentals Command Reference*. To locate documentation of other commands that appear in this chapter, use the Cisco IOS Master Command List, All Releases .

Cisco IOS XE CLI Command Modes Overview

To aid in the configuration of Cisco devices, the Cisco IOS XE 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, because 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.

Cisco IOS XE CLI Task List

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

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*)# *abbreviated-command-entry***?**	Lists commands in the current mode that begin with a particular character string.

Command	Purpose
(*prompt*)# *abbreviated-command-entry* <**Tab**>	Completes a partial command name.
(*prompt*)# **?**	Lists all commands available in the command mode.
(*prompt*)# *command***?**	Lists the available syntax options (arguments and keywords) for the command.
(*prompt*)# *command keyword* **?**	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 context-sensitive help is used, 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 **configureterminal**command to **configt**. 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 thequestion mark. The system provides the commands that begin with **co**.

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

Enter the **configure** command followed by a space and aquestion 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 keywords. In this example, the output indicates that your options for the configure command are **configurememory** (configure from NVRAM), **configurenetwork** (configure from a file on the network), **configureoverwrite-network** (configure from a file on the network and replace the file in NVRAM), or **configureterminal** (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, because 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.”

Enter the **configureterminal** 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 terminal
                   ^
% Invalid input detected at '^' marker.
Router#
```

Note that an error message (indicated by the % symbol) appears on 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
 <1100-1199>       Extended 48-bit MAC address access list
 <1300-1999>       IP standard access list (expanded range)
```

```
 <200-299>         Protocol type-code access list
 <2000-2699>       IP extended access list (expanded range)
 <700-799>         48-bit MAC address access list
 dynamic-extended  Extend the dynamic ACL absolute timer
 rate-limit        Simple rate-limit specific 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.

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 **noiprouting** form of the **iprouting** command. To reenable it, use the plain **iprouting** 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 **default***command-name* command, you can configure the command to its default setting. The Cisco IOS software command reference documents generally 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:

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

The table below shows the key combinations or sequences you can use to move the cursor 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 the table below characters are bolded in the "Function Summary" column to indicate the relation between the letter used and the function.

Table 6: Key Combinations Used to Move the Cursor

Keystrokes	Function Summary	Function Details
Left Arrow or **Ctrl-B**	**B** ack 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**	**F** orward character	Moves the cursor one character to the right.
Esc , **B**	**B** ack word	Moves the cursor back one word.
Esc , **F**	**F** orward word	Moves the cursor forward one word.
Ctrl -A	Beginning of line	Moves the cursor to the beginning of the line.
Ctrl -E	**E** nd 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 cannot 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 appear on the screen to allow you to complete the command entry.

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, Ymore 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 XE software assumes you have a terminal screen that is 80 columns wide. If you have a different screen-width, use the **terminal width** user 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.

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.

Continuing Output at the --More-- Prompt

When you use the Cisco IOS XE 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 appears 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.

Tip If output is pausing on your screen, but you do not see the --More-- prompt, try entering a lower value for the screen length using the **length** line configuration command or the **terminal length** privileged EXEC mode 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 module 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 to the right of 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 XE 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 a 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 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 user EXEC mode:

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

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

Command	Purpose
`Router# terminal editing`	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)# editing`	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 user EXEC or privileged EXEC.

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.

Using the Cisco IOS XE CLI 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
  MONTH Month of the year
```

Now you can complete the command entry:

```
Router# clock set 13:32:00 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 February ?
<1-31>     Day of the month
Router# clock set 13:32:00 February 23 ?
<1993-2035> Year
```

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

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

Searching and Filtering CLI Output Examples

The following is partial sample output from the **morenvram:startup-config|begin** privileged EXEC mode command that begins unfiltered output with the first line that contains 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
address-family ipv4
 exit-address-family
 !
 address-family ipv6
```

```
 exit-address-family
!
security passwords min-length 1
!
no aaa new-model
ip subnet-zero
no ip domain lookup
ip host sjc-tftp02 171.69.17.17
ip host sjc-tftp01 171.69.17.19
ip host dirt 171.69.1.129
!
!
multilink bundle-name authenticated
!
!
redundancy
 mode sso
!
!
bba-group pppoe global
!
!
interface GigabitEthernet0/0/0
 ip address 10.4.9.158 255.255.255.0
 media-type rj45
 speed 1000
 duplex full
 negotiation auto
 no cdp enable
!
interface GigabitEthernet0/0/1
 no ip address
 media-type rj45
 speed 1000
 duplex full
 negotiation auto
 no cdp enable
!
interface POS0/1/0
 no ip address
 shutdown
 no cdp enable
!
interface POS0/1/1
 no ip address
 shutdown
 no cdp enable
!
interface GigabitEthernet0
 vrf forwarding Mgmt-intf
 no ip address
 speed 1000
 duplex full
 negotiation auto
!
ip default-gateway 10.4.9.1
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 GigabitEthernet0/0/0
ip route 171.69.0.0 255.255.0.0 10.4.9.1
!
no ip http server
no ip http secure-server
!
```

```
!
snmp mib bulkstat schema E0
snmp mib bulkstat schema IFMIB
snmp mib bulkstat transfer 23
snmp mib bulkstat transfer bulkstat1
!
!
control-plane
!
!
line con 0
 exec-timeout 30 0
 logging synchronous
 stopbits 1
line aux 0
 stopbits 1
line vty 0 4
 privilege level 15
 password lab
 login
!
end
```

The following is partial sample output of the **morenvram:startup-config|include** privileged EXEC 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 1192.168.48.48
ip name-server 172.16.2.132
```

The following is partial sample output from the **morenvram:startup-config|exclude**privileged EXEC 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 from the**showinterface** user EXEC or privileged EXEC command mode with an output search specified. The use of the keywords **beginFastEthernet** after the pipe begins

unfiltered output with the first line that contains the regular expression Fast 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 FastEthernet
FastEthernet0/0 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 from the**showbuffers|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
```

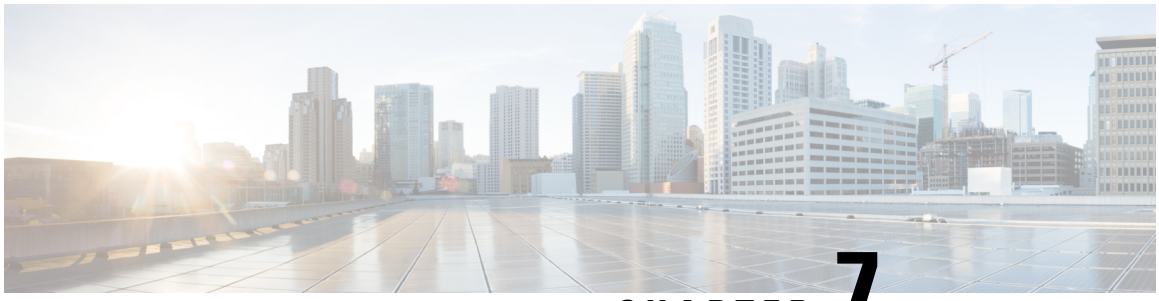
The following is partial sample output from the**showinterface|include**user EXEC or privileged EXEC command mode. The use of the **include(is)** keywords after the pipe (|) causes the command to display only lines that contain the regular expression (is). The parenthesis force the inclusion of the spaces 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
Dialer0/1 is up (spoofing), line protocol is up (spoofing)
```

```
  Hardware is Unknown
  DTR is pulsed for 1 seconds on reset
FastEthernet0/0 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
FastEthernet0/1 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.639c (bia 0060.837c.639c)
  Internet address is 10.5.5.99/24
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 10.0.0.2/8
     0 output errors, 0 collisions, 2 interface resets
  Timeslot(s) Used:14, Transmitter delay is 0 flag
```



CHAPTER 7

show Command Output Redirection

The show Command Output Redirection feature provides the capability to redirect output from Cisco IOS command-line interface (CLI) **show** commands and **more** commands to a file.

Information About show Command Output Redirection

This feature enhances the **show** commands in the Cisco IOS CLI to allow large amounts of data output to be written directly to a file for later reference. This file can be saved on local or remote storage devices such as Flash, a SAN Disk, or an external memory device.

For each **show** command issued, a new file can be created, or the output can be appended to an existing file. Command output can optionally be displayed on-screen while being redirected to a file by using the **tee** keyword. Redirection is available using a pipe (|) character after any **show** command, combined with the followingkeywords:

Output redirection keywords:

Keyword	Usage
append	Append redirected output to URL (URLs supporting append operation only)
begin	Begin with the line that matches
count	Count number of lines which match regexp
exclude	Exclude lines that match
format	Format the output using the specified spec file
include	Include lines that match
redirect	Redirect output to URL
tee	Copy output to URL

These extenstions can also be added to **more** commands.

How to Use the show Command Enhancement

No configuration tasks are associated with this enhancement. For usage guidelines, see the command reference documents listed in the “Related Documents” section.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS configuration commands	*Cisco IOS Configuration Fundamentals Command Reference*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified	--

MIBs

MIB	MIBs Link
• No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

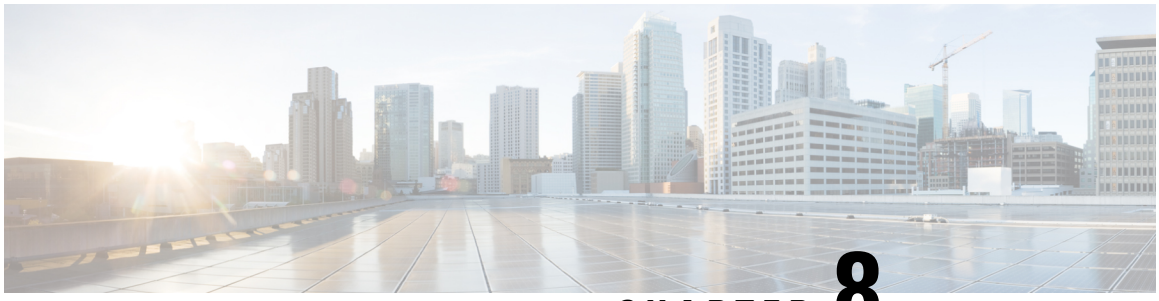
Feature Information for show Command Output Redirection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7: Feature Information for the show Command Ouput Redirection Feature

Feature Name	Releases	Feature Information
show Command Output Redirection	12.0(21)S 12.2(13)T	• The show Command Output Redirection feature provides the capability to redirect output from Cisco IOS command-line interface (CLI) **show** commands and **more** commands to a file. The following commands were introduced or modified: **show**, and**more**.



CHAPTER 8

Overview Basic Configuration of a Cisco Networking Device

Cisco IOS software provides two features, AutoInstall and Setup mode, to simplify configuring a Cisco IOS-based networking device. AutoInstall enables automatic loading of device configuration files from a remote location and can be used to configure several devices concurrently. Setup is an interactive Cisco IOS software command-line interface (CLI) mode that guides you through a basic (also called a startup) configuration but limits you to configuring a single device at a time. AutoInstall is an automatic process for the device that is being configured; Setup is a manual process for the device that is being configured.

This module provides an introduction to each feature and directs you to modules that describe the features in detail and explain how to use them.

The terms initial configuration and startup configuration are used interchangeably.

Prerequisites for Basic Configuration of a Cisco Networking Device

Prerequisites for Cisco IOS AutoInstall

- Using AutoInstall to Remotely Configure Cisco Networking Devices module is written specifically for networking devices running Cisco IOS Release 12.4(1) or newer. However most of the information in this document can be used to configure networking devices that support AutoInstall and are not running Cisco IOS release 12.4(1) or newer. The two key differences that you must allow for are:
  - Some Cisco networking devices use BOOTP instead of DHCP to request IP address addresses over LAN interfaces. Enabling BOOTP support on your DHCP server will resolve this issue.
  - Some Cisco networking devices use a DHCP client identifier format that is different from the format used by networking devices running Cisco IOS release 12.4(1) or newer. This document only explains the DHCP client identifier format used by networking devices running Cisco IOS release

12.4(1) or newer. Use the process described in the "Determining the Value for the DHCP Client Identifier Automatically" section in Using AutoInstall to Remotely Configure Cisco Networking Devices module to determine the DHCP client identifier format that your Cisco networking device is using.

- No configuration file resides in NVRAM on the networking device that is being configured with AutoInstall.
- The configuration files that you want to load on to the networking device using AutoInstall reside on a TFTP server that is connected to the network. In most cases there is more than one file; for example, a network file with the IP-to-hostname mappings and a device-specific configuration file.
- You have someone at the remote site to connect the networking device that is being configured with AutoInstall to the network and power it on.
- The network has the IP connectivity necessary to permit the networking device to load configuration files from the TFTP server during the AutoInstall process.
- A DHCP server is available on the network to provide IP addresses to networking devices that are using AutoInstall over a LAN connection.

Prerequisites for Cisco IOS Setup Mode

- A terminal is connected to the console port of the device being configured.
- You know the interfaces you want to configure.
- You know the routing protocols you want to enable.

For information about routing protocols, see the *Cisco IOS IP Routing Protocols Configuration Guide* .

- You know whether the device you are configuring will perform bridging.
- You know whether the device you are configuring has protocol translation installed.
- You have network addresses for the protocols being configured.

For information about network addresses, see the *Cisco IOS IP Addressing Services Configuration Guide*.

- You have a password strategy for your network environment.

For information about passwords and device security, see "Configuring Security with Passwords, Privilege Levels, and Login User names for CLI Sessions on Networking Devices" in the *Cisco IOS Security Configuration Guide* .

- You have or have access to documentation for the product you want to configure.

Restrictions for Basic Configuration of a Cisco Networking Device

Restrictions for Cisco IOS AutoInstall

- (Serial interfaces only) AutoInstall over a serial interface using either HDLC or Frame Relay can be performed only over the first serial port on a new device (serial interface 0 or serial interface x/0).

- (LAN interfaces only) Only LAN Token Ring interfaces that set ring speed with physical jumpers support AutoInstall.

Restrictions for Cisco IOS Setup Mode

- Setup mode is hardware dependent. You must follow instructions for the specific product you want to configure, as described in documentation for that product.
- Some configuration parameters apply only when a networking device has the protocol translation option. If a device does not have protocol translation, Setup does not prompt for these parameters.

Information About Basic Configuration of a Cisco Networking Device

Before you configure a networking device with a basic configuration, you should understand the following concepts and decide whether AutoInstall or Setup mode is the best method, based on your requirements.

Comparison of Cisco IOS AutoInstall and Cisco IOS Setup Mode

Cisco IOS AutoInstall enables automatic loading of device configuration files from a remote location and can be used to configure several devices concurrently. Setup is an interactive Cisco IOS software CLI mode that guides you through a basic (also called a startup) configuration but limits you to configuring a single device at a time. AutoInstall is an automatic process; Setup is a manual process.

Cisco IOS AutoInstall

AutoInstall is the Cisco IOS software feature that enables the configuration of a remote networking device from a central location. The configuration files must be stored on a TFTP server that is accessible by the devices that you are using AutoInstall to setup.

AutoInstall is supported over Ethernet, Token Ring, and FDDI interfaces for LANs, serial interfaces using High-Level Data Link Control (HDLC) encapsulation, serial interfaces using Frame Relay encapsulation for WANs, and WIC-1-DSU-T1v2 cards (No other T1E1 card supports Autoinstall.).

AutoInstall is designed to facilitate central management of installations at remote sites. The AutoInstall process begins when a Cisco IOS software-based device is turned on and a valid configuration file is not found in NVRAM. AutoInstall may not start if the networking device has Cisco Router and Security Device Manager (SDM) or Cisco Network Assistant already installed. In this case, to enable AutoInstall you need to disable SDM.

Using AutoInstall to Remotely Configure Cisco Networking Devices module describes how AutoInstall functions, how to disable SDM, and how to configure devices to use AutoInstall.

Cisco IOS Setup Mode

Cisco IOS Setup mode enables you to build an initial configuration file using the Cisco IOS CLI or System Configuration Dialog. The dialog guides you through initial configuration and is useful when you are unfamiliar with Cisco products or the CLI and when configuration changes do not require the level of detail the CLI provides.

Setup starts automatically when a device has no configuration file in NVRAM and is not preconfigured from the factory to use Cisco SDM. When setup completes, it presents the System Configuration Dialog. This dialog guides you through an initial configuration with prompts for basic information about your device and network and then creates an initial configuration file. After the file is created, you can use the CLI to perform additional configuration.

Using Setup Mode to Configure a Cisco Networking Device describes how to use Setup to build a basic configuration and to make configuration changes.

Where to Go Next

Proceed to either Using AutoInstall to Remotely Configure Cisco Networking Devices module or Using Setup Mode to Configure a Cisco Networking Device.

Additional References

This section provides references related to the basic configuration of a Cisco networking device.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Configuration fundamentals commands	*Cisco IOS Configuration Fundamentals Command Reference*
Configuring a networking device for the first time using the Cisco IOS software feature AutoInstall.	Using AutoInstall to Remotely Configure Cisco Networking Devices module in *Cisco IOS Configuration Fundamentals Configuration Guide*
Configuring a networking device using Cisco IOS Setup mode	Using Setup Mode to Configure a Cisco Networking Device module in *Cisco IOS Configuration Fundamentals Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Overview Basic Configuration of a Cisco Networking Device

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8: Feature Information for Overview: Basic Configuration of a Cisco Networking Device

Feature Name	Releases	Feature Information
Overview: Basic Configuration of a Cisco Networking Device	12.4(3)	Cisco IOS software provides two features, AutoInstall and Setup mode, to simplify configuring a Cisco IOS-based networking device. AutoInstall enables automatic loading of device configuration files from a remote location and can be used to configure several devices concurrently. Setup is an interactive Cisco IOS software command-line interface (CLI) mode that guides you through a basic (also called a startup) configuration but limits you to configuring a single device at a time. AutoInstall is an automatic process for the device that is being configured; Setup is a manual process for the device that is being configured.

CHAPTER 9

Using AutoInstall to Remotely Configure Cisco Networking Devices

AutoInstall enables remote, automatic configuration of networking devices. AutoInstall is typically used to set up new networking devices remotely. You can, however, use AutoInstall to configure existing networking devices after you remove the configuration file from their NVRAM. The AutoInstall process uses preexisting configuration files that are stored on a TFTP server.

In this module the term networking device means a router that runs Cisco IOS software. Also, the following terms are used interchangeably:

- initial configuration and startup configuration
- *set up* and *configure*

Restrictions

- DHCP server should be reachable via management interface, that is Gigabit Ethernet 0.
- Only Management interface Gigabit Ethernet 0 is supported.

  When you configure this feature on Cisco ASR 1000 Series Aggregation Services Routers replace Ethernet interface used in the document with Gigabit Ethernet interface.

Information About Using AutoInstall to Remotely Configure Cisco Networking Devices

Services and Servers Used by AutoInstall Dynamic Assignment of IP Addresses

The network must be able to provide the dynamic assignment of an IP address to the networking device that is being configured with AutoInstall. The type of IP address assignment server that is used depends on the type of connection that the networking that is being configured with AutoInstall has to the network.

AutoInstall uses these types of IP address servers:

DHCP Servers

Networking devices using AutoInstall over a LAN connection require a DHCP server to provide an IP address dynamically. This requirement applies to Fast Ethernet, Token Ring, and FDDI interfaces. The network must be configured to provide IP connectivity between the DHCP server and any devices that are using AutoInstall over LAN connections.

DHCP (defined in RFC 2131) is an extension of the functionality provided by the Bootstrap Protocol (defined in RFC 951). DHCP provides the framework for passing configuration information to hosts on a TCP/IP network. DHCP adds the capability of automatic allocation of reusable network addresses and additional configuration options such as a router (gateway) IP address, a TFTP server IP address, the name of a boot file to load, and the domain name to use. DHCP servers can be configured on routers, UNIX servers, Microsoft Windows-based servers, and other platforms.

DHCP servers typically assign IP addresses from a pool of IP addresses randomly. It is possible for a device that uses DHCP to obtain its IP address to have a different IP address every time it is connected to the network. This creates a problem for the AutoInstall process when you want to ensure that a particular device is assigned a specific hostname during the AutoInstall process. For example, if you are installing routers on different floors in a remote site and each router is supposed to be assigned a name that indicates its location, such as **ChicagoHQ-1st** and **ChicagoHQ-2nd**, you need to ensure that each device gets the IP address that will be mapped to its correct hostname.

The process of ensuring that a device is assigned a specific IP address is referred to as *creating a reservation*. A reservation is a manually configured relationship between an IP address and a physical layer address of a LAN interface on the device. Many Cisco IOS XE-based devices do not use their MAC address when they request an IP address via DHCP. They use a much longer client identifier instead. Due to the complexity of identifying the client identifier so that you can preconfigure a reservation, and the complexity of finding out if the new device uses its MAC address or the client identifier, we recommend that you allow a new device to obtain an IP address without using a DHCP reservation first in order to discover if the device is using its MAC address or a client identifier. When you have learned how the new device is identifying itself to the DHCP server, you can make a note of the format and create a reservation for it. The next time the new device is rebooted it should obtain the IP address that you reserved to ensure that the new device is assigned the correct hostname. Refer to the information on creating DHCP reservations that was provided with your DHCP server software. The process for creating reservations using Cisco IOS XE based DHCP servers is explained in the Using AutoInstall to Set Up Devices Connected to LANs: Example module. This section includes instructions for identifying the client identifier before the device is connected to the network so that you can preconfigure the DHCP reservations.

Note This document uses a Cisco router as the DHCP server for using AutoInstall to configure LAN-connected networking devices. If you are using a different device as your DHCP server ensure that you have the user documentation for it available in the event that you need help configuring it.

Note There are several configuration parameters such as TFTP server addresses, DNS server addresses, domain names and so on, that can be provided to LAN-connected clients by DHCP servers during the process of assigning IP addresses to clients. These parameters are not required by AutoInstall, therefore they are not included in this document. If you know how to use these parameters you can include them in your DHCP server configuration when you are using AutoInstall to setup your networking devices.

For more information on DHCP services visit the IETF RFC site (http://www.ietf.org/rfc.html) and look for RFCs about DHCP. Most server operating systems support DHCP servers. Refer to the documentation that was provided with your operating system for more information.

SLARP Servers

A router that is being configured with AutoInstall over a serial interface using HDLC encapsulation will send a Serial Line ARP (SLARP) request for an IP address over the serial interface that is connected to the staging router.

The serial interface of the staging router must be configured with an IP address in which the host portion is 1 or 2, such as 192.168.10.1 or 192.168.10.2. The staging router will send a SLARP response to the router that is being configured with AutoInstall that contains the value that the staging router is not using. For example, if the interface on the staging router that is connected to the router that is being configured with AutoInstall is using 192.168.10.1 as its IP address, the staging router will send a SLARP response with a value of 192.168.10.2 to the router that is being configured with AutoInstall.

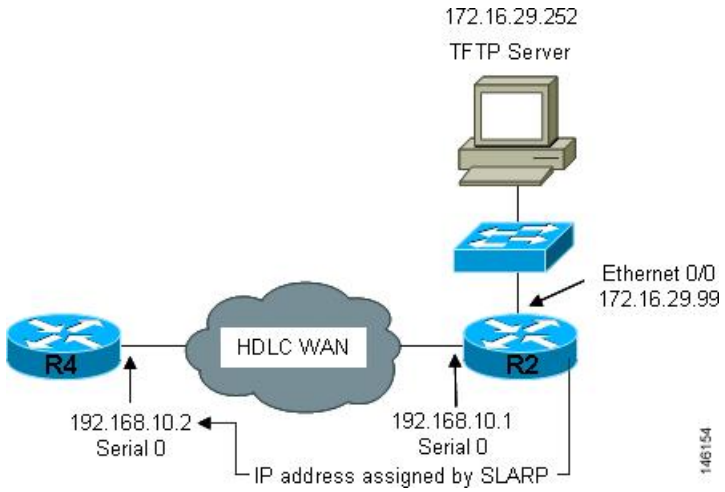
Tip If you are using a mask of 255.255.255.252 on the serial interface of the staging router SLARP will assign the available IP host address to the new device. For example, if you assign IP address 198.162.10.5 255.255.255.252 to serial 0 on the staging router, SLARP will assign 198.162.10.6 to the new device. If you assign IP addresses 198.162.10.6 255.255.255.252 to serial 0 on the staging router SLARP will assign 198.162.10.5 to the new device.

The figure below shows an example of SLARP.

In the figure below, the IP address of serial interface 0 on the staging router (R2) is 192.168.10.1. SLARP therefore assigns the IP address 192.168.10.2 to serial interface 0 on the new device.

Note Replace Ethernet interface used in this figure with Gigabit Ethernet interface, if you plan to use this topology on Cisco ASR 1000 Series Aggregation Services Routers.

Figure 2: Using SLARP to Assign an IP Address to a New Device



Note AutoInstall over a serial interface using HDLC can be performed only over the first serial port on a new device (serial interface 0 or serial interface x/0). The staging router and new device must be directly connected using the first serial interface port on the new device; for example, serial 0/0 or if the first serial port is in the second slot of the device, serial 2/0.

Tip The IP address that is assigned to the router that is being configured with AutoInstall by SLARP from the staging router is the IP address that you must use in the **ip host** *hostname ip-address*command in the AutoInstall network-confg or cisconet.cfg file to ensure that the router that is being configured with AutoInstall is assigned the correct hostname so that it can request its host-specific configuration file.

BOOTP Servers

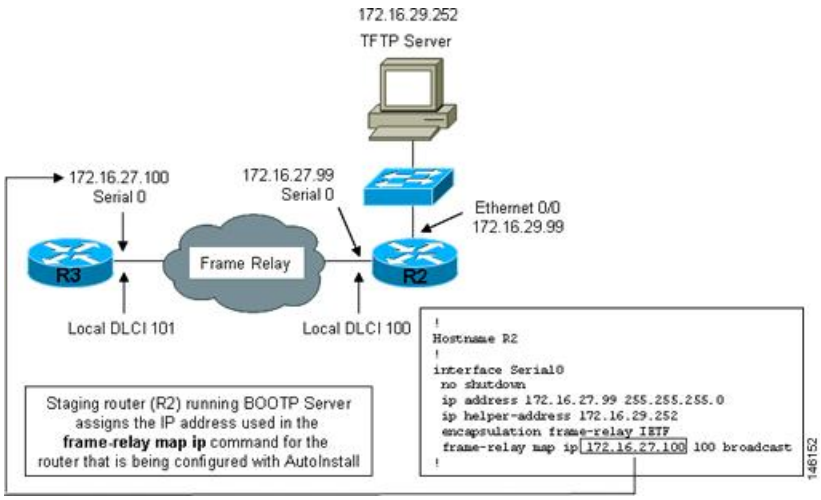
A router that is being configured with AutoInstall over a serial interface using Frame Relay encapsulation will send a BOOTP request for an IP address over the serial interface that is connected to the staging router.

The staging router learns the correct IP address to provide in its BOOTP response to the router that is being configured with AutoInstall by examining the **frame-relay map ip** *ip-address dlci* command that is configured on the interface that it is using to connect to the router that is being configured with AutoInstall.

In the figure below R2 is the staging router. R2 has the **frame-relay map ip 172.16.27.100 100** broadcast command configured on interface serial 0. When R2 receives the BOOTP request for an IP address from R3 during the AutoInstall process, R3 will reply with 172.16.27.100.

Note Replace Ethernet interface used in this figure with Gigabit Ethernet interface, if you plan to use this topology on Cisco ASR 1000 Series Aggregation Services Routers.

Figure 3: Example of Using BOOTP for Autoinstall Over a Frame Relay Network



Tip The limitation imposed by SLARP in which the IP addresses for the new device and the staging router must end in either .1 or .2 does not apply to BOOTP. BOOTP for AutoInstall over Frame Relay supports all host addresses for the IP address subnet that is assigned to the Frame Relay circuit between the router that is being configured with AutoInstall and the staging router.

Tip The IP address that is assigned to the router that is being configured with AutoInstall by BOOTP from the staging router is the IP address that you must use in the **ip host** *hostname ip-address*command in the AutoInstall network-confg or cisconet.cfg file to ensure that the router that is being configured with AutoInstall is assigned the correct hostname so that it can request its host-specific configuration file.

Note AutoInstall over a serial interface using Frame Relay encapsulation can be performed only over the first serial port on a new device (serial interface 0 or serial interface x/0). The staging router and new device must be directly connected using the first serial interface port on the new device; for example, serial 0/0 or if the first serial port is in the second slot of the device, serial 2/0.

Services and Servers Used by AutoInstall IP-to-Hostname Mapping

If you want the networking device to load a full configuration file during the AutoInstall process, the networking device must be able to determine its hostname so that it can request the configuration file that you created specifically for it.

The following caveats apply to the provisioning of IP address to hostname mapping for AutoInstall:

- Any networking device that is being configured with AutoInstall can determine its hostname by loading one of the AutoInstall network configuration files (network-confg or cisconet.cfg) from the TFTP server that contain the **iphost***hostnameip-address* commands. For example, to map host R3 to IP address 198.162.100.3, the network-confg or cisconet.cfg file must contain the **iphostr3198.162.100.3** command.

- A networking device that is being configured with AutoInstall over a LAN interface can also determine its hostname by querying a DNS server. If the DNS server is not connected to the same LAN the device must learn the IP address of the DNS server from the DHCP server during the process of obtaining its dynamically assigned IP address from the DHCP server.

DNS Servers

DNS servers are used to provide a network service that maps hostnames to IP addresses and IP addresses to hostnames (reverse DNS lookups). Anytime that you use a hostname to initiate an IP connection to a host, your PC must determine the IP address that is assigned to the hostname that you want to contact. For example, when you visit Cisco's website (http://www.cisco.com/) your PC sends a DNS query to a DNS server to discover the current IP address that can be used to contact Cisco's website.

For more information on DNS services visit the IETF RFC site (http://www.ietf.org/rfc.html) and look for RFCs about DNS. The Name Server LookUp tool (nslookup) is very useful for learning more about DNS. There are several excellent websites available about nslookup that you can find by searching for them.

Services and Servers Used by AutoInstall Storage and Transmission of Configuration Files

TFTP is a protocol used to transfer files between devices on a network. A TFTP server is a device that uses TFTP to transfer files to devices. TFTP servers can be configured on UNIX servers, Microsoft Windows-based PCs and servers, and other platforms.

Tip If you do not have a TFTP server available you can configure a Cisco IOS-based router as a TFTP server using the **tftp-serverfile-system**:*filename* command. Refer to the Configuring Basic File Transfer Services feature for more information on configuring your router as a TFTP server.

Cisco routers use TFTP to load the configuration files that are required for AutoInstall. You must have a TFTP server deployed in your network to provide file storage and file transmission services to the devices that will be using AutoInstall.

For more information on TFTP services visit the IETF RFC site (http://www.ietf.org/rfc.html) and look for RFCs about TFTP. There are several excellent websites available about TFTP that you can find by searching for them. Several freeware and shareware versions of TFTP servers for various operating systems and hardware platforms are available from the Internet.

The following caveats apply to the provisioning of TFTP servers for AutoInstall:

- Devices using AutoInstall over a LAN--If the TFTP server and the devices using AutoInstall are on different LAN segments, you must either configure the **iphelper-address** *address* command on all of the interfaces that will receive TFTP session initialization requests from the devices that are using AutoInstall.
- Devices using AutoInstall over a WAN--If the devices using AutoInstall are connected to a WAN, you must configure the **iphelper-address** *address* command on all of the interfaces that will receive TFTP session initialization requests from devices that are using AutoInstall.

ip helper-address

If the new device does not learn the IP address of the TFTP server via DHCP option 150, it will transmit the TFTP session initialization requests as network layer broadcasts using the IP destination broadcast address of 255.255.255.255. Routers block network layer broadcast datagrams which prevents the TFTP session initialization requests from reaching the TFTP server, and AutoInstall will fail. The solution to this problem is to use the **ip helper-address** *address* command. The **ip helper-address** *address* command changes the broadcast address of TFTP session initialization request from 255.255.255.255 to the address that is configured with the *address* argument. For example, the **ip helper-address 172.16.29.252** command will change IP destination broadcast address of 255.255.255.255 to 172.16.29.252.

Networking Devices Used by AutoInstall

Device That Is Being Configured with AutoInstall

A device that is being configured with AutoInstall can be any Cisco IOS XE-based router that supports AutoInstall and does not have a configuration file in its NVRAM.

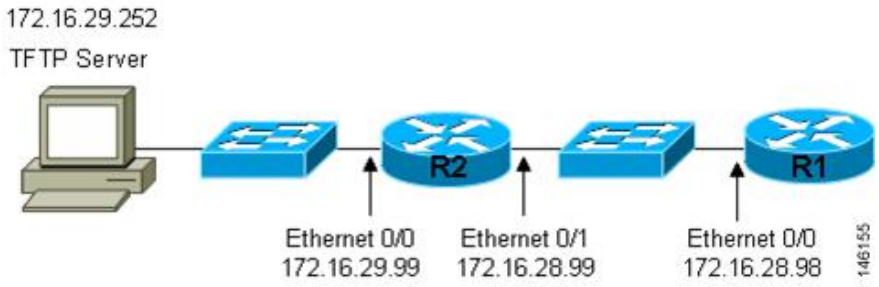
Staging Router

A staging router acts as an intermediary between the TFTP server (to which it must have IP connectivity) and a device that is being configured with AutoInstall when the new device and the TFTP server are connected to different networks. In the figure below R1 requires a staging router because it is connected to a different LAN segment than the TFTP server.

Staging routers are required in the following situations:

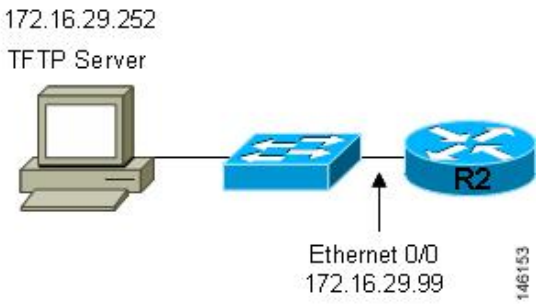
- Devices using AutoInstall over a LAN--If the TFTP and/or DHCP servers and the devices using AutoInstall are on different LAN segments you must use a staging router.
- Devices using AutoInstall over a WAN--If the devices using AutoInstall are connected to a WAN, you must configure the **ip helper-address** *address* command on all of the directly connected interfaces that will receive TFTP session initialization requests from the devices that are using AutoInstall.

Figure 4: Example of AutoInstall That Requires a Staging Router



Staging routers are not required when the new device that is being configured with AutoInstall is connected to the same LAN segment as the TFTP and DHCP servers. In the figure below R2 does not require a staging server to use AutoInstall because it is on the same LAN segment as the TFTP server.

Figure 5: Example of AutoInstall That Does Not Require a Staging Router



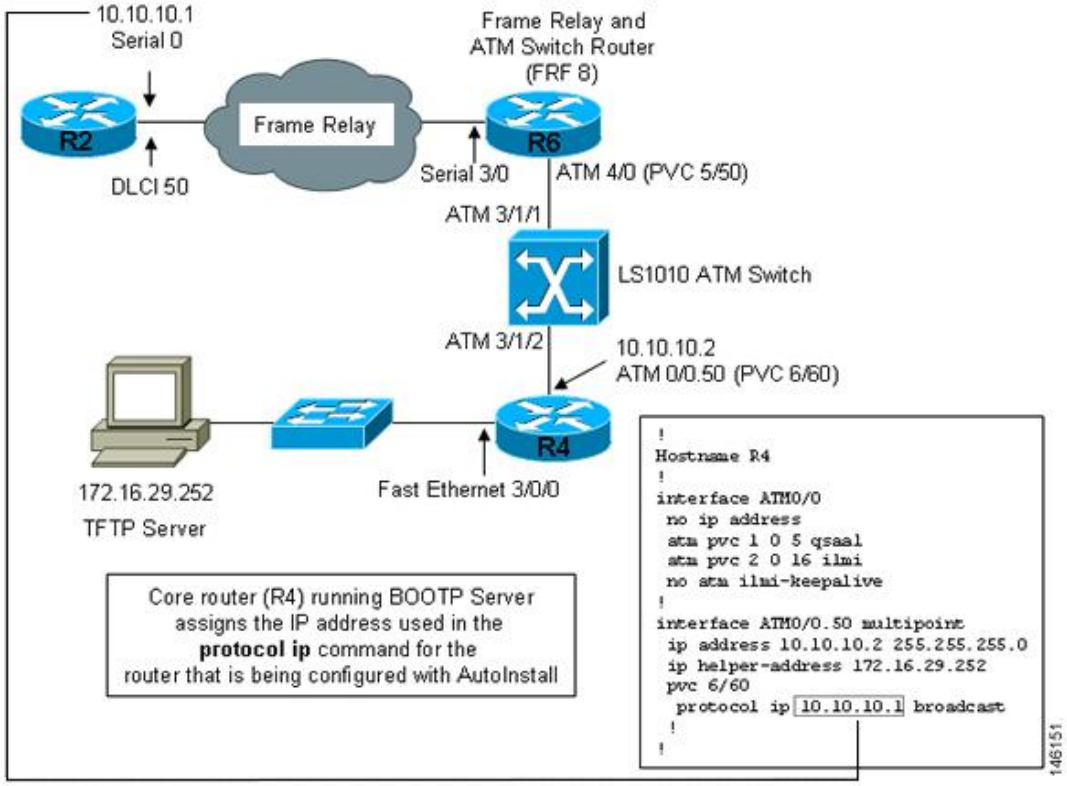
Intermediate Frame Relay-ATM Switching Device

An intermediate Frame Relay-ATM switching device is one that can perform both routing and switching operations. Frame Relay-ATM switching devices are used to connect Frame Relay and ATM networks.

The AutoInstall over Frame Relay-ATM Interworking Connections feature modifies the AutoInstall process to use Frame Relay encapsulation defined by the IETF standard instead of the Frame Relay encapsulation defined by Cisco.

The figure below shows an example topology using AutoInstall over Frame Relay-ATM Interworking Connections. Router R6 does the Frame Relay to ATM Service Internetworking (FRF8) conversion for Frame Relay DLCI 50 to ATM VPI/VCI 5/50. The LS1010 switch routes the VPI/VCI combination used by R6 (5/50) to the VPI/VCI combination used by R4 (6/60).

Figure 6: Example Topology for AutoInstall over Frame Relay-ATM Interworking Connections



Configuration Options for AutoInstall

You can provision your network to support AutoInstall using several different combinations of devices and services. For example:

- You can provision all of the services required for AutoInstall (except dynamic IP address assignment using SLARP or BOOTP that must be preformed by a Cisco router) on one network server, or you can provision each service on a different network server.
- You can provision the DHCP service on a Cisco router.
- The device using AutoInstall can determine its IP address from a DNS server, or you can use one of the AutoInstall network configuration files (network-confg or cisconet.cfg) that contain the **ip host** *hostname ip-address* commands.
- You can use provision AutoInstall to load a full configuration or a partial configuration onto a device that is using AutoInstall.

This module focuses on some of the most common methods for provisioning AutoInstall. Refer to the How to Use AutoInstall to Remotely Configure Cisco Networking Devices module for information on the most common methods for provisioning AutoInstall.

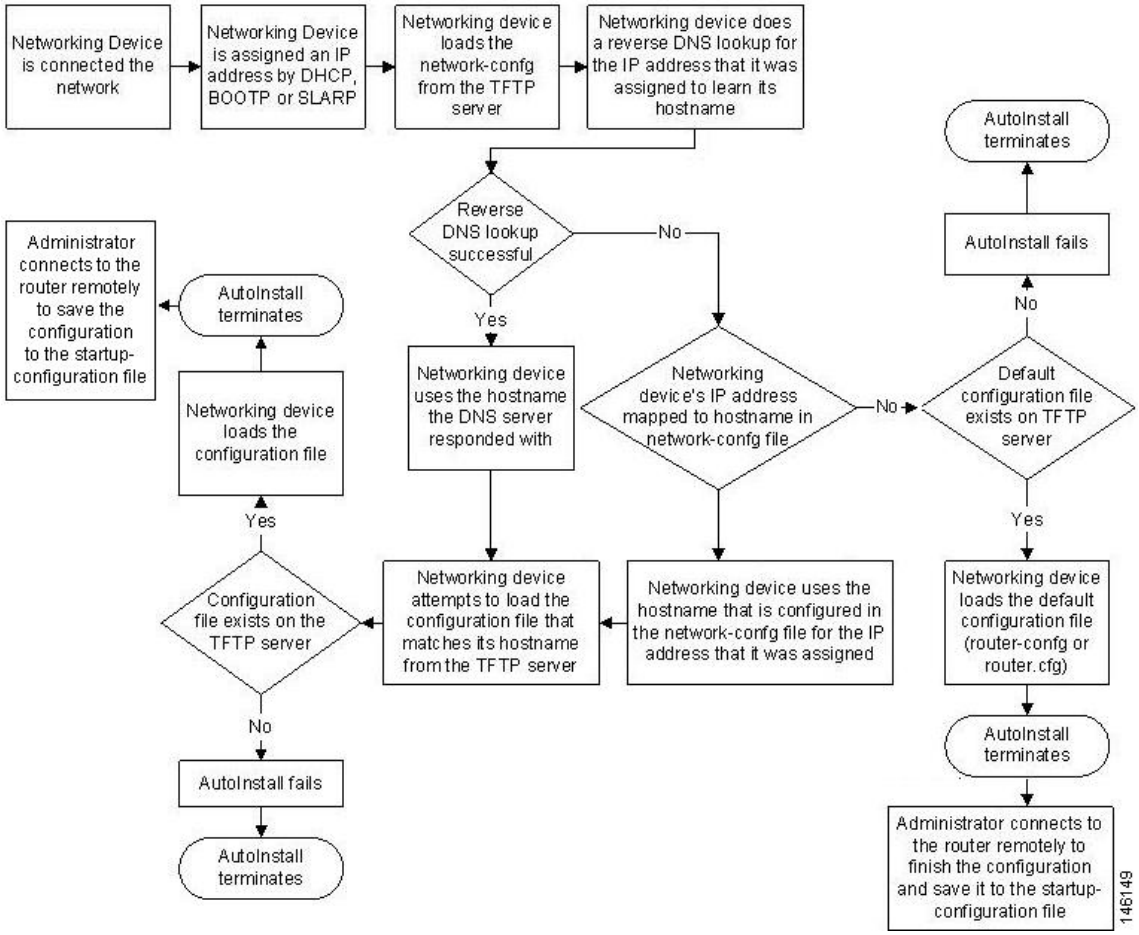
The AutoInstall Process

The AutoInstall process begins when a networking device that does not have any files in its NVRAM is connected to the network.

Timesaver You can decrease the time that the AutoInstall process takes to complete by only connecting the interface on the networking device that you want to use for AutoInstall until the AutoInstall process has finished. For example, if you want the networking device to perform AutoInstall over a WAN interface and you connect its LAN interfaces and its WAN interfaces the networking device will attempt to perform AutoInstall over the LAN interfaces before it attempts to use the WAN interfaces. Leaving the LAN interfaces disconnected until the AutoInstall process is finished causes the networking device to initiate the AutoInstall process over its WAN interface immediately.

The following figure shows the basic flow of the AutoInstall process using the configuration files.

Figure 7: AutoInstall Process Flowchart (Using Configuration Files)



How to Use AutoInstall to Remotely Configure Cisco Networking Devices

This section describes the how to prepare a router for AutoInstall. Additional examples for using AutoInstall for new routers connected to LANs, HDLC WANs, and Frame Relay networks, are provided in the Configuration Examples for Using AutoInstall to Remotely Configure Cisco Networking Devices module.

In most cases you need to configure a staging router through which a new device running AutoInstall sends TFTP, BOOTP, and DNS requests.

Tip In all cases, you must verify and save the configuration on the networking device after the AutoInstall process is complete. If you do not save the configuration, you must repeat the entire process.

Disabling the SDM Default Configuration File

Perform this task if SDM was preinstalled on your device and you want to use Setup to build an initial configuration file. SDM remains on the device.

Perform this task if SDM was pre installed on your device and you want to use AutoInstall to configure the device instead. SDM remains on the device.

SUMMARY STEPS

1. Connect the console cable, shipped with your device, from the console port on the device to a serial port on your PC. Refer to the hardware installation guide for the device for instructions.
2. Connect the power supply to the device, plug the power supply into a power outlet, and turn on the device. Refer to the quick start guide for the device for instructions.
3. Use Hyperterminal or a similar terminal emulation program on your PC, with the following terminal emulation settings, to connect to the device:
4. **enable**
5. **erase startup-config**
6. **reload**

DETAILED STEPS

Step 1 Connect the console cable, shipped with your device, from the console port on the device to a serial port on your PC. Refer to the hardware installation guide for the device for instructions.

Step 2 Connect the power supply to the device, plug the power supply into a power outlet, and turn on the device. Refer to the quick start guide for the device for instructions.

Step 3 Use Hyperterminal or a similar terminal emulation program on your PC, with the following terminal emulation settings, to connect to the device:

- 9600 baud
- 8 data bits, no parity, 1 stop bit

- No flow control

Step 4 **enable**

Enter privileged EXEC mode.

enable

Example:

```
Router> enable
Router#
```

Step 5 **erase startup-config**

Erases the existing configuration in NVRAM.

Example:

```
Router# erase startup-config
```

Step 6 **reload**

Initiates the reload process. The router will initiate the AutoInstall process after it finishes the reload process.

Example:

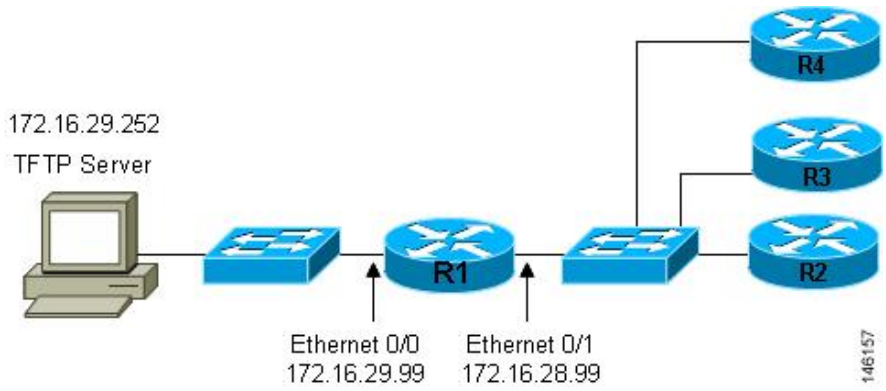
```
Router# reload
```

Configuration Examples for Using AutoInstall to Remotely Configure Cisco Networking Devices

Using AutoInstall to Set Up Devices Connected to LANs Example

This task uses the network in the figure below. This task will show how to use AutoInstall to setup routers R2, R3, and R4. Router R1 is the DHCP server that will be used to assign the IP address for Fast Ethernet 0/0 on the new routers during the AutoInstall process.

Figure 8: Network Topology for Assigning AutoInstall Configuration Files For Specific Devices



Every DHCP client has a unique DHCP client identifier. The DHCP client identifier is used by DHCP servers to keep track of IP address leases and for configuring IP address reservations. You need to know the DHCP client identifier for each of the networking devices that you want to configure with AutoInstall so that you can configure the DHCP IP address reservations which will ensure that each device is provided with the correct IP address, and subsequently its unique configuration file. You can determine the DHCP client identifier manually or automatically.

To use AutoInstall to setup routers R2, R3, and R4, perform following tasks:

Determining the Value for the DHCP Client Identifier Manually

If you want to determine the value for the client identifiers automatically, you do not need to perform this task. Proceed to the Determining the Value for the DHCP Client Identifier Automatically module.

You must know the MAC address of the Fast Ethernet interface that will be used to connect the router to the LAN during the AutoInstall process to determine the client identifier manually. This requires connecting a terminal to the router, and powering it on, so that you can enter the **show interface** *interface-type interface-number* command.

The client-identifier looks like this:

0063.6973.636f.2d30.3030.362e.3533.6237.2e38.6537.312d.4661.332f.30

The format is *nullcisco-0006.53b7.8e71-fa3/0* where *0006.53b7.8e71* is the MAC address and *fa3/0* is the short interface name for the interface that the IP address request is made for.

The values for the short-if-name field can be obtained from an SNMP workstation with the Cisco MIBs installed. This is an example of how to map ifIndex to an interface on Cisco IOS:

```
snmpwalk -c public ponch ifName
IF-MIB::ifName.1 = STRING: AT2/0
IF-MIB::ifName.2 = STRING: Et0/0
IF-MIB::ifName.3 = STRING: Se0/0
IF-MIB::ifName.4 = STRING: BR0/0
```

Use the **show interface** *interface-type interface-number* command to display the information and statistics for a Fast Ethernet interface.

```
R6> show interface fastethernet 3/0
FastEthernet3/0 is up, line protocol is up
  Hardware is AmdFE, address is 0006.53b7.8e71 (bia 0006.53b7.8e71)
```

```
.
.
.
R6>
```

The MAC address for FastEthernet 3/0 on R6 is 0006.53b7.8e71. The format of the client identifier for this interface is nullcisco-0006.53b7.8e71-fa3/0.

Note The short interface name for Fast Ethernet interfaces is fa.

The table below shows the values for converting characters to their hexadecimal equivalents. The last row in the second table below shows the client identifier for Fast Ethernet 3/0 on R6 (nullcisco-0006.53b7.8e71-fa3/0).

Table 9: Hexadecimal to Character Conversion Chart

Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char
00	NUL	1a	SUB	34	4	4e	N	68	h
01	SOH	1b	ESC	35	5	4f	O	69	I
02	STX	1c	FS	36	6	50	P	6a	j
03	ETX	1d	GS	37	7	51	Q	6b	k
04	EOT	1e	RS	38	8	52	R	6c	l
05	ENQ	1f	US	39	9	53	S	6d	m
06	ACK	20		3a	:	54	T	6e	n
07	BEL	21	!	3b	;	55	U	6f	o
08	BS	22	"	3c	<	56	V	70	p
09	TAB	23	#	3d	=	57	W	71	q
0A	LF	24	$	3e	>	58	X	72	r
0B	VT	25	%	3f	?	59	Y	73	s
0C	FF	26	&	40	@	5a	Z	74	t
0D	CR	27	‘	41	A	5b	[	75	u
0E	SO	28	(	42	B	5c	\	76	v
0F	SI	29	)	43	C	5d	]	77	w
10	DLE	2a	*	44	D	5e	^	78	x
11	DC1	2b	+	45	E	5f	_	79	y
12	DC2	2c	,	46	F	60	`	7a	z

Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char
13	DC3	2d	-	47	G	61	a	7b	{
14	DC4	2e	.	48	H	62	b	7c	|
15	NAK	2f	/	49	I	63	c	7D	}
16	SYN	30	0	4a	J	64	d	7e	~
17	ETB	31	1	4b	K	65	e	7f	D
18	CAN	32	2	4c	L	66	f		
19	EM	33	3	4d	M	67	g		

Table 10: Conversion of nullcisco-0006.53b7.8e71-fa3/0 To A Client Identifier

00	c	i	s	c	o	-	0	0	0	6	.	5	3	b	7	.	8	e	7	1	-	f	a	3	/	0
00	63	69	73	63	6f	2d	30	30	30	36	2e	35	33	62	37	2e	38	65	37	31	2d	46	61	33	2f	30

R4

Use the **show interface** *interface-type interface-number* command to display the information and statistics for Fast Ethernet 0/0 on R4.

```
R4> show interface FastEthernet 0/0
FastEthernet0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb0e (bia 00e0.1eb8.eb0e)
```

The MAC address for Fast Ethernet 0/0 on R4 is 00e0.1eb8.eb0e. The format of the client identifier for this interface is nullcisco-00e0.1eb8.eb0e-et0.

Note The short interface name for Fast Ethernet interfaces is et.

Using the values for converting characters to their hexadecimal equivalents in the first table above, the client identifier for Fast Ethernet 0/0 on R4 is shown in the last row of the table below.

Table 11: Conversion of null.cisco-00e0.1eb8.eb0e-et0 To A Client Identifier for R4

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	0	e	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	30	65	2d	45	74	30

R3

Use the **show interface** *interface-type interface-number* command to display the information and statistics for Fast Ethernet 0/0 on R3.

```
R3> show interface FastEthernet 0/0
```

```
FastEthernet0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb73 (bia 00e0.1eb8.eb73)
```

The MAC address for Fast Ethernet 0/0 on R3 is 00e0.1eb8.eb73. The format of the client identifier for this interface is: nullcisco-00e0.1eb8.eb73-et0.

Using the values for converting characters to their hexadecimal equivalents in the first table above, the client identifier for Fast Ethernet 0/0 on R3 is shown in the last row of the table below.

Table 12: Conversion of null.cisco-00e0.1eb8.eb73-et0 To A Client Identifier for R3

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	7	3	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	37	33	2d	45	74	30

R2

Use the **show interface** *interface-type interface-number* command to display the information and statistics for Fast Ethernet 0/0 on R2.

```
R2> show interface Fast Ethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb09 (bia 00e0.1eb8.eb09)
```

The MAC address for Fast Ethernet 0/0 on R2 is 00e0.1eb8.eb09. The format of the client identifier for this interface is nullcisco-00e0.1eb8.eb09-et0.

Using the values for converting characters to their hexadecimal equivalents in the first table above, the client identifier for Fast Ethernet 0/0 on R2 is shown in the last row of the table below

Table 13: Conversion of null.cisco-00e0.1eb8.eb09-et0 To A Client Identifier for R2

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	0	9	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	30	39	2d	45	74	30

You have now determined the values for the client identifiers on each router. The final step is to add a period after each group of four characters working from the left to the right as shown below:

- R4-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30
- R3-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
- R2-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30

Determining the Value for the DHCP Client Identifier Automatically

If you determined the value for the client identifiers manually, you do not need to perform this task. Proceed to the Creating a Private DHCP Pool for Each of The Routers module.

This task will create a DHCP server on R1 that will provide only one IP address. This IP address will used by each new router in sequence while you determine the value of the router's client identifier. By limiting the IP address scope to a single IP address you avoid any possible confusion about which router you are working on. If somebody powers up another router that attempts to start the AutoInstall process, it will not be able to obtain an IP address.

Tip Do not place the network-confg or router configuration files (r4-confg, r3-confg, or r2-confg) in the root directory of the TFTP server yet. You do not want any of the routers to load these files until you have ensured that each router will obtain the correct IP address from the DHCP server so that the router will load the correct configuration file.

This task is broken down into sub-tasks to make it easier to follow (all sub-tasks are required):

Configuring IP on the Interfaces on R1

Configure IP addresses on the Fast Ethernet interfaces. Configure the **ip helper-address** *ip-address* command on Fast Ethernet 0/1.

```
!
interface FastEthernet0/0
 ip address 172.16.29.99 255.255.255.0
!
interface FastEthernet0/1
 ip address 172.16.28.99 255.255.255.0
 ip helper-address 172.16.29.252
!
```

Configuring a DHCP Pool on R1

Configure these commands to setup the temporary DHCP server on R1.

Note This should be the only DHCP server in operation on R1. This should be the only DHCP server that is accessible by the routers that you will be using AutoInstall to setup.

```
ip dhcp excluded-address vrf Mgmt-intf 172.16.28.1 172.16.28.10
ip dhcp pool DHCP_Pool
vrf Mgmt-intf
network 172.16.28.0 255.255.255.0
bootfile ASR-Bootup.cfg
option 150 ip 1.1.1.1
default-router 172.16.28.1
```

Excluding All But One of the IP Addresses from the DHCP Pool on R1

You need to ensure that there is only one IP address available from the DHCP server at any time. Configure the following command to exclude every IP address except 172.16.28.1 from the DHCP pool.

```
!
ip dhcp excluded-address 172.16.28.2 172.16.28.255
!
```

Verifying The Configuration on R1

Verify that the configuration file for R1 has a DHCP server pool configured to provide a single IP address (172.16.28.1) to a DHCP client.

Verify that the configuration file has the IP addresses for the Fast Ethernet interfaces and the **ip helper-address** *ip-address* command.

```
!
ip dhcp excluded-address 172.16.28.2 172.16.28.255
!
ip dhcp pool get-client-id
   network 172.16.28.0 255.255.255.0
!
interface FastEthernet0/0
 ip address 172.16.29.99 255.255.255.0
!
interface FastEthernet0/1
 ip address 172.16.28.99 255.255.255.0
 ip helper-address 172.16.29.252
!
```

Enabling debug ip dhcp server events on R1

You use the display output from the **debug ip dhcp server events** command on the terminal connected to R1 to identify the value of the client identifier for each router.

Enable the **debug ip dhcp server events** command on R1.

```
R1# debug ip dhcp server events
```

Identifying the Value for the Client Identifier on Each of the Routers

This step is repeated for each of the routers. You should only have one of the routers powered-on at any time. When you have identified the value of the client identifier field for the router, you will turn the router off and proceed to the next router.

R4

Connect R4 to the Fast Ethernet network and power it on. The following message will be displayed on the terminal connected to R1 when R4 is assigned the IP address 172.16.28.1.

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30.
```

Copy the client identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30 to a text file and save it. Keep the text file open for the next two routers.

Turn off R4

Release the IP address binding for R4 from the DHCP pool on R1 using the **clear ip dhcp binding *** command on R1.

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

R3

Connect R3 to the Fast Ethernet network and power it on. The following message will be displayed on the terminal connected to R1 when R3 is assigned the IP address 172.16.28.1.

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30.
```

Copy the client identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30 to the text file and save it. Keep the text file open for the final router.

Turn off R3.

Release the IP address binding for R3 from the DHCP pool on R1 using the **clear ip dhcp binding *** command on R1.

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

R2

Connect R2 to the Fast Ethernet network and power it on. The following message will be displayed on the terminal connected to R1 when R2 is assigned the IP address 172.16.28.1.

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30.
```

Copy the client identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30 to the text file and save it.

Turn off R2

Release the IP address binding for R2 from the DHCP pool on R1 using the **clear ip dhcp binding *** command on R1.

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

Client Identifiers for R4, R3, and R2

You have determined the values for the client identifiers on each router.

- R4-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30
- R3-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
- R2-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30

Removing the DHCP Pool on R1 for Network 172.16.28.0 24

The temporary DHCP pool on the router is no longer required, and must be removed.

```
R1(config)# no ip dhcp pool get-client-id
```

Removing the Excluded Address Range From R1

The command for excluding all of the IP addresses except 172.16.28.1 from the DHCP pool on the router is no longer required, and must be removed.

R1(config)# **no ip dhcp excluded-address 172.16.28.2 172.16.28.255**

Creating a Private DHCP Pool for Each of The Routers

You need to create the private DHCP address pools for each router to ensure that each router is assigned the IP address that maps to its host name in the network-conf file.

```
!
ip dhcp pool r4
   host 172.16.28.100 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30
!
ip dhcp pool r3
   host 172.16.28.101 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
!
ip dhcp pool r2
   host 172.16.28.102 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30
```

Creating Configuration Files for Each Router

Create the configuration files for each router and place them in the root directory of the TFTP server.

Tip You must include the commands for configuring passwords for remote Telnet access and access to privileged EXEC mode if you are going to access the routers remotely to save their configuration files to NVRAM.

r2-confg

```
!
hostname R2
!
enable secret 7gD2A0
!
interface FastEthernet0/0
 ip address 172.16.28.102 255.255.255.0
!
interface Serial0/0
 ip address 192.168.100.1 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.5 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0/0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

r3-confg

```
!
hostname R3
!
enable secret 7gD2A0
!
interface FastEthernet0/0
 ip address 172.16.28.101 255.255.255.0
!
interface Serial0/0
 ip address 192.168.100.9 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.13 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

r4-confg

```
!
hostname R3
!
enable secret 7gD2A0
!
interface FastEthernet0/0
 ip address 172.16.28.101 255.255.255.0
!
interface Serial0/0
 ip address 192.168.100.9 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.13 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0/0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

Creating the network-confg file

Create the network-confg file with the **ip host** *hostname ip-address* commands that map the IP addresses that you will be assigning with the DHCP server to the hostname.

```
ip host r4 172.16.28.100
ip host r3 172.16.28.101
ip host r2 172.16.28.102
```

Setting Up the Routers with AutoInstall

You are now ready to set up the three routers (R4, R3, and R2) using AutoInstall.

Connect a terminal to the routers if you want to monitor the progress of AutoInstall. Use Hyperterminal or a similar terminal emulation program on your PC, with the following terminal emulation settings, to connect to the device:

- 9600 baud
- 8 data bits, no parity, 1 stop bit
- No flow control

You should have the following files in the root directory of the TFTP server.

- network-confg
- r4-confg
- r3-confg
- r2-confg

The TFTP server must be running.

Power on each router.

Timesaver You can set up all three routers concurrently.

R4

The following is an excerpt of the messages that are displayed on R4's console terminal during the AutoInstall process:

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.100 to r4
Loading r4-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

R3

The following is an excerpt of the messages that are displayed on R3's console terminal during the AutoInstall process:

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.101 to r3
Loading r3-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

R2

The following is an excerpt of the messages that are displayed on R2's console terminal during the AutoInstall process:

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.102 to r2
Loading r2-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

TFTP Server Log

The TFTP server log should contain messages similar to the following text.

```
Sent network-confg to (172.16.28.100), 76 bytes
Sent r4-confg to (172.16.28.100),687 bytes
Sent network-confg to (172.16.28.101), 76 bytes
Sent r3-confg to (172.16.28.101),687 bytes
Sent network-confg to (172.16.28.102), 76 bytes
Sent r2-confg to (172.16.28.102),687 bytes
```

Saving the Configuration Files on The Routers

You must save the running configurations on each router to the startup configuration to ensure that the routers retain their configurations if they are ever power cycled.

R4

```
R1# telnet 172.16.28.100
Trying 172.16.28.100 ... Open
User Access Verification
Password:
R4> enable
Password:
R4# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R4# exit
[Connection to 172.16.28.100 closed by foreign host]
R1#
```

R3

```
R1# telnet 172.16.28.101
Trying 172.16.28.101 ... Open
User Access Verification
Password:
R3> enable
```

```
Password:
R3# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R3# exit
[Connection to 172.16.28.101 closed by foreign host]
R1#
```

R2

```
R1# telnet 172.16.28.102
Trying 172.16.28.102 ... Open
User Access Verification
Password:
R2> enable
Password:
R2# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R2# exit
[Connection to 172.16.28.102 closed by foreign host]
R1#
```

Removing the Private DHCP Address Pools from R1

The final step in the AutoInstall process is to remove the private DHCP address pools from R1.

```
R1(config)# no ip dhcp pool r4
R1(config)# no ip dhcp pool r3
R1(config)# no ip dhcp pool r2
```

This is the final task, and step for Using AutoInstall to Setup Devices Connected to LANs.

Additional References

This section provides references related to the basic configuration of a Cisco networking device.

Related Documents

Related Topic	Document Title
Configuring a networking device for the first time using the Cisco IOS XE software feature AutoInstall.	Using AutoInstall to Remotely Configure Cisco Networking Devices
Configuring a networking device using Cisco IOS XE Setup mode	Using Setup Mode to Configure a Cisco Networking Device
Configuration fundamentals and associated commands	*Cisco IOS XE Configuration Fundamentals Configuration Guide* for your release and the release-independent Cisco IOS Configuration Fundamentals Command Reference

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

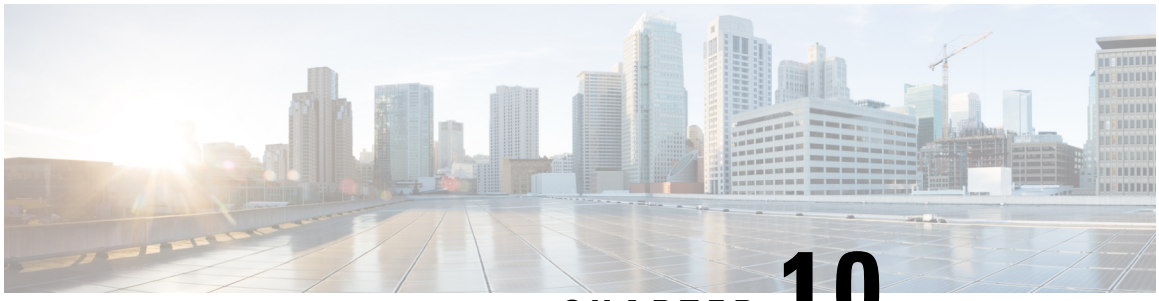
Feature Information for Using AutoInstall to Remotely Configure a Cisco Networking Device

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for Using AutoInstall to Remotely Set Up a Cisco Netowrking Device

Feature Name	Releases	Feature Configuration Information
AutoInstall Using DHCP for LAN Interfaces	Cisco IOS XE Release 2.1	The AutoInstall Using DHCP for LAN Interfaces feature enhances the benefits of AutoInstall by replacing the use of the Bootstrap Protocol (BOOTP) with the use of the Dynamic Host Configuration Protocol (DHCP) for Cisco IOS AutoInstall over LAN interfaces (specifically Fast Ethernet, Token Ring, and FDDI interfaces). In Cisco IOS XE Release 2.1, this feature was introduced on Cisco ASR 1000 Series Routers.
AutoInstall Support for TCL Script	Cisco IOS XE Release 3.3SE	The AutoInstall Using TCL Script feature enhances the AutoInstall feature by providing more flexibility in the installation process. This feature allows the users to program the device to get information about what to download, and to choose the type of file server, and the required file transfer protocol



CHAPTER 10

Unique Device Identifier Retrieval

The Unique Device Identifier Retrieval feature provides the ability to retrieve and display the Unique Device Identifier (UDI) information from any Cisco product that has electronically stored such identity information.

Prerequisites for Unique Device Identifier Retrieval

In order to use UDI retrieval, the Cisco product in use must be UDI-enabled. A UDI-enabled Cisco product supports five required Entity MIB objects. The five Entity MIB v2 (RFC-2737) objects are as follows:

- entPhysicalName
- entPhysicalDescr
- entPhysicalModelName
- entPhysicalHardwareRev
- entPhysicalSerialNum

Although the **show inventory** command may be available, using that command on devices that are not UDI-enabled will likely produce no output.

Information About Unique Device Identifier Retrieval

Unique Device Identifier Overview

Each identifiable product is an entity, as defined by the Entity MIB (RFC-2737) and its supporting documents. Some entities, such as a chassis, will have subentities like slots. A Fast Ethernet switch might be a member of a superentity like a stack. Most Cisco entities that are orderable products will leave the factory with an

assigned UDI. The UDI information is printed on a label that is affixed to the physical hardware device, and it is also stored electronically on the device in order to facilitate remote retrieval.

A UDI consists of the following elements:

- Product identifier (PID)
- Version identifier (VID)
- Serial number (SN)

The PID is the name by which the product can be ordered; it has been historically called the "Product Name" or "Part Number." This is the identifier that one would use to order an exact replacement part.

The VID is the version of the product. Whenever a product has been revised, the VID will be incremented. The VID is incremented according to a rigorous process derived from Telcordia GR-209-CORE, an industry guideline that governs product change notices.

The SN is the vendor-unique serialization of the product. Each manufactured product will carry a unique serial number assigned at the factory, which cannot be changed in the field. This is the means by which to identify an individual, specific instance of a product.

Benefits of the Unique Device Identifier Retrieval Feature

- Identifies individual Cisco products in your networks.
- Reduces operating expenses for asset management through simple, cross-platform, consistent identification of Cisco products.
- Identifies PIDs for replaceable products.
- Facilitates discovery of products subject to recall or revision.
- Automates Cisco product inventory (capital and asset management).
- Provides a mechanism to determine the entitlement level of a Cisco product for repair and replacement service.

How to Retrieve the Unique Device Identifier

Retrieving the Unique Device Identifier

Perform this task to retrieve and display identification information for a Cisco product.

SUMMARY STEPS

1. **enable**
2. **show inventory** [**raw**] [*entity*]

DETAILED STEPS

Step 1 **enable**

Enters privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show inventory** [**raw**] [*entity*]

Enter the **show inventory** command to retrieve and display information about all of the Cisco products installed in the networking device that are assigned a PID, VID, and SN. If a Cisco entity is not assigned a PID, that entity is not retrieved or displayed.

Example:

```
Router# show inventory
NAME: "Chassis", DESCR: "12008/GRP chassis"
PID: GSR8/40           ,  VID: V01,  SN: 63915640
NAME: "slot 0", DESCR: "GRP"
PID: GRP-B             ,  VID: V01,  SN: CAB021300R5
NAME: "slot 1", DESCR: "4 port ATM OC3 multimode"
PID: 4OC3/ATM-MM-SC    ,  VID: V01,  SN: CAB04036GT1
NAME: "slot 3", DESCR: "4 port 0C3 POS multimode"
PID: LC-4OC3/POS-MM    ,  VID: V01,  SN: CAB014900GU
NAME: "slot 5", DESCR: "1 port Gigabit Ethernet"
PID: GE-GBIC-SC-B      ,  VID: V01,  SN: CAB034251NX
NAME: "slot 7", DESCR: "GRP"
PID: GRP-B             ,  VID: V01,  SN: CAB0428AN4O
NAME: "slot 16", DESCR: "GSR 12008 Clock Scheduler Card"
PID: GSR8-CSC/ALRM     ,  VID: V01,  SN: CAB0429AUYH
NAME: "sfslot 1", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0428ALOS
NAME: "sfslot 2", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0429AU0M
NAME: "sfslot 3", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0429ARD7
NAME: "PSslot 1", DESCR: "GSR 12008 AC Power Supply"
PID: FWR-GSR8-AC-B     ,  VID: V01,  SN: CAB041999CW
```

Enter the **show inventory**command with an *entity* argument value to display the UDI information for a specific type of Cisco entity installed in the networking device. In this example, a list of Cisco entities that match the module RO argument string is displayed.

Example:

```
Router# show inventory "module RO"
NAME: ''module R0'', DESCR: ''Cisco ASR1000 Route Processor 2''
PID: ASR1000-RP2 ,  VID: V01,  SN: JAE13041JEX
```

Note The **raw** keyword option is primarily intended for troubleshooting problems with the **show inventory** command itself.

Example:

```
Router# show inventory raw
NAME: "Chassis", DESCR: "12008/GRP chassis"
```

```
PID:                    ,  VID: V01,  SN: 63915640
NAME: "slot 0", DESCR: "GRP"
PID:                    ,  VID: V01,  SN: CAB021300R5
NAME: "slot 1", DESCR: "4 port ATM OC3 multimode"
PID: 4OC3/ATM-MM-SC     ,  VID: V01,  SN: CAB04036GT1
NAME: "slot 3", DESCR: "4 port 0C3 POS multimode"
PID: LC-4OC3/POS-MM     ,  VID: V01,  SN: CAB014900GU
```

Troubleshooting Tips

Commands requiring a delimiting character (the *d* argument) are used throughout this chapter. Any character can be used as the delimiting character, but we recommend the use of the quote sign ("), because this character is unlikely to be needed within the message itself. Other commonly used delimiting characters include the percent sign (%) or the forward slash (/), but because these characters have meanings within certain Cisco IOS commands, they are not recommended. For example, to set the vacant message to This terminal is idle you would enter the command **vacant-message"Thisterminalisidle"**.

Configuration Examples for Unique Device Identifier Retrieval

There are no configuration examples for the UDI Retrieval feature. For sample display output from the **show inventory** command, see the Retrieving the Unique Device Identifier section.

Additional References

This section provides references related to the basic configuration of a Cisco networking device.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Configuration fundamentals commands	*Cisco IOS Configuration Fundamentals Command Reference*
Configuring a networking device for the first time using the Cisco IOS software feature AutoInstall.	Using AutoInstall to Remotely Configure Cisco Networking Devices module in *Cisco IOS Configuration Fundamentals Configuration Guide*
Configuring a networking device using Cisco IOS Setup mode	Using Setup Mode to Configure a Cisco Networking Device module in *Cisco IOS Configuration Fundamentals Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

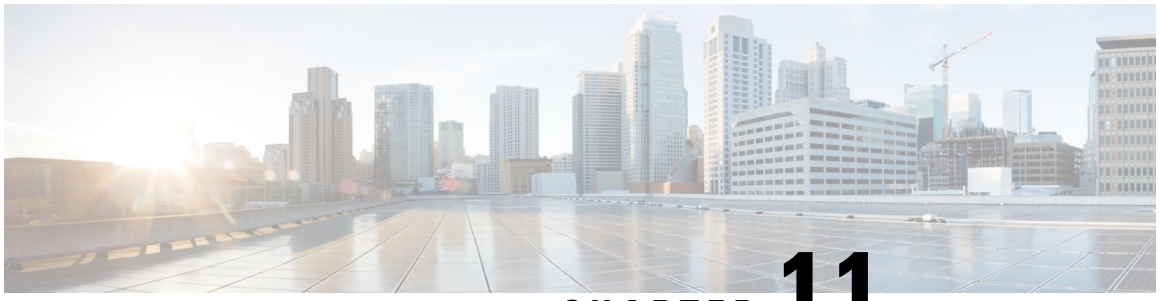
Feature Information for Unique Device Identifier Retrieval

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for Unique Device Identifier Retrieval

Feature Name	Releases	Feature Information
Unique Device Identifier Retrieval	Cisco IOS XE Release 2.1	This feature was introduced.



CHAPTER 11

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 user EXEC or privileged EXEC.

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.

Finding Feature Information

Use Cisco Feature Navigator to find information about platform support and Cisco IOS and Catalyst OS software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

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. The table below lists the keyboard characters that have special meaning.

Table 16: Characters with Special Meaning

Character	Special Meaning
.	Matches any single character, including white space.
*	Matchers 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. Insert 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 inserting a backslash before 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, **telebit3107v32bis** 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. The table below lists the special characters that specify “multiples” of a regular expression.

Table 17: 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 the table below.

Table 18: 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, although {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 number to reuse the remembered pattern. The number 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 number 2), followed by character no. 1 again, followed by character number. 2 again. So, the regular expression can match aZbcTZT. The software remembers that character number 1 is Z and character number 2 is T and then uses Z and T again later in the regular expression.

Searching and Filtering show Commands

To searc h **show** command output, use the following command in privileged EXEC mode:

Command	Purpose
`Router# show` *any-command* `| begin` *regular-expression*	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 privileged EXEC mode:

Command	Purpose
`Router# show` *any-command* `| exclude` *regular-expression*	Displays output lines that do not contain the regular expression.
`Router# show` *any-command* `| include` *regular-expression*	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 privileged EXEC mode. For example, you can enter the **showrunning-config|beginhostname** 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.

Note Characters followed by an exclamation mark (!) or a semicolon (;) are considered as a comment and hence they are ignored in a command.

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 user EXEC mode:

Command	Purpose
`Router# more any-command | begin regular-expression`	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 user EXEC mode:

Command	Purpose
`Router# more any-command | exclude regular-expression`	Displays output lines that do not contain the regular expression.
`Router# more any-command | include regular-expression`	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 user EXEC mode:

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

You can filter output from --More-- prompts. However, you can specify only one filter for each command. The filter remains until the **show** or **more** command output finishes or until you interrupt the output (usingCtrl-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 user EXEC mode:

Command	Purpose
`-More-` **-** *regular-expression*	Displays output lines that do not contain the regular expression.
`-More-` **+** *regular-expression*	Displays output lines that contain the regular expression.

Searching and Filtering CLI Output Examples

The following is partial sample output from the **morenvram:startup-config|begin** privileged EXEC mode command that begins unfiltered output with the first line that contains 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 192.168.48.48
ip name-server 172.16.2.132
!
isdn switch-type primary-5ess
.
.
.
interface Ethernet1
 ip address 10.5.5.99 10.255.255.0
 --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 **morenvram: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 1192.168.48.48
ip name-server 172.16.2.132
```

The following is partial sample output from the **morenvram: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 from the**showinterface** command with an output search specified. The use of the keywords **beginEthernet** 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 from the**showbuffers|exclude**command. It excludes lines that contain the regular expression ip. 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 from the**showinterface|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 10.5.5.99/24
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 10.0.0.2/8
     0 output errors, 0 collisions, 2 interface resets
  Timeslot(s) Used:14, Transmitter delay is 0 flag
```

CHAPTER 12

Consent Token

Consent Token is a security feature that is used to authenticate the network administrator of an organization to access system shell with mutual consent from the network administrator and Cisco Technical Assistance Centre (Cisco TAC).

Restrictions for Consent Token

- Consent Token is enabled by default and cannot be disabled.
- After the challenge has been sent from the device, the response needs to be entered within 30 minutes. If it is not entered, the challenge expires and a new challenge must be requested.
- A single response is valid only for one time for a corresponding challenge.
- The maximum authorization timeout for root-shell access is seven days.
- After a switchover event, all the existing Consent Token based authorizations would be treated as expired. You must then restart a fresh authentication sequence for service access.
- Only Cisco authorized personnel have access to Consent Token response generation on Cisco's challenge signing server.
- In System Shell access scenario, exiting the shell does not terminate authorization until the authorization timeout occurs or the shell authorization is explicitly terminated by the consent token terminate authorization command.

  We recommend that you force terminate System Shell authorization by explicitly issuing the Consent Token terminate command once the purpose of System Shell access is complete.

Information About Consent Token

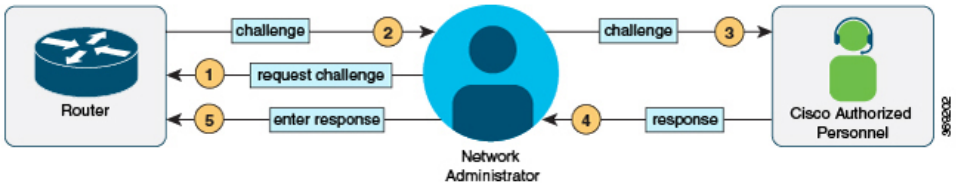
In some debugging scenarios, the Cisco TAC engineer may have to collect certain debug information or perform live debug on a production system. In such cases, the Cisco TAC engineer will ask you (the network administrator) to access system shell on your device. Consent Token is a lock, unlock and re-lock mechanism that provides you with privileged, restricted, and secure access to the system shell.

When you request access to system shell, you need to be authorized. You must first run the command to generate a challenge using the Consent Token feature on your device. The device generates a unique challenge as output. You must then copy this challenge string and send it to a Cisco Authorized Personnel through e-mail or Instant Message.

The Cisco Authorized Personnel processes the unique challenge string and generates a response that is unique. The Cisco Authorized Personnel copies this response string and sends it to you through e-mail or Instant Message.

You must then input this response string into your device. If the challenge-response pair match, you are authorized to access system shell. If not, an error is displayed and you are required to repeat the authentication process.

Once you gain access to system shell, collect the debug information required by the Cisco TAC engineer. After you are done accessing system shell, terminate the session and continue the debugging process.



Consent Token Authorization Process for System Shell Access

This section describes the process of Consent Token authorization to access system shell:

SUMMARY STEPS

1. Generate a challenge requesting for access to system shell for the specified time period.
2. Send the challenge string to a Cisco Authorized Personnel.
3. Input the response string onto your device.
4. Terminate the session.

DETAILED STEPS

Step 1 Generate a challenge requesting for access to system shell for the specified time period.

Example:

```
Device# request consent-token generate-challenge shell-access auth-timeout 900
zSSdrAAAQEBAQAAABgAEAAAAAAMACH86csUmDlOBAQOFvdTCxqRUsDJB4AwWQAEAAAC8GAAhDAEFREANIwAGENUQV9ERUlPXNUQV9ISUdSUSHXOtEWQgAC0M5ODAwLUNMUs5CQAlOVpQLEAESESRkI=
Device#
```

```
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation attempt:
Shell access 0).
```

Send a request for a challenge using the **request consent-token generate-challenge shell-access** *time-validity-slot* command. The duration in minutes for which you are requesting access to system shell is the time-slot-period.

In this example, the time period is 900 minutes after which the session expires.

The device generates a unique challenge as output. This challenge is a base-64 format string.

Step 2 Send the challenge string to a Cisco Authorized Personnel.

Send the challenge string generated by the device to a Cisco Authorized Personnel through e-mail or Instant Message.

The Cisco Authorized Personnel processes the unique challenge string and generates a response. The response is also a base-64 string that is unique. The Cisco Authorized Personnel copies this response string and sends it to you through e-mail or Instant Message.

Step 3 Input the response string onto your device.

Example:

```
Device# request consent-token accept-response shell-access
[redacted]
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success: Shell
access 0).

Device# request platform software system shell
Activity within this shell can jeopardize the functioning of the system.
Are you sure you want to continue? [y/n] y
Device#
*Jan 18 02:56:59.714: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authorization for Shell access 0
 will expire in 10 min).
```

Input the response string sent to you by the Cisco Authorized Personnel using the **request consent-token accept-response shell-access** *response-string* command.

If the challenge-response pair match, you are authorized to access system shell. If the challenge-response pair do not match, an error is displayed and you are required to repeat steps 1 to 3.

After you are authorized, you can access system shell for the requested time-slot.

The device sends a message when there is ten minutes remaining of the authorization session.

Step 4 Terminate the session.

Example:

```
Device# request consent-token terminate-auth
% Consent token authorization termination success

Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate authentication: Shell
access 0).
Device#
```

When you finish accessing system shell, you can end the session using the **request consent-token terminate-auth** command. You can also force terminate the session prior to the authorization timeout using this command. The session also gets terminated automatically when the requested time slot expires.

Dev Key and Release Key

The Cisco IOS XE secure boot functionality ensures that only Cisco signed software is loaded on a Cisco IOS XE platform. Before introducing the dev-key install functionality, Cisco IOS XE platforms are shipped with Dev Public Key and Release Public Key. These keys are used to validate the images signed by corresponding private keys. The subset of Cisco IOS XE platforms which support dev-key install functionality are shipped only with Release Public Key without a Dev Public Key. With this change in the functionality, an image that is signed with a Dev Private Key will not boot due to the absence of Dev Public Key for image verification. However, for some reason, if the Cisco IOS XE device is shipped back to Cisco, a Product Return and Replacement (RMA) specialist may need to load an image signed with Dev Private Key. This requires the RMA specialist to install a Dev Public Key on the device to ensure that the verification of the image signed with Dev Private key passes. To install the Dev Public Key, use the commands mentioned in the following section.

Consent Token Authorization Process for Dev Key Access

This section describes the process of Consent Token authorization to access dev-key:

SUMMARY STEPS

1. Generate a challenge requesting for access to dev-key for the specified time period.
2. Send the challenge string to a Cisco Authorized Personnel.
3. Input the response string onto your device.
4. Terminate the session.

DETAILED STEPS

Step 1 Generate a challenge requesting for access to dev-key for the specified time period.

Example:

```
Device# request consent-token generate-challenge dev-key auth-timeout 900
zSbzAAAQEBAQAAAAAgAEAAAAAAMACH86csUmDlOAAQOFvd7CxqRUeD7B4AW7QAAAAAC8ZAAhDAEFREAMIwaAGNUQ9ERUlPXNUQV9ISUdOSUSHXOtHMQgAC0M5CDAwIUNMUs5CQALOVpQLEAESE5RkI=
Device#
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation attempt: Dev
 key install).
```

Send a request for a challenge using the **request consent-token generate-challenge Dev-key** *time-validity-slot* command. The duration in minutes for which you are requesting access to dev-key is the time-slot-period.

In this example, the time period is 900 minutes after which the session expires.

The device generates a unique challenge as output. This challenge is a base-64 format string.

Note Auth-timeout of zero signifies permanent Dev Public Key installation. Such permanent installation is only allowed on Cisco internal devices for security reasons.

Step 2 Send the challenge string to a Cisco Authorized Personnel.

Send the challenge string generated by the device to a Cisco Authorized Personnel through e-mail or Instant Message.

The Cisco Authorized Personnel processes the unique challenge string and generates a response. The response is also a base-64 string that is unique. The Cisco Authorized Personnel copies this response string and sends it to you through e-mail or Instant Message.

Step 3 Input the response string onto your device.

Example:

```
Device# request consent-token accept-response dev-key
[redacted]
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success: dev key
access 0).
```

Input the response string sent to you by the Cisco Authorized Personnel using the **request consent-token accept-response dev-key** *response-string* command.

If the challenge-response pair match, a Dev Public Key is installed. If the challenge-response pair do not match, an error is displayed and you are required to repeat steps 1 to 3.

The device sends a message when there is ten minutes remaining of the authorization session.

Step 4 Terminate the session.

Example:

```
Device# request consent-token terminate-auth
% Consent token authorization termination success

Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate authentication: Dev key
 install).
Device#
```

This example displays the output when the system fails to terminate the authorization session.

```
Router#request consent-token terminate-auth dev-key
% No in progress authorization, please generate challenge
Router#
```

When you finish accessing dev-key, you can end the session using the **request consent-token terminate-auth** command. You can also force terminate the session prior to the authorization timeout using this command. The session also gets terminated automatically when the requested time slot expires.

Validating the Installation Authorization

To validate the key installation authorization, use the **show platform software consent-token dev-key** command.

```
Router#show platform software consent-token dev-key
Consent token statistics : dev-key
   Instance Id                      : 0
   Authorization remaining (minutes) : Permanent
   Challenge generation requests     : 1
   Challenge response timeouts       : 0
   Authentication success            : 1
   Authentication failure            : 0
```

```
Authentication expiry            : 0
Terminate authentication requests : 0
Challenge generation errors      : 0
```

Enabling or Disabling Consent Token

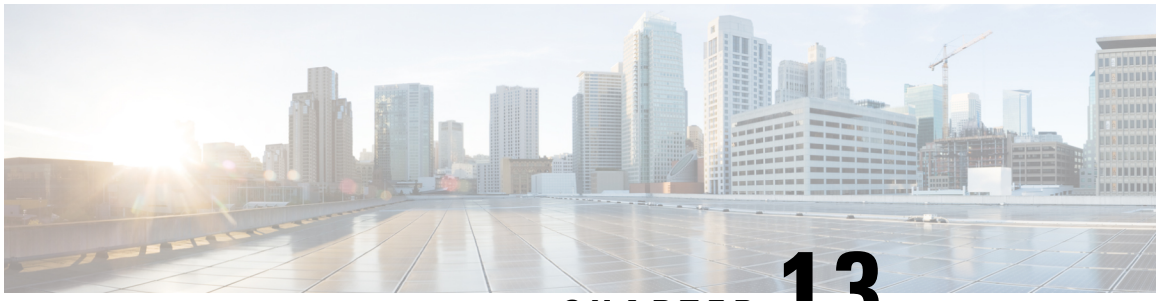
To turn on or turn off the consent token, use the following debug commands:

- **debug platform software consent-token all**
- debug platform software consent-token errors

Feature History and Information for Consent Token

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Release	Feature Information
Cisco IOS XE Gibraltar 16.11.1	This feature was introduced.
Cisco IOS XE Bengaluru 17.4.1	Dev Key and Release Key options were introduced.



CHAPTER 13

Boot Integrity Visibility

Boot integrity visibility allows Cisco's platform identity and software integrity information to be visible and actionable.

Information About Boot Integrity Visibility

Platform identity provides the platform's manufacturing installed identity, and software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code.

During the boot process, the software creates a checksum record of each stage of the boot loader activities.

You can retrieve this record and compare it with a Cisco-certified record to verify if your software image is genuine. If the checksum values do not match, you may be running a software image that is either not certified by Cisco or has been altered by an unauthorized party.

Verifying the software image and hardware

This task describes how to retrieve the checksum record that was created during switch bootup. Enter the following commands in privileged EXEC mode.

Note On executing the following commands, you might see the message **% Please Try After Few Seconds** displayed on the CLI. This does not indicate a CLI failure, but indicates setting up of underlying infrastructure required to get the required output. It is recommended to wait for few minutes and then try the command again.

The messages **% Error retrieving SUDI certificate** and **% Error retrieving integrity data** signify a real CLI failure.

SUMMARY STEPS

1. **show platform sudi certificate** [**sign** [**nonce** *nonce*]]

2. **show platform integrity** [**sign** [**nonce** *nonce*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show platform sudi certificate** [**sign** [**nonce** *nonce*]] **Example:** `# show platform sudi certificate sign nonce 123`	Displays checksum record for the specific SUDI. • (Optional) **sign** - Show signature • (Optional) **nonce** - Enter a nonce value
Step 2	**show platform integrity** [**sign** [**nonce** *nonce*]] **Example:** `# show platform integrity sign nonce 123`	Displays checksum record for boot stages. • (Optional) **sign** - Show signature • (Optional) **nonce** - Enter a nonce value

Verifying Platform Identity and Software Integrity

Verifying Platform Identity

The following example displays the Secure Unique Device Identity (SUDI) chain in PEM format. The first certificate is the Cisco Root CA 2048 and the second is the Cisco subordinate CA (ACT2 SUDI CA). Both certificates can be verified to match those published on https://www.cisco.com/security/pki/. The third is the SUDI certificate.

```
Device#show platform sudi certificate sign nonce 123

-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIQX/h7KCtU3I1CoxW1aMmt/zANBgkqhkiG9w0BAQUFADA1
MRYwFAYDVQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENB
IDIwNDgwHhcNMDQwNTE0MjAxNzEyWhcNMjkwNTE0MjAyNTQyWjA1MRYwFAYDVQQK
Ew1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgwggEg
MA0GCSqGSIb3DQEBAQUAA4IBDQAwggEIAoIBAQCwmrmrp68Kd6ficba0ZmKUeIhH
xmJVhEAyv8CrLqUccda8bnuoqrpu0hWISEWdovyD0My5jOAmaHBKeN8hF570YQXJ
FcjPFto1YYmUQ6iEqDGYeJu5Tm8sUxJszR2tKyS7McQr/4NEb7Y9JHcJ6r8qqB9q
VvYgDxFUl4F1pyXOWWqCZe+36ufijXWLbvLdT6ZeYpzPEApk0E5tzivMW/VgpSdH
jWn0f84bcN5wGyDWbs2mAag8EtKpP6BrXruOIIt6keO1aO6g58QBdKhTCytKmg9l
Eg6CTY5j/e/rmxrbU6YTYK/CfdfHbBcl1HP7R2RQgYCUTOG/rksc35LtLgXfAgED
o1EwTzALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJ/PI
FR5umgIJFq0roIlgX9p7L6owEAYJKwYBBAGCNxUBBAMCAQAwDQYJKoZIhvcNAQEF
BQADggEBAJ2dhISjQal8dwy3U8pORFBi71R803UXHOjgxkhLtv5MOhmBVrBW7hmW
Yqpao2TB9k5UM8Z3/sUcuuVdJcr18JOagxEu5sv4dEX+5wW4q+ffy0vhN4TauYuX
cB7w4ovXsNgOnbFp1iqRe6lJT37mjpXYgyc81WhJDtSd9i7rp77rMKSsH0T8lasz
Bvt9YAretIpjsJyp8qS5UwGH0GikJ3+r/+n6yUA4iGe0OcaEb1fJU9u6ju7AQ7L4
CYNu/2bPPu8Xs1gYJQk0XuPL1hS27PKSb3TkL4Eq1ZKR4OCXPDJoBYVL0fdX4lId
kxpUnwVwwEpxYB5DC2Ae/qPOgRnhCzU=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEPDCCAySgAwIBAgIKYQlufQAAAAAADDANBgkqhkiG9w0BAQUFADA1MRYwFAYD
VQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgw
HhcNMTEwNjMwMTc1NjU3WhcNMjkwNTE0MjAyNTQyWjAnMQ4wDAYDVQQKEwVDaXNj
bzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
```

```
MIIBCgKCAQEA0m5l3THIxA9tN/hS5qR/6UZRpdd+9aE2JbFkNjht6gfHKd477AkS
5XAtUs5oxDYVt/zEbslZq3+LR6qrqKKQVu6JYvH05UYLBqCj38s76NLk53905Wzp
9pRcmRCPuX+a6tHF/qRuOiJ44mdeDYZo3qPCpxzprWJDPclM4iYKHumMQMqmgmg+
xghHIooWS80BOcdiynEbeP5rZ7qRuewKMpl1TiI3WdBNjZjnpfjg66F+P4SaDkGb
BXdGj13oVeF+EyFWLrFjj97fL2+8oauV43Qrvnf3d/GfqXj7ew+z/sXlXtEOjSXJ
URsyMEj53Rdd9tJwHky8neapszS+r+kdVQIDAQABo4IBWjCCAVYwCwYDVR0PBAQD
AgHGMB0GA1UdDgQWBBRI2PHxwnDVW7t8cwmTr7i4MAP4fzAfBgNVHSMEGDAWgBQn
88gVHm6aAgkWrSugiWBf2nsvqjBDBgNVHR8EPDA6MDigNqA0hjJodHRwOi8vd3d3
LmNpc2NvLmNvbS9zZWN1cml0eS9wa2kvY3JsL2NyY2EyMDQ4LmNybDBQBggrBgEF
BQcBAQREMEIwQAYIKwYBBQUHMAKGNGh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3Vy
aXR5L3BraS9jZXJ0cy9jcmNhMjA0OC5jZXIwXAYDVR0gBFUwUzBRBgorBgEEAQkV
AQwAMEMwQQYIKwYBBQUHAgEWNWh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3VyaXR5
L3BraS9wb2xpY2llcy9pbmRleC5odG1sMBIGA1UdEwEB/wQIMAYBAf8CAQAwDQYJ
KoZIhvcNAQEFBQADggEBAGh1qclr9tx4hzWgDERm371yeuEmqcIfi9b9+GbMSJbi
ZHc/CcCl0lJu0a9zTXA9w47H9/t6leduGxb4WeLxcwCiUgvFtCa51Iklt8nNbcKY
/4dw1ex+7amATUQO4QggIE67wVIPu6bgAE3Ja/nRS3xKYSnj8H5TehimBSv6TECi
i5jUhOWryAK4dVo8hCjkjEkzu3ufBTJapnv89g9OE+H3VKM4L+/KdkUO+52djFKn
hyl47d7cZR4DY4LIuFM2P1As8YyjzoNpK/urSRI14WdIlplR1nH7KNDl5618yfVP
0IFJZBGrooCRBjOSwFv8cpWCbmWdPaCQT2nwIjTfY8c=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIDhzCCAm+gAwIBAgIEAJT3DDANBgkqhkiG9w0BAQsFADAnMQ4wDAYDVQQKEwVD
aXNjbzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMB4XDTE1MTExNDA5MzMzN1oXDTI1
MTExNDA5MzMzN1owczEsMCoGA1UEBRMjUElEOldTLUMzNjUwLTEyWDQ4VVEgU046
RkRPMTk0NkJHMDUxDjAMBgNVBAoTBUNpc2NvMRgwFgYDVQQLEw9BQ1QtMiBMaXRl
IFNVREkxGTAXBgNVBAMTEFdTLUMzNjUwLTEyWDQ4VVEwggEiMA0GCSqGSIb3DQEB
AQUAA4IBDwAwggEKAoIBAQC6SARWyImWrRV/x7XQogAE+02WmzKki+4arMVBvl9o
GgvJfkoJDdaHOROSUkEE3qXtd8N3lfKy3TZ+jtHD85m2aGz6+IRx/e/lLsQzi6dl
WIB+N94pgecFBONPR9wJriox1IGD3B43b0hMLkmro4R5Zrs8XFkDo9k1tBU7F2O7
GEzb/WkO5NLexznef2Niglx9fCDL0HC27BbsR5+03p8jhG0+mvrp8M9du1HKiGin
ZIV4XgTMp1/k/TVaIepEGZuWM3hxdUZjkNGG1c1m+oB8vLX3UlSL76sDBBoiaprD
rjXBgBIozyFW8tTjh50jMDG84hKD5s31ifOe4KpqEcnVAgMBAAGjbzBtMA4GA1Ud
DwEB/wQEAwIF4DAMBgNVHRMBAf8EAjAAME0GA1UdEQRGMESgQgYJKwYBBAEJFQID
oDUTM0NoaXBJRD1VWUpOTlZJMENBUkhVM1Z1SUVSbFl5QXlPQ0F4TXpvek5Ub3lN
U0EwS0NnPTANBgkqhkiG9w0BAQsFAAOCAQEADjtM8vdlf+p1WKSKX1C1qQ4aEnD5
p8T5e4iTer7Y1fbCrHIEEm3mnip+568j299z0H8V7PDp1ljuLHyMFTC+945F9RfA
eAuVWVb5A9dnGL8MssBJe2lVSnZwrWkT1EIdxLYrTiPAQHtll6CN77S4u/f71oYE
tzPE5AGfyGw7ro1MEPVGffaQmYUDAwKFNBH1uI7c2S1qlwk4WWZ6xxci+lhaQnIG
pWzapaiAYL1XrcBz4KwFc1ZZpQT6hHw24jzYaYimvCo+/kSKuA9xNdtSu18ycox0
zKnXQ17s6aChMMt7Y8Nh4iz9BDejoOF6/b3sM0wRi+2/4j+6/GhcMRs0Og==
-----END CERTIFICATE-----

Signature version: 1
Signature:
405C770D802B73947EDBF8DD0D2C8180F10D4B3EF9699444514219C579D2ED52F7D5
83E0F4408133FC4E9F549B2EB1C21725F7CB1C79F98271E47E780E703E674723880F
B52D4963E1D1FB9787B38E28B8E696570A180B7A2F1311B1F174EAA79F55DB4765DF
67386126D899E07EDF6C26E0A81272EAA114437DD03F26992937082756AE1F1BFAFB
BFACD6BE9CF9C84C961FACE9FA0FEE64D85AE4FA0086969D0702C536ABDB8FBFDC47
C14C17D02FEBF4F7F5BB24D2932FA876F56B4C07816270AA0B4195C53D975C85AEAE
3A74F2DBF293F52423ECB7B8539667080A9C57DA3E4B08B2B2CA623B2CBAF7080A0A
EB09B222E5B756970A3AA27E0F1D17C8A243
```

The optional RSA 2048 signature is across the three certificates, the signature version and the user-provided nonce

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

Cisco management solutions are equipped with the ability to interpret the above output. However, a simple script using OpenSSL commands can also be used to display the identity of the platform and to verify the signature, thereby ensuring its Cisco unique device identity.

```
[linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:WS-C3650-12X48UQ SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite
SUDI/CN=WS-C3650-12X48UQ
```

Verifying Software Integrity

The following example displays the checksum record for the boot stages. The hash measurements are displayed for each of the three stages of software successively booted. These hashes can be compared against Cisco-provided reference values. An option to sign the output gives a verifier the ability to ensure the output is genuine and is not altered. A nonce can be provided to protect against replay attacks.

```
Device #show platform integrity sign nonce 456

Platform: WS-C3650-12X48UQ
Boot Loader Version: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.16, engineering
software (D)
Boot Loader Hash: DB5A686E9F4CE358481DE3AF8B9C762F0A604E3B4764DF2A351F176E3D7
D3C60EB85C02906BD8CF28228C0DFC2AA8960CAFE6675D696E4ABA0CD687C0609E7E2
Boot 0 Version: F01062R15.0508d68fa2015-09-15
Boot 0 Hash: 6EF15CD54D3C66A8B644194A67B7ED57044C8C2E0EECB69736A7FFEC1F6D0EAD
OS Version: 2016-10-18_10.57_mundru
OS Hash: 4C85AECC88DAA49D940BBF65B1F17269F55C8D98DEFB4140F981923AA961140293E1
3B3E6E68CE3F8ED7F596CD858ACDD4BEF6538F59C1E243C351353026E6CD
PCR0: 90214167AAF35C06B2AC97292596E5669EAB72578FCDAD0B91746683BAA7B2B0
PCR8: FC2CE1BAC397F97008936DF372A2218BB16A798222B8FF55A7B6AEDA8018EDF5
Signature version: 1
Signature:
632A724F1AB6ADE134F6B0E8724D2052B3157F45B47E547763EE224A848E807CD737600587FF68
2526A8FE354A116CC9EDEBD9C659B9927336542EE4295084368327D01BD22AB4849BB3C007B6EB
B67708685FD6BC85DD045431E19A389FEB358894D4FBCF7C0FC960AC9133B61099DFD507F316C1
BF82F7F98687C7E7E8F99355DC1A95BD511B0B8DCB0CA909828F9EFBDF18847930392A8E3D072D
F3D90536880BAE9B7D7CF0E301D3F5AF16E7517FC2700E2F75911B836D6559A18E15B4CF452555
91656DF22DFF73392F777AEB796BCF9AC046C581ADEF19CA48A98F620BB58A79B32DA8B3BFB1CF
8399468A096E2F0C54B8B3ECD15EE3FE2C5ABDB5A029
```

The optional RSA 2048 signature is produced with the SUDI private key and can be verified with the SUDI public key contained in the SUDI certificate. The signature across PCR values, the signature version and the user-provided nonce is displayed.

```
RSA PKCS# 1 v1.5 Sign { <Nonce (UINT64)> || <Signature Version (UINT32)> || <PCR0 (32 bytes)>
 || <PCR8 (32 bytes)> }
```

Cisco management solutions are equipped with the ability to interpret the above output, compare the results against published Cisco values, and to verify the signature.

Feature Information for Boot Integrity Visibility

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

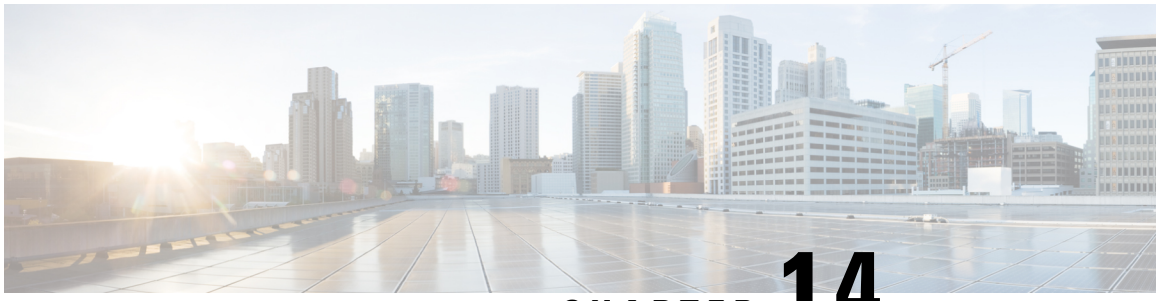
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19: Feature Information for Open Plug-n-Play Agent

Feature Name	Releases	Feature Information
Management and Control: Boot Integrity Visibility	Cisco IOS XE Everest 16.5.1	The Boot Integrity Visibility feature allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity, and software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code. In Cisco IOS XE Everest 16.5.1, support was added for Cisco ASR 1000 Aggregation Series Routers. No commands were introduced or modified for this release.

PART III

Managing Configuration Files



CHAPTER 14

Managing Configuration Files

Creating, loading, and maintaining configuration files enable you to generate a set of user-configured commands to customize the functionality of your Cisco routing device. For a complete description of the configuration file management commands, refer to the *Cisco IOS Configuration Fundamentals Command Reference*.

Prerequisites for Managing Configuration Files

- You should have at least a basic familiarity with the Cisco IOS environment and the command-line interface.
- You should have at least a minimal configuration running on your system. You can create a basic configuration file using the **setup** command (see Using Setup Mode to Configure a Cisco Networking Device for details).

Restrictions for Managing Configuration Files

- Many of the Cisco IOS commands described in this document are available and function only in certain configuration modes on the router.

Information About Managing Configuration Files

Types of Configuration Files

Configuration files contain the Cisco IOS software commands used to customize the functionality of your Cisco routing device (router, access server, switch, and so on). Commands are parsed (translated and executed) by the Cisco IOS software when the system is booted (from the startup-config file) or when you enter commands at the CLI in a configuration mode.

Startup configuration files (startup-config) are used during system startup to configure the software. Running configuration files (running-config) contain the current configuration of the software. The two configuration files can be different. For example, you may want to change the configuration for a short time period rather than permanently. In this case, you would change the running configuration using the **configure terminal** EXEC command but not save the configuration using the **copy running-config startup-config** EXEC command.

To change the running configuration, use the **configure terminal** command, as described in the "Modifying the Configuration File at the CLI " section. As you use the Cisco IOS configuration modes, commands generally are executed immediately and are saved to the running configuration file either immediately after you enter them or when you exit a configuration mode.

To change the startup configuration file, you can either save the running configuration file to the startup configuration using the **copy running-config startup-config** EXEC command or copy a configuration file from a file server to the startup configuration (see the "Copying a Configuration File from a TFTP Server to the Router " section for more information).

Configuration Mode and Selecting a Configuration Source

To enter configuration mode on the router, enter the **configure**command at the privileged EXEC prompt. The Cisco IOS software responds with the following prompt asking you to specify the terminal, memory, or a file stored on a network server (network) as the source of configuration commands:

```
Configuring from terminal, memory, or network [terminal]?
```

Configuring from the terminal allows you to enter configuration commands at the command line, as described in the following section. Configuring from memory loads the startup configuration file. See the "Reexecuting the Configuration Commands in the Startup Configuration File " section for more information. Configuring from the network allows you to load and execute configuration commands over the network. See the "Copying a Configuration File from a TFTP Server to the Router " section for more information.

Configuration File Changes Using the CLI

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the**show running-config or more system:running-config** EXEC command. Comments do not display when you list the startup configuration with the **show startup-config** or **more nvram:startup-config** EXEC mode command. Comments are stripped out of the configuration file when it is loaded onto the router. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), remote copy protocol (rcp), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them.

Location of Configuration Files

Configuration files are stored in the following locations:

- The running configuration is stored in RAM.
- On all platforms except the Class A Flash file system platforms, the startup configuration is stored in nonvolatile random-access memory (NVRAM).

- On Class A Flash file system platforms, the startup configuration is stored in the location specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM and can be a file in the following file systems:
  - **nvram:** (NVRAM)

Copy Configuration Files from a Network Server to the Router

You can copy configuration files from a TFTP, rcp, or FTP server to the running configuration or startup configuration of the router. You may want to perform this function for one of the following reasons:

- To restore a backed-up configuration file.
- To use the configuration file for another router. For example, you may add another router to your network and want it to have a similar configuration to the original router. By copying the file to the new router, you can change the relevant parts rather than re-creating the whole file.
- To load the same configuration commands on to all the routers in your network so that all the routers have similar configurations.

The **copy**{**ftp:** | **rcp:** | **tftp:system:running-config**EXECcommand loads the configuration files into the router as if you were typing the commands in at the command line. The router does not erase the existing running configuration before adding the commands. If a command in the copied configuration file replaces a command in the existing configuration file, the existing command will be erased. For example, if the copied configuration file contains a different IP address in a particular command than the existing configuration, the IP address in the copied configuration will be used. However, some commands in the existing configuration may not be replaced or negated. In this case, the resulting configuration file will be a mixture of the existing configuration file and the copied configuration file, with the copied configuration file having precedence.

In order to restore a configuration file to an exact copy of a file stored on a server, you need to copy the configuration file directly to the startup configuration (using the **copyftp:**| **rcp:**| **tftp:**} **nvram:startup-config**command) and reload the router.

To copy configuration files from a server to a router, perform the tasks described in the following sections:

The protocol you use depends on which type of server you are using. The FTP and rcp transport mechanisms provide faster performance and more reliable delivery of data than TFTP. These improvements are possible because the FTP and rcp transport mechanisms are built on and use the TCP/IP stack, which is connection-oriented.

Copy a Configuration File from the Router to a TFTP Server

In some implementations of TFTP, you must create a dummy file on the TFTP server and give it read, write, and execute permissions before copying a file over it. Refer to your TFTP documentation for more information.

Copy a Configuration File from the Router to an FTP Server

You can copy a configuration file from the router to an FTP server.

Understanding the FTP Username and Password

The FTP protocol requires a client to send a remote username and password on each FTP request to a server. When you copy a configuration file from the router to a server using FTP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip ftp username** global configuration command, if the command is configured.
3. Anonymous.

The router sends the first valid password it encounters in the following sequence:

1. The password specified in the **copy** command, if a password is specified.
2. The password set by the **ip ftp password** command, if the command is configured.
3. The router forms a password *username @routername .domain* . The variable *username* is the username associated with the current session, *routername* is the configured host name, and *domain* is the domain of the router.

The username and password must be associated with an account on the FTP server. If you are writing to the server, the FTP server must be properly configured to accept the FTP write request from the user on the router.

If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user name as the remote username.

Refer to the documentation for your FTP server for more information.

Use the **ip ftp username** and **ip ftp password** global configuration commands to specify a username and password for all copies. Include the username in the **copy** EXEC command if you want to specify a username for that copy operation only.

Note The password must not contain the special characters '@', ':', and '/'. If these special characters are used, the copy fails to parse the IP address of the server.

Copying Files Through a VRF

You can copy files through a VRF interface specified in the **copy** command. Specifying the VRF in the **copy** command is easier and more efficient because you can directly change the source interface without having the need to change the configuration via a change request.

The following example shows how to copy files through a VRF, using the **copy** command:

```
Device# copy scp: slot0: vrf test-vrf
Device# copy scp: slot0: vrf test-vrf
Address or name of remote host [10.1.2.3]?
Source username [ScpUser]?
Source filename [/auto/tftp-server/ScpUser/vrf_test.txt]?
Destination filename [vrf_test.txt]?
Getting the vrf name as test-vrf
Password:
Sending file modes: C0644 10 vrf_test.txt
!
223 bytes copied in 22.740 secs (10 bytes/sec)
```

Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds size of NVRAM, you should be aware of the information in the following sections:

Compressing the Configuration File

The **servicecompress-config** global configuration command specifies that the configuration file be stored compressed in NVRAM. Once the configuration file has been compressed, the router functions normally. When the system is booted, it recognizes that the configuration file is compressed, expands it, and proceeds normally. The **morenvram:startup-config** EXEC command expands the configuration before displaying it.

Before you compress configuration files, refer to the appropriate hardware installation and maintenance publication. Verify that your system's ROMs support file compression. If not, you can install new ROMs that support file compression.

The size of the configuration must not exceed three times the NVRAM size. For a 128-KB size NVRAM, the largest expanded configuration file size is 384 KB.

The **servicecompress-config**global configurationcommand works only if you have Cisco IOS software Release 10 or later release boot ROMs. Installing new ROMs is a one-time operation and is necessary only if you do not already have Cisco IOS Release 10 in ROM. If the boot ROMs do not recognize a compressed configuration, the following message is displayed:

```
Boot ROMs do not support NVRAM compression Config NOT written to NVRAM
```

Loading the Configuration Commands from the Network

You can also store large configurations on FTP or TFTP servers and download them at system startup. To use a network server to store large configura tions, see the "Copying a Configuration File from the Router to a TFTP Server" and "Configure the Router to Download Configuration Files" sections for more information on these commands.

Control of the Parser Cache

The Cisco IOS command-line parser in the Cisco IOS software performs the translation and execution (parsing) of command lines. The Parser Cache feature was developed to rapidly process large configuration files, thereby dramatically improving load time.

The Parser Cache feature allows the rapid recognition and translation of configuration lines in a configuration file that differ slightly from previously used configuration lines (for example, pvc 0/100, pvc 0/101, and so on) by dynamically creating, caching, and reusing simplified parse graphs. This improvement is useful primarily for configuration files that repeat similar commands hundreds or thousands of times, such as cases in which thousands of virtual circuits must be configured for subinterfaces, or hundreds of access lists must be configured. Performance will improve the most for those files in which the same commands are used repeatedly but the numerical arguments change from command to command.

The Parser Cache is enabled by default on all platforms using Cisco IOS Release 12.1(5)T and later releases. However, users with Cisco devices that do not require large configuration files may want to disable the Parser Cache to free the resources used by this feature. (Memory used by this feature depends on the size of the configuration files parsed, but is generally less than 512 KB.)

There are several ways to control the parser cache (these are all optional):

- Clearing the Parser Cache--To free resources or to reset the parser cache memory, you may wish to clear the parse entries and hit/miss statistics stored by the Parser Cache feature
- Disabling the Parser Cache--The Parser Cache feature is enabled by default. To disable the Parser Cache feature, use the no parser cache command in global configuration mode. When the parser cache is disabled, the **noparsercache** command line is written to the running configuration file. If you wish to disable the parser cache to free system resources, you should clear the parser cache before issuing the **noparsercache** command. You will not be able to clear the parser cache after disabling it.
- Reenabling the Parser Cache--To reenable the Parser Cache feature after disabling it, use the **parsercache** command in global configuration mode
- Monitoring the Parser--Statistics about the last configuration file parsed are kept in the system memory, along with hit/miss statistics on the commands parsed by the Parser Cache feature. “Hits” and “misses” refer to the matches that the parser cache was able to make to similar commands used previously in the configuration session. Those commands that are matched (“hits”) be parsed more efficiently. The parser cache cannot improve the parse time for those commands it was unable to match (“misses”).

Configure the Router to Download Configuration Files

You can configure the router to load one or two configuration files at system startup. The configuration files are loaded into memory and read in as if you were typing the commands at the command line. Thus, the configuration for the router will be a mixture of the original startup configuration and the one or two downloaded configuration files.

Network Versus Host Configuration Files

For historical reasons, the first file the router downloads is called the network configuration file. The second file the router downloads is called the host configuration file. Two configuration files can be used when all of the routers on a network use many of the same commands. The network configuration file contains the standard commands used to configure all of the routers. The host configuration files contain the commands specific to one particular host. If you are loading two configuration files, the host configuration file should be the configuration file you want to have precedence over the other file. Both the network and host configuration files must reside on a network server reachable via TFTP, rcp, or FTP, and must be readable.

How to Manage Configuration File Information

Displaying Configuration File Information

To display information about configuration files, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **show boot**
3. **more** *file-url*
4. **show running-config**
5. **show startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show boot** **Example:** `Device# show boot`	Lists the contents of the BOOT environment variable, the name of the configuration file pointed to by the CONFIG_FILE environment variable, and the contents of the BOOTLDR environment variable.
Step 3	**more** *file-url* **Example:** `Device# more 10.1.1.1`	Displays the contents of a specified file.
Step 4	**show running-config** **Example:** `Device# show running-config`	Displays the contents of the running configuration file. (Command alias for the **more system:running-config** command.)
Step 5	**show startup-config** **Example:** `Device# show startup-config`	Displays the contents of the startup configuration file. (Command alias for the **more nvram:startup-config** command.) On all platforms except the Class A Flash file system platforms, the default startup-config file usually is stored in NVRAM. On the Class A Flash file system platforms, the CONFIG_FILE environment variable points to the default startup-config file. The CONFIG_FILE variable defaults to NVRAM.

Modifying the Configuration File at the CLI

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the**show running-config or more system:running-config** EXEC command. Comments do not display when you list the startup configuration with the **showstartup-config** or **more nvram:startup-config** EXEC mode command. Comments are stripped out of the configuration file when it is loaded onto the router. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), remote copy protocol (rcp), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them. To configure the software using the CLI, use the following commands beginning in privileged EXEC mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **end**
   - **^Z**
4. **copy system:running-config nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode. Enter the necessary configuration commands. The Cisco IOS documentation set describes configuration commands organized by technology.
Step 3	Do one of the following: • **end** • **^Z** **Example:** `Device(config)# end`	Ends the configuration session and exits to EXEC mode. **Note** When you press the Ctrl and Z keys simultaneously, ^Z is displayed to the screen.
Step 4	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration file as the startup configuration file. You may also use the **copy running-config startup-config** command alias, but you should be aware that this command is less precise. On most platforms, this command saves the configuration to NVRAM. On the Class A Flash file system platforms, this step saves the configuration to the location specified by the CONFIG_FILE environment variable (the default CONFIG_FILE variable specifies that the file should be saved to NVRAM).

Examples

In the following example, the Device prompt name of the Device is configured. The comment line, indicated by the exclamation mark (!), does not execute any command. The **hostname** command is used to change the Device name from Device to new_name. By pressing Ctrl-Z (^Z) or entering the **end**command, the user quits configuration mode. The **copy system:running-config nvram:startup-config** command saves the current configuration to the startup configuration.

```
Device# configure terminal
Device(config)# !The following command provides the Device host name.
Device(config)# hostname new_name
new_name(config)# end
new_name# copy system:running-config nvram:startup-config
```

When the startup configuration is NVRAM, it stores the current configuration information in text format as configuration commands, recording only nondefault settings. The memory is checksummed to guard against corrupted data.

Note Some specific commands might not get saved to NVRAM. You will need to enter these commands again if you reboot the machine. These commands are noted in the documentation. We recommend that you keep a list of these settings so that you can quickly reconfigure your Device after rebooting.

Copying a Configuration File from the Router to a TFTP Server

To copy configuration information on a TFTP network server, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*]
3. **copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy system:running-config tftp: //server1/topdir/file10`	Copies the running configuration file to a TFTP server.
Step 3	**copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy nvram:startup-config tftp: //server1/1stdir/file10`	Copies the startup configuration file to a TFTP server.

Examples

The following example copies a configuration file from a Device to a TFTP server:

```
Tokyo# copy system:running-config tftp://172.16.2.155/tokyo-confg
Write file tokyo-confg on host 172.16.2.155? [confirm] y
Writing tokyo-confg!!! [OK]
```

What to Do Next

After you have issued the **copy** command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Copying a Configuration File from the Router to the FTP Server

To copy a startup configuration file or a running configuration file from the router to an FTP server, complete the following tasks:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*
4. **ip ftp password** *password*
5. **end**
6. Do one of the following:
   - **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]/*filename*]
   -
   -
   - **copy nvram:sta rtup-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]/*filename*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username user1`	(Optional) Specifies the default remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp username guessme`	(Optional) Specifies the default password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).
Step 6	Do one of the following: • **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]/*filename*] • • • **copy nvram:sta rtup-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]/*filename*] **Example:** `Device# copy system:running-config ftp: //user1:guessme@company.com /dir10/file1`	Copies the running configuration or startup configuration file to an FTP server.

Examples

Storing a Running Configuration File on an FTP Server

The following example copies the running configuration file named rtr2-confg to the netadmin1 directory on the remote host with an IP address of 172.16.101.101:

```
Device# copy system:running-config ftp://netadmin1:mypass@172.16.101.101/Rtr2-confg
Write file rtr2-confg on host 172.16.101.101?[confirm]
Building configuration...[OK]
Connected to 172.16.101.101
Device#
```

Storing a Startup Configuration File on an FTP Server

The following example shows how to store a startup configuration file on a server by using FTP to copy the file:

```
Rtr2# configure terminal

Rtr2(config)# ip ftp username netadmin2
```

```
Rtr2(config)# ip ftp password mypass

Rtr2(config)# end

Rtr2# copy nvram:startup-config ftp:

Remote host[]? 172.16.101.101

Name of configuration file to write [rtr2-confg]?
Write file rtr2-confg on host 172.16.101.101?[confirm]
![OK]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Copying a Configuration File from a TFTP Server to the Router

To copy a configuration file from a TFTP server to the Device, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **copy tftp:** [[[*//location*]*/directory*]*/filename*] **system:running-config**
3. **copy tftp:** [[[*//location*]*/directory*]*/filename*] **nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **system:running-config** **Example:** `Device# copy tftp://server1/dir10/datasource system:running-config`	Copies a configuration file from a TFTP server to the running configuration.
Step 3	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **nvram:startup-config** **Example:** `Device# copy tftp://server1/dir10/datasource nvram:startup-config`	Copies a configuration file from a TFTP server to the startup configuration.

Examples

In the following example, the software is configured from the file named tokyo-config at IP address 172.16.2.155:

```
Device1# copy tftp://172.16.2.155/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Copying a Configuration File from an FTP Server to the Router

To copy a configuration file from an F TP server to the running configuration or startup configuration, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*
4. **ip ftp password** *password*
5. **end**
6. **copy ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*]**system:running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Allows you to enter global configuration mode. This step is required only if you want to override the default remote username or password (see Steps 2 and 3).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username user1`	(Optional) Specifies the default remote username.

	Command or Action	Purpose
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password guessme`	(Optional) Specifies the default password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).
Step 6	**copy ftp:** [[[*//*[*username* [**:***password*]**@**]*location*]*/directory*]*/filename*]**system:running-config** **Example:** **Example:** `or` **Example:** **copy ftp:**[[[**//**[*username* [**:***password*]**@**]*location***/***directory*]**/***filename*] **nvram:startup-config** **Example:** `Device# copy ftp://user1:guessme@company.com /dir10/datasource nvram:startup-config`	Using FTP, copies the configuration file from a network server to running memory or the startup configuration.

Examples

Copy FTP Running-Config

The following example copies a host configuration file named host1-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101, and loads and runs those commands on the Device:

```
Device# copy rcp://netadmin1:mypass@172.16.101.101/host1-confg system:running-config

Configure using host1-confg from 172.16.101.101? [confirm]
Connected to 172.16.101.101
Loading 1112 byte file host1-confg:![OK]
Device#
%SYS-5-CONFIG: Configured from host1-config by ftp from 172.16.101.101
```

Copy FTP Startup-Config

The following example specifies a remote username of netadmin1. Then it copies the configuration file named host2-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101 to the startup configuration.

```
Rtr2# configure terminal
Rtr2(config)# ip ftp username
 netadmin1
Rtr2(config)# ip ftp password
 mypass
Rtr2(config)# end
Rtr2# copy ftp: nvram:startup-config
Address of remote host [255.255.255.255]? 172.16.101.101
Name of configuration file[rtr2-confg]? host2-confg
Configure using host2-confg from 172.16.101.101?[confirm]
Connected to 172.16.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
Rtr2#
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by ftp from 172.16.101.101
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Maintaining Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds size of NVRAM, perform the tasks described in the following sections:

Compressing the Configuration File

To compress configuration files, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service compress-config**
4. **end**
5. Do one of the following:
   - Use FTP, rcp, or TFTP to copy the new configuration.
   - **configure terminal**
6. **copy system:running-config nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service compress-config** **Example:** `Device(config)# service compress-config`	Specifies that the configuration file be compressed.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 5	Do one of the following: • Use FTP, rcp, or TFTP to copy the new configuration. • **configure terminal** **Example:** `Device# configure terminal`	Enters the new configuration: • If you try to load a configuration that is more than three times larger than the NVRAM size, the following error message is displayed: "[buffer overflow - *file-size* /*buffer-size* bytes]."
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	When you have finished changing the running-configuration, saves the new configuration.

Examples

The following example compresses a 129-KB configuration file to 11 KB:

```
Device# configure terminal

Device(config)# service compress-config

Device(config)# end

Device# copy tftp://172.16.2.15/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
Device# copy system:running-config nvram:startup-config

Building configuration...
Compressing configuration from 129648 bytes to 11077 bytes
[OK]
```

Managing the Parser Cache

To control the Parser Cache feature, perform the tasks described in the following sections. All of these tasks are optional:

Clearing the Parser Cache

To clear the information stored by the Parser Cache feature, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **clear parser cache**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear parser cache** **Example:** `Device# clear parser cache`	Clears the parse cache entries and hit/miss statistics stored for the Parser Cache feature.

Disabling the Parser Cache

The Parser Cache feature is enabled by default. To disable the Parser Cache feature, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no parser cache**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**no parser cache** **Example:** `Device(config)# no parser cache`	Disables the Parser Cache feature: • When the parser cache is disabled, the **noparsercache** command line is written to the running configuration file. • If you wish to disable the parser cache to free system resources, you should clear the parser cache before issuing the **noparsercache** command. You will not be able to clear the parser cache after disabling it.

Reenabling the Parser Cache

To reenable the Parser Cache feature after disabling it, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parser cache**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parser cache** **Example:** `Device(config)# parser cache`	Enables the Parser Cache feature.

What to Do Next

The**showparserstatistics** command displays two sets of data, as follows:

- The number of commands in the configuration file that was last copied into the running configuration, and the time it took for the system to parse them (a configuration file can be loaded into the running

configuration at system startup, or by issuing commands such as the **copy***source***running-config** EXEC command).

- The status of the parser cache (enabled or disabled) and the number of command matches (hits or misses) since the system was started or since the parser cache was cleared.

Copying Configuration Files from Flash Memory to the Startup or Running Configuration

To copy a configuration file from Flash memory directly to your startup configuration in NVRAM or your running configuration, enter one of the commands in Step 2:

SUMMARY STEPS

1. **enable**
2. Do one of the following:
   - **copy** *filesystem* **:** [*partition-number***:**][*filename*] **nvram:startup-config**
   - **copy** *filesystem* **:** [*partition-number***:**][*filename*] **system:running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Do one of the following: • **copy** *filesystem* **:** [*partition-number***:**][*filename*] **nvram:startup-config** • **copy** *filesystem* **:** [*partition-number***:**][*filename*] **system:running-config** **Example:** `Device# copy slot0:4:ios-upgrade-1 nvram:startup-config`	Loads a configuration file directly into NVRAM. or Copies a configuration file to your running configuration.

Examples

The following example copies the file named ios-upgrade-1 from partition 4 of the Flash memory PC Card in slot 0 to the router startup configurations:

```
Device# copy slot0:4:ios-upgrade-1 nvram:startup-config

Copy '
ios-upgrade-1
' from flash device
```

```
  as 'startup-config' ? [yes/no] yes

[OK]
```

Copying a Configuration File from an FTP Server to Flash Memory Devices

To copy a configuration file from an FTP server to a Flash memory device, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*
4. **ip ftp password** *password*
5. **end**
6. **copy ftp:** [[[//[*username***:***password***@**]*location*]/*directory*]/*filename*] *flash-filesystem***:**[*partition-number***:**][*filename*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username user1`	(Optional) Specifies the remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password guessme`	(Optional) Specifies the remote password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode. This step is required only if you override the default remote username (see Steps 3 and 4).

	Command or Action	Purpose
Step 6	**copy ftp:** [[[//[*username*:*password*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*] **Example:** `Device> copy ftp:router-config slot0:new-config`	Copies the configuration file from a network server to the Flash memory device using FTP.

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Copying a Configuration File from an rcp Server to Flash Memory Devices

To copy a configuration file from an rcp server to a Flash memory device, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-username** *username*
4. **end**
5. **copy rcp:** [[[//[*username*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode.This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username user1`	(Optional) Specifies the remote username.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode.This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 5	**copy rcp:** [[[//[*username*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*] **Example:** `Device# copy rcp:router-config slot0:new-config`	Copies the configuration file from a network server to the Flash memory device using rcp. Reply to any router prompts for additional information or confirmation. The prompting will depending on how much information you provide in the **copy** command and the current setting of the **fileprompt** command.

Copying a Configuration File from a TFTP Server to Flash Memory Devices

To copy a configuration file from a TFTP server to a Flash memory device, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **copy tftp:** [[[//*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy tftp:** [[[//*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*] **Example:** `Device# copy tftp:router-config slot0:new-config`	Copies the file from a TFTP server to the Flash memory device. Reply to any Device prompts for additional information or confirmation. The prompting will depending on how much information you provide in the **copy** command and the current setting of the **fileprompt** command.

Examples

The following example shows the copying of the configuration file named Device-config from a TFTP server to the Flash memory card inserted in slot 0 of the Network Processing Engine (NPE) or Route Switch Processor (RSP) card of a Cisco 7500 series Device. The copied file is renamed new-config.

```
Device# copy tftp:router-config slot0:new-config
```

Reexecuting the Configuration Commands in the Startup Configuration File

To reexecute the commands located in the startup configuration file, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure memory**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure memory** **Example:** `Device# configure memory`	Reexecutes the configuration commands located in the startup configuration file .

Clearing the Startup Configuration

You can clear the configuration information from the startup configuration. If you reboot the router with no startup configuration, the router will enter the Setup command facility so that you can configure the router from scratch.To clear the contents of your startup configuration, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **erase nvram**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**erase nvram** **Example:**	Clears the contents of your startup configuration.

	Command or Action	Purpose
	`Device# erase nvram`	**Note** For all platforms except the Class A Flash file system platforms, this command erases NVRAM. The startup configuration file cannot be restored once it has been deleted. On Class A Flash file system platforms, when you use the **erasestartup-config**EXECcommand, the Device erases or deletes the configuration pointed to by CONFIG_FILE environment variable. If this variable points to NVRAM, the Device erases NVRAM. If the CONFIG_FILE environment variable specifies a Flash memory device and configuration filename, the Device deletes the configuration file. That is, the Device marks the file as "deleted," rather than erasing it. This feature allows you to recover a deleted file.

Deleting a Specified Configuration File

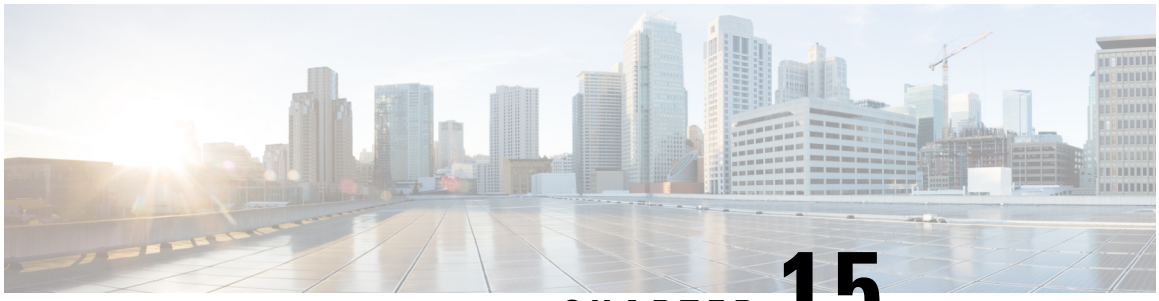
To delete a specifi ed configuration on a specific Flash device, complete the task in this section:

SUMMARY STEPS

1. **enable**
2. **delete** *flash-filesystem* **:** *filename*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**delete** *flash-filesystem* **:** *filename* **Example:** `Device# delete slot0:myconfig`	Deletes a specified configuration file on a specified Flash device. **Note** On Class A and B Flash file systems, when you delete a specific file in Flash memory, the system marks the file as deleted, allowing you to later recover a deleted file using the **undelete** EXEC command. Erased files cannot be recovered. To permanently erase the configuration file, use the **squeeze** EXEC command. On Class C Flash file systems, you cannot recover a file that has been deleted. If you attempt to erase or delete the configuration file specified by the CONFIG_FILE environment variable, the system prompts you to confirm the deletion.



CHAPTER 15

Configuration Generation Performance Enhancement

The Configuration Generation Performance Enhancement feature assists configuration management by enabling faster collection of running configuration file information. This feature is especially useful in managing large networks with numerous interfaces configured.

Restrictions for Configuration Generation Performance Enhancement

The device on which the Configuration Generation Performance Enhancement feature is used must have enough memory available to store (cache) a large interface configuration file. For example, if the interface configurations take up 15 KB of memory, using this feature would require having an additional 15 KB of memory space available.

Information About Configuration Generation Performance Enhancement

Cisco IOS XE Software Configuration Storage

In the Cisco IOS XE software configuration model, the configuration state is maintained in a distributed manner, with each component storing its own configuration state. To retrieve configuration information, the software must poll every component to collect the distributed information. This configuration state retrieval operation is performed by a process known as nonvolatile generation (NVGEN), and it is used by command-line interface (CLI) commands such as **show running-configuration**, **write memory**, and **copy**

system:running-configuration to display or copy the running system configuration. When invoked, NVGEN queries each system component and each instance of interface or other configuration objects. A running configuration file is constructed as NVGEN traverses the system performing these queries.

Benefits of the Configuration Generation Performance Enhancement

Before the Configuration Generation Performance Enhancement feature was introduced, NVGEN always had to query the entire system and could generate only a total configuration. The time required to process the running configuration creates performance problems for configuration management, because completion of the NVGEN operation can take many minutes.

The Configuration Generation Performance Enhancement feature reduces the execution time for NVGEN processes and is especially useful for managing large configuration files that contain numerous interface configurations. This feature provides faster execution of commands that process the running system configuration by caching interface configuration information in system memory, and by retrieving only configuration information that has changed.

How to Configure the Configuration Generation Performance Enhancement

Configuring the Configuration Generation Performance Enhancement

Perform this task to enable the Configuration Generation Performance Enhancement.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parser config cache interface**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**parser config cache interface** **Example:** `Device(config)# parser config cache interface`	Reduces the time required for the CLI to execute commands that manage the running system configuration, especially for large configuration files.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for the Configuration Generation Performance Enhancement

Configuring the Configuration Generation Performance Enhancement Example

The following example shows how to enable the Configuration Generation Performance Enhancement feature:

```
Device(config)# parser config cache interface
```

Verifying the Configuration Generation Performance Enhancement Example

You can verify that the **parserconfigcacheinterface** command has been enabled by checking for the command in the system configuration file displayed when you enter the **showrunning-configuration** EXEC command.

Note The first time you display the configuration file, you will not see much evidence of improvement in performance because the interface cache will be filled up. However, you will notice performance improvements when you enter subsequent NVGEN-type commands such as the **showrunning-config** EXEC command. Each time the interface configuration changes, the cache of the specified interface is flushed. The other interface data remains cached as before. Entering an NVGEN-type command after modifying the interface configuration will once again not show much evidence of improvement until the next NVGEN-type command is entered.

```
Device# show running-config
!
!
parser config cache interface
!
!
```

Additional References

The following sections provide references related to the Configuration Partitioning feature.

Related Documents

Related Topic	Document Title
Running configuration performance enhancement--**parserconfigcache** for interfaces.	Configuration Generation Performance Enhancement
Provisioning of customer services, Config Rollback, Config Locking, and configuration access control	Contextual Configuration Diff Utility
Configuration management--Config change logging.	Configuration Change Notification and Logging
Configuration management --Quick-save for config change logging[1].	Configuration Logger Persistency
Cisco IOS software configuration access control and config session locking ("Config Lock").	Exclusive Configuration Change Access and Access Session Locking

[1] The "Configuration Logger Persistency" feature allows saving just the commands entered since the last startup-config file was generated, rather than saving the entire startup configuration.

Standards

Standard	Title
No standards are associated with this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	--

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password..	http://www.cisco.com/techsupport

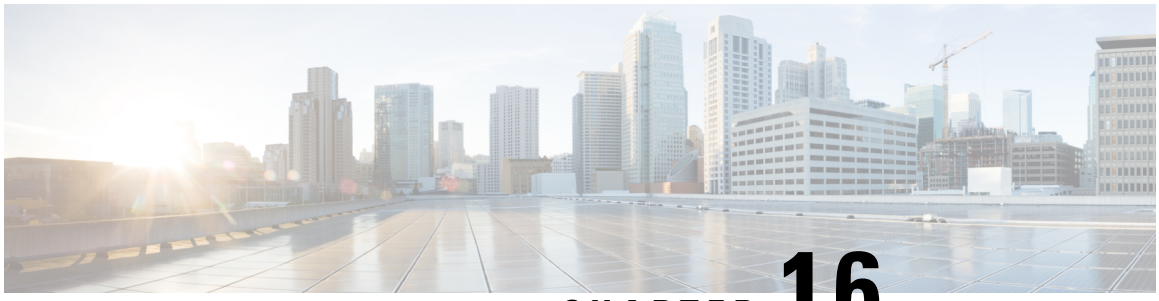
Feature Information for Configuration Generation Performance Enhancement

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 20: Feature Information for the Configuration Generation Performance Enhancement Feature

Feature Name	Releases	Feature Information
Configuration Generation Performance Enhancement		The Configuration Generation Performance Enhancement feature assists configuration management by enabling faster collection of running configuration file information. This feature is especially useful in managing large networks with numerous interfaces configured. Commands associated with this feature: • **parser config cache interface** • **parser config partition** • **parser cache**



CHAPTER 16

Exclusive Configuration Change Access and Access Session Locking

Exclusive Configuration Change Access (also called the “Configuration Lock” feature) allows you to have exclusive change access to the Cisco IOS XE running configuration, preventing multiple users from making concurrent configuration changes.

The Access Session Locking addition to this feature extends the Exclusive Configuration Change Access feature such that **show** and **debug** commands entered by the user holding the configuration lock always have execution priority; **show** and **debug** commands entered by other users are only allowed to run after the processes initiated by the configuration lock owner have finished.

The Exclusive Configuration Change Access feature (“exposed lock”) is complementary with the locking mechanism in the Configuration Replace and Configuration Rollback feature (“rollback lock”).

Information About Locking the Configuration

Exclusive Configuration Change Access and Access Session Locking

Devices running Cisco IOS software maintain a running configuration that determines the configuration state of the device. Changes to the running configuration alter the behavior of the device. Because Cisco IOS software allows multiple users to change the running configuration via the device CLI (including the device console and telnet Secure Shell (SSH)), in some operating environments it would be beneficial to prevent multiple users from making concurrent changes to the Cisco IOS running configuration. Temporarily limiting access to the Cisco IOS running configuration prevents inadvertent conflicts or cases where two users attempt to configure the same portion of the running configuration.

The Exclusive Configuration Change Access feature (also called the "Configuration Lock" feature) allows you to have exclusive change access to the Cisco IOS running configuration, preventing multiple users from making concurrent configuration changes.

This feature provides exclusive change access to the Cisco IOS running configuration from the time you enter global configuration mode by using the **configure terminal** command. This gives the effect of a "configuration lock," preventing other users from changing the Cisco IOS running configuration. The configuration lock is automatically released when the user exits Cisco IOS configuration mode.

The Exclusive Configuration Change Access feature is enabled using the **configuration mode exclusive** command in global configuration mode. Exclusive configuration change access can be set to **auto**, so that the Cisco IOS configuration mode is locked whenever anyone uses the **configure terminal** command, or it can be set to **manual**, so that the Cisco IOS configuration mode is locked only when the **configure terminal lock** command is issued.

The Exclusive Configuration Change Access feature is complementary with the locking mechanism for the Configuration Replace and Configuration Rollback feature introduced in Cisco IOS Release 12.2(25)S and 12.3(7)T.

Access Session Locking

The Access Session Locking feature extends the Exclusive Configuration Change Access feature such that **show** and **debug** commands entered by the user holding the configuration lock always have execution priority. This feature prevents concurrent configuration access and also provides an option to prevent simultaneous processes, such as a **show** command entered by another user, from executing while other configuration commands are being executed. When this feature is enabled, the commands entered by the user with the configuration lock (such as configuration commands) always have priority over commands entered by other users.

How to Configure Configuration Exclusive Confguration Change-Access and Access Session Locking

Enabling Exclusive Configuration Change Access and Access Session Locking

Note Effective with Cisco IOS Release 12.2(33)SRE, the Exclusive Configuration Change Access and Access Session Locking feature is not available in Cisco IOS software. Use the Parser Concurrency and Locking Improvements feature instead of this feature. See the "Enabling Parser Concurrency and Locking Improvements" section for more information.

Perform this task to enable the Exclusive Configuration Change Access and Access Session Locking feature.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **configuration mode exclusive**

4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**configuration mode exclusive** **Example:** `Router(config)# configuration mode exclusive`	Enables exclusive configuration change access (configuration lock feature). • When the command is enabled, configuration sessions are performed in single-user (exclusive) mode.
Step 4	**end** **Example:** `Router(config)# end`	Ends your configuration session and returns the CLI to privileged EXEC mode.

Obtaining Exclusive Configuration Change Access

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **configure terminal lock**
4. Configure the system by entering your changes to the running configuration.
5. Do one of the following:
   - **end**
   - or
   - **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**configure terminal lock** **Example:** `Router(config)# configure terminal lock`	(Optional) Locks the Cisco IOS software in exclusive (single-user) mode. • This command can be used only if you have previously enabled configuration locking by using the **configuration mode exclusive** command. • This command is available in Cisco IOS Release 12.3(14)T or later releases.
Step 4	Configure the system by entering your changes to the running configuration.	--
Step 5	Do one of the following: • **end** • or • **exit** **Example:** `Router(config)# end` **Example:** **Example:** `Router(config)# exit`	Ends your configuration session, automatically releases the session lock obtained in Step 1, and exits to privileged EXEC mode. **Note** Either the **end** command, the **exit** command, or the Ctrl-Z key combination releases the configuration lock. Use of the **end** command is recommended.

Monitoring and Troubleshooting Configuration Locking

Perform either or both steps in this task to monitor or troubleshoot the Exclusive Configuration Change Access and Access Session Locking feature.

SUMMARY STEPS

1. **show configuration lock**
2. **debug configuration lock**

DETAILED STEPS

Step 1 **show configuration lock**

Use this command to display the status and details of any current configuration locks, including the owner, user, terminal, lock state, and lock class.

If you cannot enter global configuration mode, you can use this command to determine if the configuration session is locked by another user, and who that user is.

Example:

Step 2 **debug configuration lock**

Use this command to enable debugging of Cisco IOS configuration locks (exposed class locks or rollback class locks):

Example:

```
Router# debug configuration lock

Session1 from console
=========================
Router# configure terminal lock
Configuration mode locked exclusively. The lock will be cleared once you exit out of configuration
mode using end/exit
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Parser : LOCK REQUEST in EXCLUSIVE mode
Parser: <configure terminal lock> - Config. Lock requested by process <3> client <PARSER Client>
Parser: <configure terminal lock> - Config. Lock acquired successfully !
Router(config)#
```

Configuration Examples for Locking the Configuration

Configuring an Exclusive Lock in Auto Mode Example

The following example shows how to enable the exclusive lock in auto mode for single-user auto configuration mode using the **configurationmodeexclusive** command. Once the Cisco IOS configuration file is locked exclusively, you can verify this configuration by using the **showconfigurationlock**command.

```
Router# configure terminal
Router(config)#
Router(config)# exit
Router# configure terminal
! Locks configuration mode exclusively.
Router# show configuration lock
Parser Configure Lock
Owner PID        :  10
User             :  User1
TTY              :  3
Type             :  EXCLUSIVE
State            :  LOCKED
Class            :  Exposed
Count            :  0
Pending Requests :  0
User debug info  :  0
```

Configuring an Exclusive Lock in Manual Mode Example

Additional References

The following sections provide references related to locking the configuration.

Related Documents

Related Topic	Document Title
Commands for managing configuration files	*Cisco IOS Configuration Management Command Reference*
Information about managing configuration files	*Managing Configuration Files*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

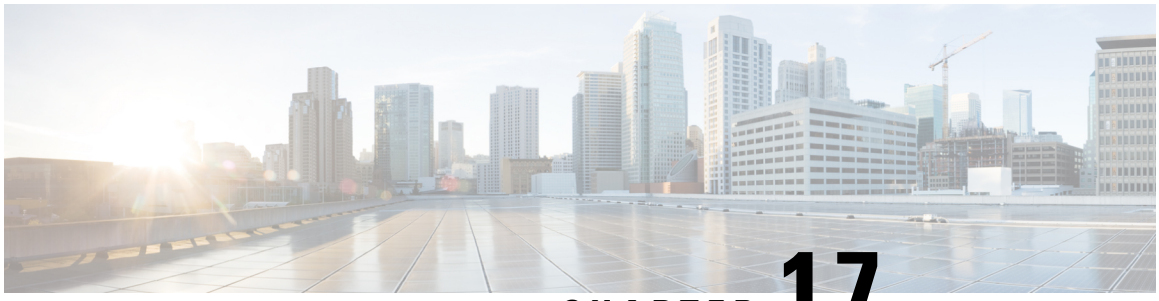
Feature Information for Exclusive Configuration Change Access and Access Session Locking

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 21: Feature Information for Exclusive Configuration Change Access and Access Session Locking

Feature Name	Releases	Feature Information
Exclusive Configuration Change Access and Access Session Locking	12.3(14)T 12.0(31)S 12.2(33)SRA 12.4(11)T 12.2(33)SXH 12.2(33)SB	The Exclusive Configuration Change Access feature (also called the "Configuration Lock" feature) allows you to have exclusive change access to the Cisco IOS running configuration, preventing multiple users from making concurrent configuration changes. The Access Session Locking addition to this feature extends the Exclusive Configuration Change Access feature such that **show** and **debug** commands entered by the user holding the configuration lock always have execution priority;**show** and **debug** commands entered by other users are allowed to run only after the processes initiated by the configuration lock owner have finished. The Exclusive Configuration Change Access feature is complementary with the locking mechanism for the Configuration Replace and Configuration Rollback feature ("rollback lock"). The Configuration Lock feature feature was integrated into Release 12.0S, and the Access Session Locking feature extension was implemented. The **configuration mode exclusive**command was extended to include the following keyword options: **config_wait**, **expire**, **interleave**, **lock-show**, **retry_wait**, and **terminate**. The output of the **show configuration lock**command was improved. The extended feature was integrated into Releases 12.2(33)SRA, 12.4(11)T, 12.2(33)SXH, and 12.2(33)SB. The following sections provide information about this feature: • Information About Locking the Configuration • How to Configure Configuration Lock The following commands were introduced or modified: **clear configuration lock**, **configuration mode exclusive**, and **configure terminal lock**.
Parser Concurrency and Locking Improvements	12.2(33)SRE 15.1(1)T	The Parser Concurrency and Locking Improvements feature provides a common interface that ensures that exclusive access is granted to the requested process and prevents others from concurrently accessing the Cisco IOS configuration. It allows access only to the user holding the lock and prevents other clients from accessing the configuration. The following sections provide information about this feature: • Parser Concurrency and Locking Improvements • Enabling Parser Concurrency and Locking Improvements The following commands were introduced or modified: **parser command serializer** and **test parser session-lock**.



CHAPTER 17

Configuration Replace and Configuration Rollback

The Configuration Replace and Configuration Rollback feature provides the capability to replace the current running configuration with any saved Cisco IOS configuration file. This functionality can be used to revert to a previous configuration state, effectively rolling back any configuration changes that were made since that configuration file was saved.

Prerequisites for Configuration Replace and Configuration Rollback

The format of the configuration files used as input by the Configuration Replace and Configuration Rollback feature must comply with standard Cisco software configuration file indentation rules as follows:

- Start all commands on a new line with no indentation, unless the command is within a configuration submode.
- Indent commands within a first-level configuration submode one space.
- Indent commands within a second-level configuration submode two spaces.
- Indent commands within subsequent submodes accordingly.

These indentation rules describe how the software creates configuration files for such commands as **show running-config** or **copy running-config** *destination-url*. Any configuration file generated on a Cisco device complies with these rules.

Free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration) is required.

Restrictions for Configuration Replace and Configuration Rollback

If the device does not have free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration), the configuration replace operation is not performed.

Certain Cisco configuration commands such as those pertaining to physical components of a networking device (for example, physical interfaces) cannot be added or removed from the running configuration. For example, a configuration replace operation cannot remove the **interface ethernet 0** command line from the current running configuration if that interface is physically present on the device. Similarly, the **interface ethernet 1** command line cannot be added to the running configuration if no such interface is physically present on the device. A configuration replace operation that attempts to perform these types of changes results in error messages indicating that these specific command lines failed.

In very rare cases, certain Cisco configuration commands cannot be removed from the running configuration without reloading the device. A configuration replace operation that attempts to remove this type of command results in error messages indicating that these specific command lines failed.

When using the Public Key Infrastructure (PKI) for signature certificate validation, the **copy startup-config running-config** and the **configure replace** commands, are not supported. A device reload is needed when you replace or load configuration instructions from a different file.

Follow these steps to accomplish this task:

- **Step 1:** Create a backup file of your running configuration file. Copy your running-configuration file to the startup-configuration file.

```
Router#copy startup-config running-config
```

- **Step 2:** Restore your configuration from the backup file. Copy your startup-configuration file to the running-configuration file.

```
Router#copy running-config startup-config
```

- **Step 3:** Remove the PKI certificate.

```
Router#no crypto pki trustpoint trustpoint-name

% Removing an enrolled trustpoint will destroy all certificates
 received from the related Certificate Authority.

Are you sure you want to do this? [yes/no]: yes
% Be sure to ask the CA administrator to revoke your certificates.
```

- **Step 4:** Import the certificate again.

Note If you issue the **configure replace** command to replace the current running configuration with any saved Cisco IOS configuration file, the CLI will prompt you to reload the device after you issue the command for your configuration to persist.

If you copy the running configuration to the startup configuration using the **copy startup-config running-config** command, the CLI will prompt you to reload the device for your configuration changes to take effect.

Information About Configuration Replace and Configuration Rollback

Configuration Archive

The Cisco IOS configuration archive is intended to provide a mechanism to store, organize, and manage an archive of Cisco IOS configuration files to enhance the configuration rollback capability provided by the **configurereplace** command. Before this feature was introduced, you could save copies of the running configuration using the **copyrunning-config***destination-url* command, storing the replacement file either locally or remotely. However, this method lacked any automated file management. On the other hand, the Configuration Replace and Configuration Rollback feature provides the capability to automatically save copies of the running configuration to the Cisco IOS configuration archive. These archived files serve as checkpoint configuration references and can be used by the **configurereplace** command to revert to previous configuration states.

The **archiveconfig** command allows you to save Cisco IOS configurations in the configuration archive using a standard location and filename prefix that is automatically appended with an incremental version number (and optional timestamp) as each consecutive file is saved. This functionality provides a means for consistent identification of saved Cisco IOS configuration files. You can specify how many versions of the running configuration are kept in the archive. After the maximum number of files are saved in the archive, the oldest file is automatically deleted when the next, most recent file is saved. The **showarchive** command displays information for all configuration files saved in the Cisco IOS configuration archive.

The Cisco IOS configuration archive, in which the configuration files are stored and available for use with the **configurereplace** command, can be located on the following file systems:

- If your platform has disk0--disk0:, disk1:, ftp:, pram:, rcp:, slavedisk0:, slavedisk1:, or tftp:
- If your platform does not have disk0--ftp:, http:, pram:, rcp:, or tftp:

Configuration Replace

The **configurereplace** command provides the capability to replace the current running configuration with any saved Cisco IOS configuration file. This functionality can be used to revert to a previous configuration state, effectively rolling back any configuration changes that were made since the previous configuration state was saved.

When using the **configurereplace** command, you must specify a saved Cisco IOS configuration as the replacement configuration file for the current running configuration. The replacement file must be a complete

configuration generated by a Cisco IOS device (for example, a configuration generated by the **copyrunning-config***destination-url* command), or, if generated externally, the replacement file must comply with the format of files generated by Cisco IOS devices. When the **configurereplace** command is entered, the current running configuration is compared with the specified replacement configuration and a set of diffs is generated. The algorithm used to compare the two files is the same as that employed by the **showarchiveconfigdifferences** command. The resulting diffs are then applied by the Cisco IOS parser to achieve the replacement configuration state. Only the diffs are applied, avoiding potential service disruption from reapplying configuration commands that already exist in the current running configuration. This algorithm effectively handles configuration changes to order-dependent commands (such as access lists) through a multiple pass process. Under normal circumstances, no more than three passes are needed to complete a configuration replace operation, and a limit of five passes is performed to preclude any looping behavior.

The Cisco IOS **copy***source-url***running-config** command is often used to copy a stored Cisco IOS configuration file to the running configuration. When using the **copy***source-url***running-config**command as an alternative to the **configurereplace***target-url*command, the following major differences should be noted:

- The **copy***source-url***running-config** command is a merge operation and preserves all the commands from both the source file and the current running configuration. This command does not remove commands from the current running configuration that are not present in the source file. In contrast, the **configurereplace***target-url* command removes commands from the current running configuration that are not present in the replacement file and adds commands to the current running configuration that need to be added.
- The **copy***source-url***running-config** command applies every command in the source file, whether or not the command is already present in the current running configuration. This algorithm is inefficient and, in some cases, can result in service outages. In contrast, the **configurereplace***target-url* command only applies the commands that need to be applied--no existing commands in the current running configuration are reapplied.
- A partial configuration file may be used as the source file for the **copy***source-url***running-config** command, whereas a complete Cisco IOS configuration file must be used as the replacement file for the **configurereplace***target-url* command.

In Cisco IOS Release 12.2(25)S and 12.3(14)T, a locking feature for the configuration replace operation was introduced. When the **configurereplace** command is used, the running configuration file is locked by default for the duration of the configuration replace operation. This locking mechanism prevents other users from changing the running configuration while the replacement operation is taking place, which might otherwise cause the replacement operation to terminate unsuccessfully. You can disable the locking of the running configuration by using the **nolock** keyword when issuing the **configurereplace**command.

The running configuration lock is automatically cleared at the end of the configuration replace operation. You can display any locks that may be currently applied to the running configuration using the **showconfigurationlock** command.

Note In a scenario when you are performing a configuration replace using a configuration that is not sourced from IOS(such as a custom written configuration) if the login banner has a delimiter that is not the EXT character (ASCII code 003), the banner configuration is rejected and not included in the replaced configuration. Non-working delimiters include ^C, %, #, CC etc.

Configuration Rollback

The concept of rollback comes from the transactional processing model common to database operations. In a database transaction, you might make a set of changes to a given database table. You then must choose whether to commit the changes (apply the changes permanently) or to roll back the changes (discard the changes and revert to the previous state of the table). In this context, rollback means that a journal file containing a log of the changes is discarded, and no changes are applied. The result of the rollback operation is to revert to the previous state, before any changes were applied.

The **configurereplace** command allows you to revert to a previous configuration state, effectively rolling back changes that were made since the previous configuration state was saved. Instead of basing the rollback operation on a specific set of changes that were applied, the Cisco IOS configuration rollback capability uses the concept of reverting to a specific configuration state based on a saved Cisco IOS configuration file. This concept is similar to the database idea of saving a checkpoint (a saved version of the database) to preserve a specific state.

If the configuration rollback capability is desired, you must save the Cisco IOS running configuration before making any configuration changes. Then, after entering configuration changes, you can use that saved configuration file to roll back the changes (using the **configurereplace** *target-url* command). Furthermore, since you can specify any saved Cisco IOS configuration file as the replacement configuration, you are not limited to a fixed number of rollbacks, as is the case in some rollback models based on a journal file.

Configuration Rollback Confirmed Change Operation

The Configuration Rollback Confirmed Change feature enables an added criterion of a confirmation to configuration changes. This functionality enables a rollback to occur if a confirmation of the requested changes is not received in a configured time frame. Command failures can also be configured to trigger a configuration rollback.

The following steps outline how this process is achieved:

1. A new option allows you to request confirmation (a confirmation time limit must be supplied) of the configuration changes.
2. You must enter the confirmation command. If no confirmation is entered within the requested time limit, the configuration reverts to its previous state.

Benefits of Configuration Replace and Configuration Rollback

- Allows you to revert to a previous configuration state, effectively rolling back configuration changes.
- Allows you to replace the current running configuration file with the startup configuration file. After you replace the file, you must reload the device for your configuration changes to take effect.
- Allows you to revert to any saved Cisco IOS configuration state.
- Simplifies configuration changes by allowing you to apply a complete configuration file to the router, where only the commands that need to be added or removed are affected.
- When using the **configure replace** command as an alternative to the **copy** *source-url* **running-config** command, increases efficiency and prevents risk of service outages by not reapplying existing commands in the current running configuration. After you replace the file, you must reload the device for your configuration changes to take effect.

How to Use Configuration Replace and Configuration Rollback

Creating a Configuration Archive

No prerequisite configuration is needed to use the **configurereplace** command. Using the **configurereplace** command in conjunction with the Cisco IOS configuration archive and the **archiveconfig** command is optional but offers significant benefit for configuration rollback scenarios. Before using the **archiveconfig** command, the configuration archive must be configured. Perform this task to configure the characteristics of the configuration archive.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **path** *url*
5. **maximum** *number*
6. **time-period** *minutes*
7. **end**
8. **archive config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**path** *url* **Example:** `Device(config-archive)# path flash:myconfig`	Specifies the location and filename prefix for the files in the Cisco IOS configuration archive. **Note** If a directory is specified in the path instead of file, the directory name must be followed by a forward slash as follows: path flash:/directory/. The forward slash is not necessary after a filename; it is only necessary when specifying a directory.

	Command or Action	Purpose
Step 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	(Optional) Sets the maximum number of archive files of the running configuration to be saved in the Cisco IOS configuration archive. • The *number* argument is the maximum number of archive files of the running configuration to be saved in the Cisco IOS configuration archive. Valid values are from 1 to 14. The default is 10. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 10`	(Optional) Sets the time increment for automatically saving an archive file of the current running configuration in the Cisco IOS configuration archive. • The minutes argument specifies how often, in minutes, to automatically save an archive file of the current running configuration in the Cisco IOS configuration archive. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 7	**end** **Example:** `Device(config-archive)# end`	Exits to privileged EXEC mode.
Step 8	**archive config** **Example:** `Device# archive config`	Saves the current running configuration file to the configuration archive. **Note** The **path** command must be configured before using this command.

Performing a Configuration Replace or Configuration Rollback Operation

Perform this task to replace the current running configuration file with a saved Cisco IOS configuration file.

Note You must create a configuration archive before performing this procedure. See Creating a Configuration Archive for detailed steps. The following procedure details how to return to that archived configuration in the event of a problem with the current running configuration.

SUMMARY STEPS

1. **enable**
2. **configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**reverttrigger[error]***[***timer***minutes***]*|**time***minutes*]
3. **configure revert {now** |**timer{***minutes*|**idle***minutes***}}**
4. **configure confirm**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**reverttrigger[error]***[***timer***minutes***]*|**time***minutes*] **Example:** `Device# configure replace flash:myconfig-1 list time 30`	Replaces the current running configuration file with a saved Cisco IOS configuration file. After you replace the file, you must reload the device for your configuration changes to take effect. • The *target -url* argument is a URL (accessible by the Cisco IOS file system) of the saved Cisco IOS configuration file that is to replace the current running configuration, such as the configuration file created using the **archiveconfig** command. • The **list** keyword displays a list of the command lines applied by the Cisco IOS software parser during each pass of the configuration replace operation. The total number of passes performed is also displayed. • The **force**keyword replaces the current running configuration file with the specified saved Cisco IOS configuration file without prompting you for confirmation. • The **time***minutes*keyword and argument specify the time (in minutes) within which you must enter the **configureconfirm** command to confirm replacement of the current running configuration file. If the **configureconfirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configurereplace** command). • The **nolock** keyword disables the locking of the running configuration file that prevents other users

	Command or Action	Purpose
		from changing the running configuration during a configuration replace operation. • The **reverttrigger** keywords set the following triggers for reverting to the original configuration: • **error**--Reverts to the original configuration upon error. • **timer***minutes*--Reverts to the original configuration if specified time elapses. • The **ignorecase** keyword allows the configuration to ignore the case of the confirmation command.
Step 3	**configure revert {now |timer{***minutes***|idle***minutes***}}** **Example:** `Device# configure revert now` **Example:**	(Optional) To cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback, use the **configurerevert**command in privileged EXEC mode. • **now** --Triggers the rollback immediately. • **timer** --Resets the configuration revert timer. • Use the *minutes* argument with the **timer** keyword to specify a new revert time in minutes. • Use the **idle** keyword along with a time in minutes to set the maximum allowable time period of no activity before reverting to the saved configuration.
Step 4	**configure confirm** **Example:** `Device# configure confirm`	(Optional) Confirms replacement of the current running configuration file with a saved Cisco IOS configuration file. After you replace the file, you must reload the device for your configuration changes to take effect. **Note** Use this command only if the **time***seconds* keyword and argument of the **configurereplace** command are specified.
Step 5	**exit** **Example:** `Device# exit`	Exits to user EXEC mode.

Monitoring and Troubleshooting the Feature

Perform this task to monitor and troubleshoot the Configuration Replace and Configuration Rollback feature.

SUMMARY STEPS

1. **enable**
2. **show archive**
3. **debug archive versioning**
4. **debug archive config timestamp**
5. **exit**

DETAILED STEPS

Step 1 enable

Use this command to enable privileged EXEC mode. Enter your password if prompted. For example:

Example:

```
Device> enable
Device#
```

Step 2 show archive

Use this command to display information about the files saved in the Cisco IOS configuration archive. For example:

Example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfig-2
 Archive #  Name
   0
   1       flash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

The following is sample output from the **showarchive** command after several archive files of the running configuration have been saved. In this example, the maximum number of archive files to be saved is set to three.

Example:

```
Device# show archive
There are currently 3 archive configurations saved.
The next archive file will be named flash:myconfig-8
 Archive #  Name
   0
   1       :Deleted
   2       :Deleted
   3       :Deleted
   4       :Deleted
```

```
5       flash:myconfig-5
6       flash:myconfig-6
7       flash:myconfig-7 <- Most Recent
8
9
10
11
12
13
14
```

Step 3 debug archive versioning

Use this command to enable debugging of the Cisco IOS configuration archive activities to help monitor and troubleshoot configuration replace and rollback. For example:

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file flash:myconfig-7
Jan  9 06:46:29.547: backup worked
```

Step 4 debug archive config timestamp

Use this command to enable debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled. For example:

Example:

```
Device# debug archive config timestamp
Device# configure replace flash:myconfig force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file slot0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

Step 5 exit

Use this command to exit to user EXEC mode. For example:

Example:

```
Device# exit
Device>
```

Configuration Examples for Configuration Replace and Configuration Rollback

Creating a Configuration Archive Example

The following example shows how to perform the initial configuration of the Cisco IOS configuration archive. In this example, flash:myconfig is specified as the location and filename prefix for the files in the configuration archive and a value of 10 is set as the maximum number of archive files to be saved.

```
configure terminal
!
archive
 path flash:myconfig
 maximum 10
 end
```

Replacing the Current Running Configuration with a Saved Cisco IOS Configuration File Example

The following example shows how to replace the current running configuration with a saved Cisco IOS configuration file named flash:myconfig. The **configure replace** command interactively prompts you to confirm the operation.

```
Device# configure replace flash:myconfig
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
```

In the following example, the **list** keyword is specified in order to display the command lines that were applied during the configuration replace operation:

```
Device# configure replace flash:myconfig list
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
!Pass 1
!List of Commands:
no snmp-server community public ro
snmp-server community mystring ro
```

```
end
Total number of passes: 1
Rollback Done
```

Reverting to the Startup Configuration File Example

The following example shows how to revert to the Cisco IOS startup configuration file using the **configurereplace** command. This example also shows the use of the optional **force** keyword to override the interactive user prompt.

```
Device# configure replace nvram:startup-config force
Total number of passes: 1
Rollback Done
```

Example: Performing a Configuration Replace Operation with the configure confirm Command

The following example shows the use of the **configure replace** command with the **time** *minutes* keyword and argument. You must enter the **configure confirm** command within the specified time limit to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command).

```
Device# configure replace nvram:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
Device# configure confirm
```

The following example shows the use of the **configure revert** command with the **timer** keyword. You must enter the **configure revert** command to cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback.

```
Device# configure revert timer 100
```

Performing a Configuration Rollback Operation Example

The following example shows how to make changes to the current running configuration and then roll back the changes. As part of the configuration rollback operation, you must save the current running configuration before making changes to the file. In this example, the **archiveconfig** command is used to save the current running configuration. The generated output of the **configurereplace** command indicates that only one pass was performed to complete the rollback operation.

Note Before using the **archiveconfig** command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.

You first save the current running configuration in the configuration archive as follows:

```
archive config
```

You then enter configuration changes as shown in the following example:

```
configure terminal
!
user netops2 password rain
user netops3 password snow
exit
```

After having made changes to the running configuration file, assume you now want to roll back these changes and revert to the configuration that existed before the changes were made. The **showarchive** command is used to verify the version of the configuration to be used as a replacement file. The **configurereplace** command is then used to revert to the replacement configuration file as shown in the following example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfig-2
 Archive #  Name
   0
   1       flash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
Device# configure replace flash:myconfig-1
Total number of passes: 1
Rollback Done
```

Additional References

The following sections provide references related to the Configuration Replace and Configuration Rollback feature.

Related Documents

Related Topic	Document Title
Configuration Locking	*Exclusive Configuration Change Access and Access Session Locking*
Commands for managing configuration files	*Cisco IOS Configuration Fundamentals Command Reference*
Information about managing configuration files	*Managing Configuration Files*
Using the Contextual Configuration Diff Utility feature	*Contextual Configuration Diff Utility*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

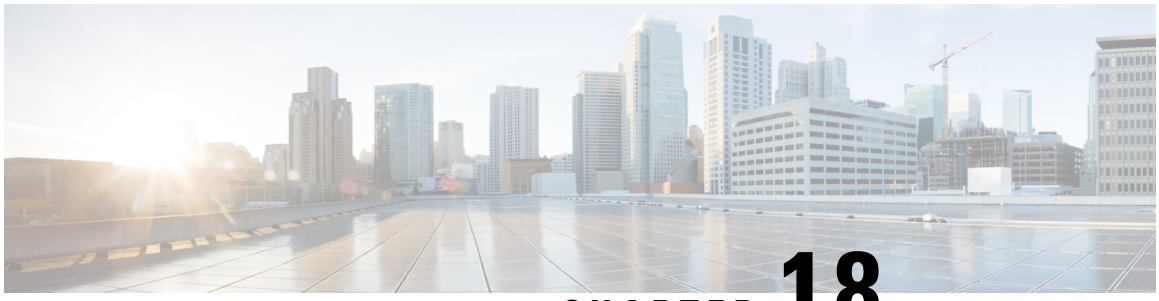
Feature Information for Configuration Replace and Configuration Rollback

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 22: Feature Information for Configuration Replace and Configuration Rollback

Feature Name	Releases	Feature Information
Configuration Replace and Configuration Rollback		The Configuration Replace and Configuration Rollback feature provides the capability to replace the current running configuration with any saved Cisco IOS configuration file. This functionality can be used to revert to a previous configuration state, rolling back any configuration changes that were made since that configuration file was saved. The following sections provide feature information: The following commands were modified by this feature: **archive config**, **configure confirm**, **configure replace**, **debug archive config timestamp**, **debug archive versioning**, **maximum**, **path (archive configuration)**, **show archive**, **show configuration lock**, **time-period**.
Configuration Versioning		The Configuration Versioning feature allows you to maintain and manage backup copies of the Cisco IOS running configuration on or off the device. The Configuration Replace feature uses the Configuration Versioning feature to provide a rollback to a saved copy of the running configuration.
Exclusive Configuration Change Access		The Exclusive Configuration Change Access feature (also called the Configuration Lock feature) allows you to have exclusive change access to the Cisco IOS running configuration, preventing multiple users from making concurrent configuration changes. The following command was modified by this feature and applies to the Configuration Replace and Configuration Rollback feature: **show configuration lock**. Refer to the separate module, Exclusive Configuration Change Access and Access Session Locking, for details
Configuration Rollback Confirmed Change		The Configuration Rollback Confirmed Change feature allows configuration changes to be performed with an optional requirement that they be confirmed. If this confirmation is not received, the configuration is returned to the state prior to the changes being applied. This mechanism provides a safeguard against inadvertent loss of connectivity between a network device and the user or management application due to configuration changes. The following sections provide information about this feature: The following commands were modified by this feature: **configure confirm**,**configure replace**,**configure revert**,**configure terminal**



CHAPTER 18

Contextual Configuration Diff Utility

The Contextual Configuration Diff Utility feature provides the ability to perform a line-by-line comparison of any two configuration files (accessible through the Cisco IOS XE Integrated File System [IFS]) and generate a list of the differences between them. The generated output includes information regarding configuration lines that have been added, modified, or deleted, and the configuration modes within which a changed configuration line exists.

Prerequisites for Contextual Configuration Diff Utility

The format of the configuration files used for the Contextual Configuration Diff Utility feature must comply with standard Cisco IOS XE configuration file indentation rules as follows:

- Start all commands on a new line with no indentation, unless the command is within a configuration submode.
- Indent commands within a first-level configuration submode one space.
- Indent commands within a second-level configuration submode two spaces.
- Indent commands within subsequent submodes accordingly.

The router must have a contiguous block of memory larger than the combined size of the two configuration files being compared.

Restrictions for Contextual Configuration Diff Utility

If the device does not have a contiguous block of memory larger than the combined size of the two configuration files being compared, the diff operation fails.

Information About Contextual Configuration Diff Utility

Benefits of the Contextual Configuration Diff Utility

The Contextual Configuration Diff Utility feature provides the ability to perform a line-by-line comparison of any two configuration files (accessible through the Cisco IOS XE File System [IFS]) and generate a list of the differences between them. The generated output includes information regarding the following items:

- Configuration lines that have been added, modified, or deleted.
- Configuration modes within which a changed configuration line exists.
- Location changes of configuration lines that are order-sensitive. For example, the **ip access-list** and **community-lists** commands are order-sensitive commands dependent on where they are listed within a configuration file in relation to other Cisco IOS XE commands of similar type.

Contextual Configuration Diff Utility Output Format

Diff Operation

The Contextual Configuration Diff Utility feature uses the filenames of two configuration files as input. A diff operation is performed on the specified files and a list of differences between the two files is generated as output by using the **show archive config differences** command. Interpreting the output is dependent on the order in which the two files are specified in the command. In this section, we assume that the filename of the file entered first is file1 and the filename of the file entered second is file2. Each entry in the generated output list is prefixed with a unique text symbol to indicate the type of difference found. The text symbols and their meanings are as follows:

- A minus symbol (–) indicates that the configuration line exists in file1 but not in file2.
- A plus symbol (+) indicates that the configuration line exists in file2 but not in file1.
- An exclamation point (!) with descriptive comments identifies order-sensitive configuration lines whose location is different in file1 than in file2.

Incremental Diff Operation

Some applications require that the generated output of a diff operation contain configuration lines that are unmodified (in other words, without the minus and plus symbols). For these applications, an incremental diff operation can be performed by using the **show archive config incremental-diffs** command, which compares a specified configuration file to the running configuration file ().

When an incremental diff operation is performed, a list of the configuration lines that do not appear in the running configuration file (in other words, configuration lines that appear only in the specified file that is being compared to the running configuration file) is generated as output. An exclamation point (!) with descriptive comments identifies order-sensitive configuration lines whose location is different in the specified configuration file than in the running configuration file.

How to Use the Contextual Configuration Diff Utility

Performing a Line-by-Line File Comparison Using the Contextual Configuration Diff Utility

SUMMARY STEPS

1. **enable**
2. Enter one of the following:
   - **show archive config differences** [*file1* [*file2*]]
   - **show archive config incremental-diffs** *file*
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Enter one of the following: • **show archive config differences** [*file1* [*file2*]] • **show archive config incremental-diffs** *file* **Example:** `Device# show archive config differences running-config startup-config` **Example:** `Device# show archive config incremental-diffs nvram:startup-config`	Performs a line-by-line comparison of any two configuration files (accessible through the Cisco IOS File System and generates a list of the differences between them. or Performs a line-by-line comparison of a specified configuration file to the running configuration file and generates a list of the configuration lines that do not appear in the running configuration file.
Step 3	**exit** **Example:** `Device# exit`	Exits to user EXEC mode.

Configuration Examples for the Contextual Configuration Diff Utility

Diff Operation Example

In this example, a diff operation is performed on the running and startup configuration files. The table below shows the configuration files used for this example.

Table 23: Configuration Files Used for the Diff Operation Example

Running Configuration File	Startup Configuration File
no ip subnet-zero	ip subnet-zero
ip cef	ip cef
interface FastEthernet1/0	ip name-server 10.4.4.4
ip address 10.7.7.7 255.0.0.0	voice dnis-map 1
no ip route-cache	dnis 111
no ip mroute-cache	interface FastEthernet1/0
duplex half	no ip address
no ip classless	no ip route-cache
snmp-server community public RO	no ip mroute-cache
	shutdown
	duplex half
	ip default-gateway 10.5.5.5
	ip classless
	access-list 110 deny ip any host 10.1.1.1
	access-list 110 deny ip any host 10.1.1.2
	access-list 110 deny ip any host 10.1.1.3
	snmp-server community private RW

The following is sample output from the **show archive config differences**command. This sample output displays the results of the diff operation performed on the configuration files in the table below.

```
Device# show archive config differences system:running-config nvram:startup-config

+ip subnet-zero

+ip name-server 10.4.4.4

+voice dnis-map 1

 +dnis 111

interface FastEthernet1/0

 +no ip address

 +shutdown

+ip default-gateway 10.5.5.5

+ip classless

+access-list 110 deny ip any host 10.1.1.1

+access-list 110 deny ip any host 10.1.1.2

+access-list 110 deny ip any host 10.1.1.3

+snmp-server community private RW

-no ip subnet-zero

interface FastEthernet1/0

 -ip address 10.7.7.7 255.0.0.0

-no ip classless

-snmp-server community public RO
```

Incremental Diff Operation Example

In this example, an incremental diff operation is performed on the startup and running configuration files. The table below shows the configuration files used for this example.

Table 24: Configuration Files Used for the Incremental Diff Operation Example

Startup Configuration File	Running Configuration File
ip subnet-zero	no ip subnet-zero
ip cef	ip cef
ip name-server 10.4.4.4	interface FastEthernet1/0
voice dnis-map 1	ip address 10.7.7.7 255.0.0.0
dnis 111	no ip route-cache
interface FastEthernet1/0	no ip mroute-cache
no ip address	duplex half
no ip route-cache	no ip classless
no ip mroute-cache	snmp-server community public RO
shutdown	
duplex half	
ip default-gateway 10.5.5.5	
ip classless	
access-list 110 deny ip any host 10.1.1.1	
access-list 110 deny ip any host 10.1.1.2	
access-list 110 deny ip any host 10.1.1.3	
snmp-server community private RW	

The following is sample output from the **show archive config incremental-diffs**command. This sample output displays the results of the incremental diff operation performed on the configuration files in the table below.

```
Device# show archive config incremental-diffs startup-config

ip subnet-zero

ip name-server 10.4.4.4

voice dnis-map 1
```

```
 dnis 111
interface FastEthernet1/0
 no ip address
 shutdown
ip default-gateway 10.5.5.5
ip classless
 access-list 110 deny   ip any host 10.1.1.1
 access-list 110 deny   ip any host 10.1.1.2
 access-list 110 deny   ip any host 10.1.1.3
snmp-server community private RW
```

Additional References

The following sections provide references related to the Configuration Partitioning feature.

Related Documents

Related Topic	Document Title
Running configuration performance enhancement--**parserconfigcache** for interfaces.	Configuration Generation Performance Enhancement
Provisioning of customer services, Config Rollback, Config Locking, and configuration access control	Contextual Configuration Diff Utility
Configuration management--Config change logging.	Configuration Change Notification and Logging
Configuration management --Quick-save for config change logging[2].	Configuration Logger Persistency
Cisco IOS software configuration access control and config session locking ("Config Lock").	Exclusive Configuration Change Access and Access Session Locking

[2] The "Configuration Logger Persistency" feature allows saving just the commands entered since the last startup-config file was generated, rather than saving the entire startup configuration.

Standards

Standard	Title
No standards are associated with this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	--

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password..	http://www.cisco.com/techsupport

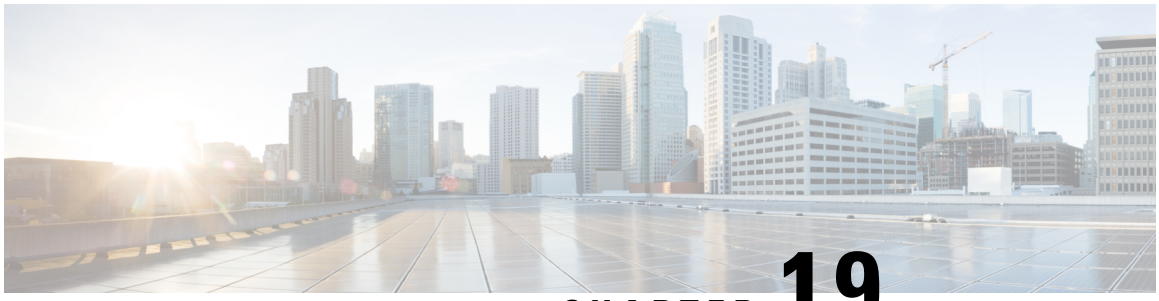
Feature Information for Contextual Configuration Diff Utility

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 25: Feature Information for Contextual Configuration Diff Utility

Feature Name	Releases	Feature Information
Contextual Configuration Diff Utility	Cisco IOS XE Release 2.1	The Contextual Configuration Diff Utility feature provides the ability to perform a line-by-line comparison of any two configuration files and generate a list of the differences between them. The generated output includes information regarding configuration lines that have been added, modified, or deleted, and the configuration modes within which a changed configuration line exists. In Cisco IOS XE Release 2.1, this feature was introduced on Cisco ASR 1000 Series Routers. The following commands were modified by this feature: **show archive config differences**, **show archive config incremental-diffs**.



CHAPTER 19

Configuration Change Notification and Logging

The Configuration Change Notification and Logging (Config Log Archive) feature allows the tracking of configuration changes entered on a per-session and per-user basis by implementing an archive function. This archive saves *configuration logs* that track each configuration command that is applied, who applied the command, the parser return code (PRC) for the command, and the time the command was applied. This feature also adds a notification mechanism that sends asynchronous notifications to registered applications whenever the configuration log changes.

Before the introduction of the Configuration Change Notification and Logging feature, the only way to determine if the Cisco software configuration had changed was to save a copy of the running and startup configurations to a local computer and do a line-by-line comparison. This comparison method can identify changes that occurred, but does not specify the sequence in which the changes occurred, or the person responsible for the changes.

Restrictions for Configuration Change Notification and Logging

- Only complete commands input in a configuration mode are logged.
- Commands that are part of a configuration file applied with the **copy** command are not logged.

Information About Configuration Change Notification and Logging

Configuration Log

The Configuration Change Notification and Logging feature tracks changes made to the Cisco software running configuration by maintaining a configuration log. This configuration log tracks changes initiated only through

the CLI or HTTP. Only complete commands that result in the invocation of action routines are logged. The following types of entries are not logged:

- Commands that result in a syntax error message
- Partial commands that invoke the device help system

For each configuration command that is executed, the following information is logged:

- The command that was executed
- The configuration mode in which the command was executed
- The name of the user that executed the command
- The time at which the command was executed
- A configuration change sequence number
- Parser return codes for the command

You can display information from the configuration log by using the **show archive log config** command, with the exception of the parser return codes, which are for use by internal Cisco applications only.

Configuration Change Notifications and Config Change Logging

You can configure the Configuration Change and Notification Logging feature to send notification of configuration changes to the software system logging (syslog) process. Syslog notifications allow monitoring of the configuration log information without performing polling and information gathering tasks.

The Configuration Change Notification and Logging feature allows the tracking of configuration changes entered by users on a per-session and per-user basis. This tool allows administrators to track any configuration change made to the software running configuration, and identify the user that made that change.

Config Logger Enhancements for EAL4+ Certification

The Config Logger Enhancements for EAL4+ Certification feature ensures that the logging process meets the requirements set forth in the Conformance to Common Criteria, Evaluation Assurance Level 4+ (EAL4+) Firewall Protection Profiles. These enhancements include changes to meet the following requirements:

- If you change any logging parameters, those changes are logged. This is effected by the sending of a syslog message for each change to the running configuration from a copy operation (for example, **copy** *source* **running-config**).
- Modifications to the group of administrative users are logged; failure attempts for access to privileged EXEC mode (“enable” mode) are logged.

Note EAL Certification is not claimed by Cisco. These enhancements provide the groundwork for future certification.

The logging actions described above are disabled by default. To enable these logging characteristics, perform the task described in the “Configuring the Configuration Change Notification and Logging Feature” section in the "Configuration Change Notification and Logging" feature module.

How to Configure Configuration Change Notification and Logging

Configuring Configuration Change Notification and Logging

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging enable**
6. **logging size** *entries*
7. **hidekeys**
8. **notify syslog**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**log config** **Example:** `Device(config-archive)# log config`	Enters configuration change logger configuration mode.
Step 5	**logging enable** **Example:** `Device(config-archive-log-config)# logging enable`	Enables the logging of configuration changes. • Logging of configuration changes is disabled by default.

	Command or Action	Purpose
Step 6	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 200`	(Optional) Specifies the maximum number of entries retained in the configuration log. • Valid values for the *entries* argument range from 1 to 1000. The default value is 100 entries. • When the configuration log is full, the oldest entry is deleted every time a new entry is added. **Note** If a new log size is specified that is smaller than the current log size, the oldest log entries are immediately purged until the new log size is satisfied, regardless of the age of the log entries.
Step 7	**hidekeys** **Example:** `Device(config-archive-log-config)# hidekeys`	(Optional) Suppresses the display of password information in configuration log files. **Note** Enabling the **hidekeys** command increases security by preventing password information from being displayed in configuration log files.
Step 8	**notify syslog** **Example:** `Device(config-archive-log-config)# notify syslog`	(Optional) Enables the sending of notifications of configuration changes to a remote syslog.
Step 9	**end** **Example:** `Device(config-archive-log-config)# end`	Returns to privileged EXEC mode.

Displaying Configuration Log Entries and Statistics

Perform this task to display entries from the configuration log or statistics about the memory usage of the configuration log. You can enter the commands in any order.

To display configuration log entries and to monitor the memory usage of the configuration log, the Configuration Change Notification and Logging feature provides the **show archive log config** command.

SUMMARY STEPS

1. **enable**
2. **show archive log config** *number* [*end-number*]
3. **show archive log config all provisioning**
4. **show archive log config statistics**
5. **exit**

DETAILED STEPS

Step 1 **enable**

Use this command to enable privileged EXEC mode. Enter your password if prompted. For example:

Example:

```
Device> enable
```

Step 2 **show archive log config** *number* [*end-number*]

Use this command to display configuration log entries by record numbers. If you specify a record number for the optional *end-number* argument, all log entries with record numbers in the range from the value entered for the *number* argument through the *end-number* argument are displayed. For example:

```
Device# show archive log config 1 2

idx   sess   user@line        Logged command
 1     1     user1@console    logging enable
 2     1     user1@console    logging size 200
```

Example:

This example displays configuration log entry numbers 1 and 2. The range for the *number* and *end-number* arguments is 1 to 2147483647.

Step 3 **show archive log config all provisioning**

Use this command to display all configuration log files as they would appear in a configuration file rather than in tabular format. For example:

Example:

```
Device# show archive log config all provisioning

archive
 log config
  logging enable
  logging size 200
```

This display also shows the commands used to change configuration modes, which are required to correctly apply the logged commands.

Step 4 **show archive log config statistics**

Use this command to display memory usage information for the configuration. For example:

Example:

```
Device# show archive log config statistics

Config Log Session Info:
   Number of sessions being tracked: 1
   Memory being held: 3910 bytes
   Total memory allocated for session tracking: 3910 bytes
   Total memory freed from session tracking: 0 bytes
Config Log log-queue Info:
   Number of entries in the log-queue: 3
   Memory being held in the log-queue: 671 bytes
```

```
Total memory allocated for log entries: 671 bytes
Total memory freed from log entries:: 0 bytes
```

Step 5 **exit**

Use this command to exit to user EXEC mode. For example:

Example:

```
Device# exit
Device>
```

Clearing Configuration Log Entries

Entries from the configuration log can be cleared in one of two ways. The size of the configuration log can be reduced by using the **logging size** command, or the configuration log can be disabled and then reenabled with the **logging enable** command.

Clearing the Configuration Log by Resetting the Log Size

This task shows how to clear the configuration log by reducing the log size to 1, then resetting the log size to the desired value, by entering the **logging size** command twice.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging size** *entries*
6. **logging size** *entries*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:**	Enters archive configuration mode.

	Command or Action	Purpose
	`Device(config)# archive`	
Step 4	**log config** **Example:** `Device(config-archive)# log config`	Enters configuration change logger configuration mode.
Step 5	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 1`	Specifies the maximum number of entries retained in the configuration log. **Note** Setting the size of the configuration log to 1 results in all but the most recent entry being purged.
Step 6	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 200`	Specifies the maximum number of entries retained in the configuration log. **Note** The size of the configuration log should be reset to the desired value after clearing the configuration log.
Step 7	**end** **Example:** `Device(config-archive-log-config)# end`	Exits to privileged EXEC mode.

Clearing the Configuration Log by Disabling the Configuration Log

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **no logging enable**
6. **logging enable**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**log config** **Example:** `Device(config-archive)# log config`	Enters configuration change logger configuration mode.
Step 5	**no logging enable** **Example:** `Device(config-archive-log-config)# no logging enable`	Disables the logging of configuration changes. **Note** Disabling the configuration log results in all records being purged.
Step 6	**logging enable** **Example:** `Device(config-archive-log-config)# logging enable`	Enables the logging of configuration changes.
Step 7	**end** **Example:** `Device(config-archive-log-config)# end`	Exits to privileged EXEC mode.

Automatic Log Deletion

This feature allows you to delete the entries from the logging buffer automatically after a configurable time. You must configure the local syslog retention period after which the entries are purged from the device. To automatically purge the logging data after a given time, use the **logging purge-log buffer days x time <x:y>** command. The maximum retention time for log entries can be configured in a unit of days with a range of 1-120 days. The feature also allows one buffer clean up per day, which will clean up the buffer log based on the configured duration every 24 hours.

Note If the command specifies retention time only in days, then the deletion of logs occurs the following day at the same time as the command was configured.

To configure automatic log deletion, perform these steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging purge-log buffer days entries**
4. **logging purge-log buffer days x time <x:y>**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device > enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device # configure terminal`	Enters global configuration mode.
Step 3	**logging purge-log buffer days entries** **Example:** `Device(config)#logging purge-log buffer days 90`	Specifies the maximum retention time for log entries. **Note** Valid values range from 1 to 120.
Step 4	**logging purge-log buffer days x time <x:y>** **Example:** `Device(config)#logging purge-log buffer days 90 time 15:45`	(Optional) Specifies the particular time for automated deletion of logs. **Note** • The logs are deleted. • If the time is less than the current system time, then deletion happens the following day at the time provided.
Step 5	**end** **Example:** `Device(config)# end`	Exits to privileged EXEC mode.

Configuration Examples for Automatic Log Deletion

The following example shows how to enable automatic log deletion to retain only 90 days old data. The deletion of logs will take place at the specified time, which is 15:45.

```
Router (config)# logging purge-log buffer days 90 time 15:45
*May 18 20:20:20 UTC: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration
change requiring running configuration sync detected - ' logging purgelog
buffer days 90 time 15:45
'. The running configuration will be sy
nchronized to the NETCONF running data store.
◦ May 18 20:20:21 UTC: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization
of the running configuration to the NETCONF running data store has
```

```
started.
May 18 20:20:26 UTC: %DMI-5-SYNC_COMPLETE: R0/0: dmiauthd: The running
configuration has been synchronized to the NETCONF running data store.
```

The following example shows how to enable automatic log deletion to retain only 10 days old data and delete the remaining logs from buffer

```
Router(config)# logging purge-log buffer days 10
Jul  5 19:48:16.974: %PARSER-5-CFGLOG_LOGGEDCMD: User:test  logged command:logging purge-log
 buffer days 10
*Jul  5 19:48:17.330: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration change requiring
running configuration sync detected - '  logging purge-log buffer days 10'.
The running configuration will be synchronized  to the NETCONF running data store.
*Jul  5 19:48:17.451: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization of the running
configuration to the NETCONF running data store has started.
```

Sample output for the **no logging purge-log buffer** command.

```
Router(config)# no logging purge-log buffer
Jul  5 19:49:29.601: %PARSER-5-CFGLOG_LOGGEDCMD: User:test  logged command:no logging
purge-log buffer
*Jul  5 19:49:29.980: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration change requiring
running configuration sync detected - '  no logging purge-log buffer '.
The running configuration will be synchronized  to the NETCONF running data store.
*Jul  5 19:49:30.110: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization of the running
configuration to the NETCONF running data store has started.
```

Configuration Examples for Configuration Change Notification and Logging

Example: Configuring Configuration Change Notification and Logging

The following example shows how to enable configuration logging with a maximum of 200 entries in the configuration log. In the example, security is increased by suppressing the display of password information in configuration log records with the **hidekeys** command, and syslog notifications are turned on with the **notify syslog** command.

```
configure terminal
archive
 log config
 logging enable
 logging size 200
 hidekeys
 notify syslog
```

Additional References

Related Documents

Related Topic	Document Title
Information about managing configuration files	"Managing Configuration Files" module in the *Managing Configuration Files Configuration Guide*
Commands for managing configuration files	Cisco IOS Configuration Fundamentals Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

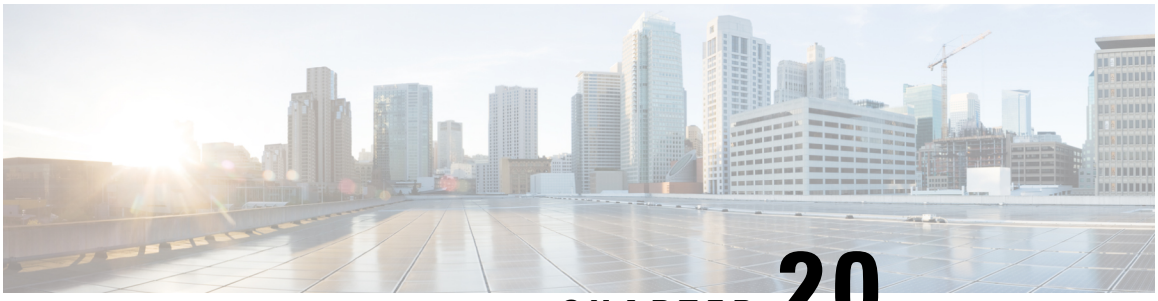
Feature Information for Configuration Change Notification and Logging

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 26: Feature Information for Configuration Change Notification and Logging

Feature Name	Releases	Feature Information
Configuration Change Notification and Logging		The Configuration Change Notification and Logging (Configuration Logging) feature allows the tracking of configuration changes entered on a per-session and per-user basis by implementing a configuration log. The configuration log tracks each configuration command that is applied, who applied the command, the parser return code for the command, and the time the command was applied. This feature also adds a notification mechanism that sends asynchronous notifications to registered applications whenever the configuration log changes. The following commands were introduced or modified: **archive**, **hidekeys**, **log config**, **logging enable**, **logging size**, **notify syslog**, **show archive log config**.
Support for Automatic Log Deletion	Cisco IOS XE Dublin 17.12.1a	This feature allows you to delete the entries from the logging buffer. You can configure the local syslog retention period after which the entries are purged from the device automatically. To enable this feature, use the **logging purge-log buffer days** command.



CHAPTER 20

Configuration Partitioning

The Configuration Partitioning feature provides modularization (“partitioning”) of the running configuration state to provide granular access to the running configuration in Cisco IOS software.

This feature is enabled by default in Cisco IOS software images that include this feature.

The configuration state of a device is retrieved dynamically whenever a user issues the **showrunning-config** command. When the Configuration Partitioning feature is enabled, the system groups the configuration state of the device into parts (called “partitions”) so that only the configuration state the user wishes to review is retrieved when generating a displayed list of commands in the running configuration. This feature improves performance for high-end systems with complex configurations because only a part of the running configuration state is processed when generating the running configuration command list, as opposed to the existing method of processing the entire system configuration state.

Default configuration partitions are provided by the introduction of this feature; other Cisco IOS software features may define their own command partitions in later releases.

Information About Configuration Partitioning

System Running Configurations

Managing the configuration of any Cisco IOS software-based device involves managing the startup configuration (startup-config), which is a file stored in nonvolatile memory, and the running configuration (running-config), which is the set of all configuration options currently in effect on the system. Typically, the startup configuration file is loaded when the system boots, and changes to the system’s running configuration, applied using the command-line interface (CLI), are saved by copying the running configuration to a configuration file (either locally or on the network), which can then be used to configure the device at startup, or used to configure other devices.

Retrieving the Running Configuration for Display or Copy Operations

In the Cisco IOS software configuration model, the configuration state is maintained in a distributed manner, with each component storing its own configuration state. To retrieve global configuration information, the software must poll every component to collect the distributed information. This configuration state retrieval operation is performed by a process known as nonvolatile generation (NVGEN), and it is invoked by commands such as **showrunning-config**, which is used to display the current configuration state, and **copysystem:running-configuration**, which is used to save the running configuration by copying it to a file. When invoked, the NVGEN process queries each system component, each interface instance, and all other configured component objects in a standard sequence. A running configuration file is constructed as NVGEN traverses the system performing these queries, and it is this "virtual file" that is displayed or copied.

Benefits of Partitioning the Running Configuration

The Configuration Partitioning feature is the latest in a series of Configuration Generation Performance Enhancement Features for Cisco IOS software. (See the "Related Documents" section for related features.) This feature improves the system's response time by providing a method for querying only the system component you wish to review when issuing the **showrunning-config** command.

When the Configuration Partitioning feature is enabled, the system groups the configuration state of the device into parts (called "partitions") for the purpose of generating the virtual running configuration file (the list of configuration commands). A new command, **showrunning-configpartition**, allows you to display only the part of the running configuration that you want to examine, rather than having to display the entire running configuration at once, or displaying only lines that match a certain string.

The key benefit of this feature is that it increases system performance by allowing the system to run the NVGEN process for only the collection of system components (such as specific interfaces) that you need to display. This is in contrast to other existing extensions to the **showrunning-config** command, which only filter the generated list after all system components have been processed.

The selective processing of the system's configuration state for the purpose of generating a partial running configuration is called "configuration partitioning."

More granular access to configuration information offers important performance benefits for high-end routing platforms with very large configuration files, while also enhancing configuration management by allowing advanced configuration features to be implemented at a more granular level. Advanced configuration options include Cisco IOS software support for provisioning of customer services, Config Rollback, Config Locking, and configuration access control.

How to Use the Configuration Partitioning Feature

Displaying Configuration Partitions

The main method of taking advantage of this feature is by using the **showrunning-configpartition***part*command in priveledge exec mode, which is a specialized extension to the **showrunning-config**command.

Note The **partition***part* command extension is not available for the **more:systemrunning-config** command.

Because this feature offers improved performance for existing commands, this feature is enabled by default in Cisco IOS software images that support this feature. To quickly determine if this feature is supported and running on your system, issue the **showrunning-configpartition?** command in privileged EXEC mode.

SUMMARY STEPS

1. **show running-config partition ?**
2. **show running-config partition** *part*

DETAILED STEPS

Step 1 show running-config partition ?

Issuing this command will show you the list of running configuration parts available for display on your system.

If the Configuration Partitioning feature is supported on your system and is enabled, you will see the string " config partition is TRUE " as the first line of help output.

If you receive an error message when entering the command syntax shown here, this feature is not supported on your system. See the command documentation for the **showrunning-config** command for existing extensions of that command in other releases that allow you to show only part of the running configuration.

Note The list of available configuration parts may vary by software image and is dependent on what features are currently configured.

Example:

```
Router# show running-config partition ?
 config partition is TRUE
  access-list       All access-list configurations
  boot              All boot configurations
  class-map         All class-map configurations
  common            All remaining unregistered configurations
  global-cdp        All global cdp configurations
  interface         All Interface specific Configurations
  ip-as-path        All IP as-path configurations
  ip-community      All IP community list configurations
  ip-domain-list    All ip domain list configurations
  ip-prefix-list    All ip prefix-list configurations
  ip-static-routes  All IP static configurations
  line              All line mode configurations
  policy-map        All policy-map configurations
  route-map         All route-map configurations
  router            All routing configurations
  snmp              All SNMP configurations
  tacacs            All TACACS configurations
```

Choose the part of the running configuration you want to display, and use the associated keyword as the *part* argument in Step 2.

Step 2 show running-config partition *part*

As an example, to have the system perform the NVGEN process on only the components associated with the access-list parts of the running configuration state, and display only the access-list related configurations, you would enter the **showrunning-configpartitionaccess-list** command:

Example:

```
Router# show running-config partition access-list
        Building configuration...
Current configuration : 127 bytes
!
Configuration of Partition access-list
!
!
!
access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
!
end
```

Note This command also allows you to run the NVGEN process and display the resulting output for specific interfaces. This is a key capability of this feature, as it was designed for systems with numerous active interfaces.

In the following example, the main configuration partition is the interface configuration, and the specific part of the configuration to be generated is the configuration for Fast Ethernet interface 0/0.

Example:

```
Router# show running-config partition interface fastethernet0/0
Building configuration...
Current configuration : 213 bytes
!
Configuration of Partition interface FastEthernet0/0
!
!
interface FastEthernet0/0
 ip address 10.4.2.39 255.255.255.0
 no ip route-cache cef
 no ip route-cache
 duplex half
 ipv6 enable
 no cdp enable
!
!
end
```

Disabling the Configuration Partitioning Feature

Because this feature offers improved performance for existing commands, this feature is enabled by default for Cisco IOS software images that support this feature. However, you may want to disable this feature if you determine that it is not needed, as this feature does use a small amount of system resources (memory and CPU utilization). To disable configuration partitioning, perform the following task, which assumes you are starting in user EXEC mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no parser config partition**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no parser config partition** **Example:** `Router(config)# no parser config partition` **Example:** `Disabling config partitioning` **Example:** `Router(config)#`	Disables the configuration partitioning feature.

What to do next

What to Do Next

To reenable the feature after it has been disabled, use the **parserconfigpartition** command in global configuration mode.

Note As this feature is enabled by default, only the **no** form will appear in the running configuration file, or will be written to the startup configuration file when you issue the **copyrunning-configstartup-config**command.

Configuration Examples for Configuration Partitioning

This section provides examples of displaying configuration partitions with the **show running-config partition** command:

Displaying Configuration Partitions Example

In this example, the **showrunning-configpartition**command is used with related commands in a series of steps an administrator might take to check the status of a specific interface and the current configuration of some of the system's other components. Comparable filtered output from the standard

showrunning-configcommand (for example, **showrunning-config|includeaccess-list**) is included for demonstration purposes.

Note The *part* argument can consist of multiple partition name keywords, as in **showrunning-configpartroutereigrp1**.

```
gt3-7200-3# show running-config partition ?
  access-list       All access-list configurations
  boot              All boot configurations
  class-map         All class-map configurations
  global-cdp        All global cdp configurations
  interface         All Interface specific Configurations
  ip-as-path        All IP as-path configurations
  ip-community      All IP community list configurations
  ip-domain-list    All ip domain list configurations
  ip-static-routes  All IP static configurations
  line              All line mode configurations
  policy-map        All policy-map configurations
  route-map         All route-map configurations
  router            All routing configurations
  service           All service configurations
  snmp              All SNMP configurations
gt3-7200-3# show running-config partition access-list

Building configuration...
Current configuration : 87 bytes
!
!
!
!
access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
!
end
gt3-7200-3# show running-config | include access-list

access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
gt3-7200-3#
gt3-7200-3# show running-config partition boot

Building configuration...
Current configuration : 51 bytes
!
boot network tftp:/service_config.txt
!
!
!
end
gt3-7200-3# show running-config partition class-map

Building configuration...
Current configuration : 78 bytes
!
!
!
class-map match-all abc
  match any
class-map match-all xyz
```

```
!
!
!
end
gt3-7200-3# show running-config | begin class-map

class-map match-all abc
  match any
class-map match-all xyz
!
!
gt3-7200-3# show running-config partition global-cdp

Building configuration...
Current configuration : 43 bytes
!
!
!
cdp timer 20
cdp holdtime 100
!
end
gt3-7200-3# show running-config | include

global-cdp

cdp timer 20
cdp holdtime 100
gt3-7200-3#
gt3-7200-3# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES NVRAM  administratively down down
Ethernet2/0                10.4.2.32       YES NVRAM  up                    up
Ethernet2/1                unassigned      YES NVRAM  administratively down down
Ethernet2/2                unassigned      YES NVRAM  administratively down down
Ethernet2/3                unassigned      YES NVRAM  administratively down down
Serial3/0                  unassigned      YES NVRAM  administratively down down
Serial3/1                  unassigned      YES NVRAM  administratively down down
Serial3/2                  unassigned      YES NVRAM  administratively down down
Serial3/3                  unassigned      YES NVRAM  administratively down down
Loopback0                  unassigned      YES NVRAM  administratively down down
Loopback234                unassigned      YES NVRAM  administratively down down
gt3-7200-3# show running-config partition interface fastethernet0/0
Building configuration...
Current configuration : 98 bytes
!
!
!
interface FastEthernet0/0
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/0

Building configuration...
Current configuration : 122 bytes
!
!
!
interface Ethernet2/0
```

```
 ip address 10.4.2.32 255.255.255.0
 no ip proxy-arp
 no ip route-cache
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/1
Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/1
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/2

Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/2
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/3
Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/3
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface serial3/0
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/0
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
```

```
gt3-7200-3# show running-config partition interface serial3/1
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/1
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface serial3/2
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/2
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface serial3/3
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/3
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface loopback0
Building configuration...
Current configuration : 79 bytes
!
!
!
interface Loopback0
 no ip address
 no ip route-cache
 shutdown
!
!
end
gt3-7200-3# show running-config partition interface loopback1
                                                            ^
% Invalid input detected at '^' marker.
gt3-7200-3# show running-config partition interface loopback234
Building configuration...
Current configuration : 81 bytes
!
!
!
```

```
interface Loopback234
 no ip address
 no ip route-cache
 shutdown
!
!
end
gt3-7200-3# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
gt3-7200-3(config)# interface ethernet 2/0.1
gt3-7200-3(config-subif)# exit
gt3-7200-3(config)# exit
gt3-7200-3#
00:13:05: %SYS-5-CONFIG_I: Configured from console by console
gt3-7200-3# show running-config partition interface ethernet2/0.1
Building configuration...
Current configuration : 58 bytes
!
!
!
interface Ethernet2/0.1
 no ip route-cache
!
!
end
gt3-7200-3# show run partition ip?
ip-as-path  ip-community  ip-domain-list  ip-static-routes
gt3-7200-3#sh run part ip-as
gt3-7200-3#sh run part ip-as-path

Building configuration...
Current configuration : 125 bytes
!
!
!
ip as-path access-list 2 permit $ABC
ip as-path access-list 2 permit $xyz*
ip as-path access-list 2 permit qwe*
!
end
gt3-7200-3# show running-config partition ip-community

Building configuration...
Current configuration : 92 bytes
!
!
!
ip community-list standard asd permit
ip community-list expanded qwe deny uio*
!
end
gt3-7200-3# show running-config | include ip community
ip community-list standard asd permit
ip community-list expanded qwe deny uio*
gt3-7200-3#
gt3-7200-3# show running-config partition ip-domain-list

Building configuration...
Current configuration : 70 bytes
!
ip domain-list iop
ip domain-list tyu
ip domain-list jkl
```

```
!
!
!
end
gt3-7200-3# show running-config partition
 ip-static-routes

Building configuration...
Current configuration : 98 bytes
!
!
!
ip route 0.0.0.0 0.0.0.0 Ethernet2/0
ip route 171.69.1.129 255.255.255.255 10.4.29.1
!
end
gt3-7200-3# show running-config partition line
Building configuration...
Current configuration : 489 bytes
!
!
!
!
line con 0
 exec-timeout 0 0
 transport output lat pad v120 mop telnet rlogin udptn nasi
 stopbits 1
line aux 0
 transport output lat pad v120 mop telnet rlogin udptn nasi
 stopbits 1
line vty 0
 password lab
 login
 transport input lat pad v120 mop telnet rlogin udptn nasi
 transport output lat pad v120 mop telnet rlogin udptn nasi
line vty 1 4
 login
 transport input lat pad v120 mop telnet rlogin udptn nasi
 transport output lat pad v120 mop telnet rlogin udptn nasi
!
end
gt3-7200-3# show running-config partition policy-map
Building configuration...
Current configuration : 162 bytes
!
!
!
policy-map qwer
 description policy-map qwer.
  class xyz
    shape peak 8000 32 32
policy-map p1
policy-map sdf
  class abc
   set precedence 4
!
!
!
end
gt3-7200-3# show running-config partition route-map
Building configuration...
Current configuration : 65 bytes
!
!
```

```
!
route-map iop permit 10
!
route-map rty permit 10
!
!
end
gt3-7200-3#sh run part router bgp 1
Building configuration...
Current configuration : 111 bytes
!
!
!
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 distance bgp 2 2 2
 no auto-summary
!
!
end
gt3-7200-3#sh run part router egp ?
<0-65535>  Remote autonomous system number
gt3-7200-3#sh run part router egp 1
Building configuration...
Current configuration : 46 bytes
!
!
!
router egp 1
 timers egp 20 20
!
!
end
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router eigrp ?
<1-65535>  Autonomous system number
gt3-7200-3# show running-config partition router eigrp 1
Building configuration...
Current configuration : 13 bytes
!
!
!
!
end
gt3-7200-3#
gt3-7200-3# sh run part router eigrp 2

Building configuration...
Current configuration : 57 bytes
!
!
!
router eigrp 2
 variance 10
```

```
 auto-summary
!
!
end
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router isis ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router isis qwe
Building configuration...
Current configuration : 86 bytes
!
!
!
router isis qwe
 set-attached-bit route-map qwer
 use external-metrics
!
!
end
gt3-7200-3# show running-config partition router isis ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router iso
gt3-7200-3# show running-config partition router iso-igrp ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router iso-igrp

Building configuration...
Current configuration : 31 bytes
!
!
!
router iso-igrp
!
!
end
gt3-7200-3# show running-config | begin iso
router iso-igrp
!
router isis qwe
 set-attached-bit route-map qwer
 use external-metrics
!
router egp 1
 timers egp 20 20
!
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 distance bgp 2 2 2
```

```
 no auto-summary
!

gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router mobile ?
  |  Output modifiers
  <cr>
gt3-7200-3# show running-config partition router mobile

Building configuration...
Current configuration : 42 bytes
!
!
!
router mobile
 distance 20
!
!
end
gt3-7200-3# sh run | include router

router mobile
router odr
router eigrp 2
router ospf 4
router iso-igrp
router isis qwe
router egp 1
router bgp 1
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router ospf ?
  <1-65535>  Process ID
gt3-7200-3# show running-config partition router ospf 4
Building configuration...
Current configuration : 64 bytes
!
!
!
router ospf 4
 log-adjacency-changes
 distance 4
!
!
end
gt3-7200-3# sh run part service
```

```
Building configuration...
Current configuration : 190 bytes
!
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service internal
service udp-small-servers
service tcp-small-servers
!
!
!
end
gt3-7200-3# sh run part snmp

Building configuration...
Current configuration : 84 bytes
!
!
!
snmp-server community user101 RW
snmp mib target list qwe host 0.0.0.0
!
end
```

Additional References

The following sections provide references related to the Configuration Partitioning feature.

Related Documents

Related Topic	Document Title
Running configuration performance enhancement--**parserconfigcache** for interfaces.	Configuration Generation Performance Enhancement
Provisioning of customer services, Config Rollback, Config Locking, and configuration access control	Contextual Configuration Diff Utility
Configuration management--Config change logging.	Configuration Change Notification and Logging
Configuration management --Quick-save for config change logging[3].	Configuration Logger Persistency
Cisco IOS software configuration access control and config session locking ("Config Lock").	Exclusive Configuration Change Access and Access Session Locking

[3] The "Configuration Logger Persistency" feature allows saving just the commands entered since the last startup-config file was generated, rather than saving the entire startup configuration.

Standards

Standard	Title
No standards are associated with this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	--

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password..	http://www.cisco.com/techsupport

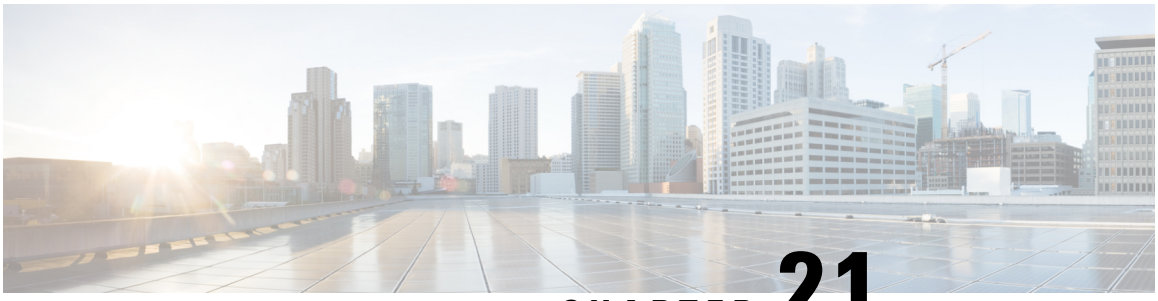
Feature Information for Configuration Partitioning

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 27: Feature Information for Configuration Partitioning

Feature Name	Releases	Feature Information
Configuration Partitioning	12.2(33)SRB 12.2(33)SB 12.2(33)SXI	The Configuration Partitioning feature provides modularization (“partitioning”) of the running configuration state to provide granular access to the running configuration in Cisco IOS software. This feature is enabled by default in Cisco IOS software images that include this feature. In 12.2(33)SB, this feature was implemented on the Cisco 10000 series. The following sections provide information about this feature: • Information About Configuration Partitioning • How to Use the Configuration Partitioning Feature



CHAPTER 21

Configuration Versioning

The Configuration Versioning feature allows you to maintain and manage backup copies of the Cisco running configuration on or off the device. The Configuration Replace feature uses the Configuration Versioning feature to provide a rollback to a saved copy of the running configuration.

Information About Configuration Versioning

Configuration Archive

The Cisco configuration archive provides a mechanism to store, organize, and manage an archive of Cisco configuration files to enhance the configuration rollback capability provided by the **configure replace** command. Before this feature was introduced, you could save copies of the running configuration using the **copy running-config** *destination-url* command, storing the replacement file either locally or remotely. However, this method lacked any automated file management. With the Configuration Replace and Configuration Rollback feature, you can automatically save copies of the running configuration to the configuration archive. These archived files serve as checkpoint configuration references and can be used by the **configure replace** command to revert the configuration to a previous state.

The **archive config** command allows you to save Cisco configurations in the configuration archive using a standard location and filename prefix that is automatically appended with an incremental version number (and optional time stamp) as each consecutive file is saved. This functionality provides consistent identification of saved configuration files. You can specify how many versions of the running configuration are kept in the archive. After the maximum number of files are saved in the archive, the oldest file is automatically deleted when the next, most recent file is saved. The **show archive** command displays information for all configuration files saved in the configuration archive.

The configuration archive, in which the configuration files are stored and available for use with the **configure replace** command, can be located on the following file systems depending on your platform:

- If your platform has disk0:—disk0:, disk1:, ftp:, pram:, rcp:, slavedisk0:, slavedisk1:, or tftp:
- If your platform does not have disk0:—bootflash:, ftp:, harddisk:, http:, pram:, rcp:, tftp:, usb0:, or usb1:

How to Configure Configuration Versioning

Configuring the Characteristics of the Configuration Archive

Before using the **archive config** command, the configuration archive must be configured. Perform this task to configure the characteristics of the configuration archive.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **path** *url*
5. **maximum** *number*
6. **time-period** *minutes*
7. **end**
8. **archive config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**path** *url* **Example:** `Device(config-archive)# path bootflash:myconfig`	Specifies the location and filename prefix for the files in the configuration archive. • Depending on your hardware platform, the name of your file system might be different than shown in the example.

	Command or Action	Purpose
		Note If a directory is specified in the path instead of file, the directory name must be followed by a forward slash as follows: path flash:/*directory*/. The forward slash is not necessary after a filename; it is necessary only when specifying a directory.
Step 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	(Optional) Sets the maximum number of archive files of the running configuration to be saved in the configuration archive. • The *number* argument is the maximum number of archive files of the running configuration to be saved in the configuration archive. The range is 1 to 14. The default is 10. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the configuration archive.
Step 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 10`	(Optional) Sets the time increment for automatically saving an archive file of the current running configuration in the configuration archive. • The *minutes* argument specifies how often, in minutes, to automatically save an archive file of the current running configuration in the configuration archive. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the configuration archive.
Step 7	**end** **Example:** `Device(config-archive)# end`	Exits to privileged EXEC mode.
Step 8	**archive config** **Example:** `Device# archive config`	Saves the current running configuration file to the configuration archive. **Note** You must configure the **path** command before using the **archive config** command.

Monitoring and Troubleshooting the Configuration

SUMMARY STEPS

1. **enable**
2. **show archive**
3. **debug archive versioning**
4. **debug archive config timestamp**
5. **exit**

DETAILED STEPS

Step 1 enable

Use this command to enable privileged EXEC mode. Enter your password if prompted. For example:

Example:

```
Device> enable
Device#
```

Step 2 show archive

Use this command to display information about the files saved in the configuration archive. For example:

Example:

```
Device# show archive

There are currently 1 archive configurations saved.
The next archive file will be named bootflash:myconfig-2
 Archive #  Name
   0
   1       bootflash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

The following is sample output from the **show archive** command after several archive files of the running configuration have been saved. In this example, the maximum number of archive files to be saved is set to three.

Example:

```
Device# show archive

There are currently 3 archive configurations saved.
The next archive file will be named bootflash:myconfig-8
```

```
 Archive #  Name
   0
   1       :Deleted
   2       :Deleted
   3       :Deleted
   4       :Deleted
   5       bootflash:myconfig-5
   6       bootflash:myconfig-6
   7       bootflash:myconfig-7 <- Most Recent
   8
   9
   10
   11
   12
   13
   14
```

Step 3 **debug archive versioning**

Use this command to enable debugging of the configuration archive activities to help monitor and troubleshoot configuration replace and rollback. For example:

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file bootflash:myconfig-7
Jan  9 06:46:29.547: backup worked
```

Step 4 **debug archive config timestamp**

Use this command to enable debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled. For example:

Example:

```
Device# debug archive config timestamp
Device# configure replace bootflash:myconfig force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file slot0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

Step 5 **exit**

Use this command to exit to user EXEC mode. For example:

Example:

```
Device# exit
Device>
```

Configuration Examples for Configuration Versioning

Example: Creating a Configuration Archive

The following example shows how to perform the initial configuration of the configuration archive. In this example, bootflash:myconfig is specified as the location and filename prefix for the files in the configuration archive and a value of 10 is set as the maximum number of archive files to be saved. Depending on your hardware platform, the name of your file system might be different than shown in the example.

```
configure terminal
!
archive
 path bootflash:myconfig
 maximum 10
 end
```

Additional References

Related Documents

Related Topic	Document Title
Information about managing configuration files	"Managing Configuration Files" module in the *Managing Configuration Files Configuration Guide*
Commands for managing configuration files	Cisco IOS Configuration Fundamentals Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

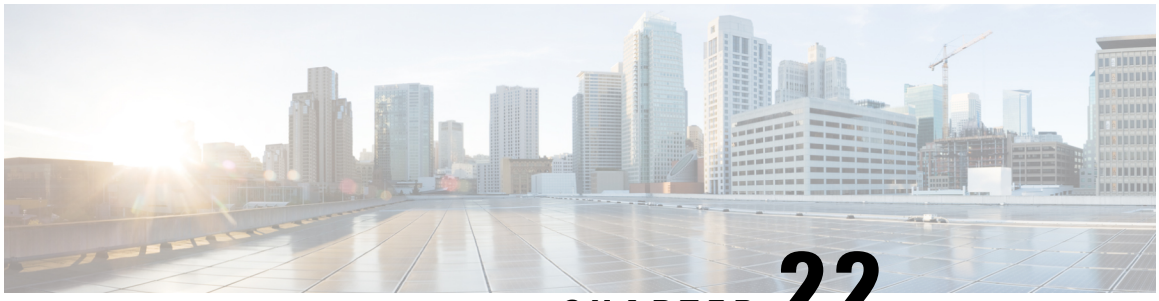
Feature Information for Configuration Versioning

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 28: Feature Information for Configuration Versioning

Feature Name	Releases	Feature Information
Configuration Versioning	12.2(25)S 12.2(33)SRA 12.3(7)T Cisco IOS XE Release 2.1	The Configuration Versioning feature allows you to maintain and manage backup copies of the Cisco running configuration on or off the device. The Configuration Replace feature uses the Configuration Versioning feature to provide a rollback to a saved copy of the running configuration. The following commands were introduced or modified: **archive config**, **debug archive versioning**, **log config**, **maximum**, **path** (archive configuration), **show archive**, **time-period**, **write-memory**.



CHAPTER 22

Configuration Rollback Confirmed Change

The Configuration Rollback Confirmed Change feature allows configuration changes to be performed with an optional requirement that they be confirmed. If this confirmation is not received, the configuration is returned to the state prior to the changes being applied. The mechanism provides a safeguard against inadvertent loss of connectivity between a network device and the user or management application due to configuration changes.

Information About Configuration Rollback Confirmed Change

Configuration Rollback Confirmed Change Operation

The Configuration Rollback Confirmed Change feature enables an added criterion of a confirmation to configuration changes. This functionality enables a rollback to occur if a confirmation of the requested changes is not received in a configured time frame. Command failures can also be configured to trigger a configuration rollback.

The following steps outline how this process is achieved:

1. A new option allows you to request confirmation (a confirmation time limit must be supplied) of the configuration changes.
2. You must enter the confirmation command. If no confirmation is entered within the requested time limit, the configuration reverts to its previous state.

How to Configure Configuration Rollback Confirmed Change

Performing a Configuration Replace or Configuration Rollback Operation with Confirmation

Perform this task to replace the current running configuration file with a saved Cisco configuration file.

Note You must configure a configuration archive before performing this procedure. For detailed steps, see the "Configuring the Characteristics of the Configuration Archive" module in the *Managing Configuration Files Configuration Guide*. The following procedure details how to return to that archived configuration in the event of a problem with the current running configuration.

SUMMARY STEPS

1. **enable**
2. **configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**revert trigger** [**error**] [**timer** *minutes*] | **time** *minutes*]
3. **configure revert** {**now** | **timer** {*minutes* | **idle** *minutes*}}
4. **configure confirm**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**revert trigger** [**error**] [**timer** *minutes*]	**time** *minutes*] **Example:** `Device# configure replace bootflash:myconfig-1 list time 30`	Replaces the current running configuration file with a saved configuration file. • *target-url*—Specifies a URL (accessible by the Cisco file system) of the saved configuration file that is to replace the current running configuration, such as the configuration file created by using the **archive config** command. Depending on your hardware platform, the name of your file system might be different than shown in the example. • **nolock**—Disables the locking of the running configuration file that prevents other users from changing the running configuration during a configuration replace operation.

	Command or Action	Purpose
		• **list**—Displays a list of the command lines applied by the Cisco software parser during each pass of the configuration replace operation. The total number of passes performed is also displayed. • **force**—Replaces the current running configuration file with the specified saved configuration file without prompting you for confirmation. • **ignorecase**—Allows the configuration to ignore the case of the confirmation command. • **time** *minutes*—Specifies the time (in minutes) within which you must enter the **configure confirm** command to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command). • **revert trigger**—Sets the following triggers for reverting to the original configuration: • **error**—Reverts to the original configuration upon error. • **timer** *minutes*—Reverts to the original configuration if the specified time elapses.
Step 3	**configure revert** {**now** | **timer** {*minutes* | **idle** *minutes*}} **Example:** `Device# configure revert now`	(Optional) Cancels the timed rollback and triggers the rollback immediately or resets parameters for the timed rollback. • **now**—Triggers the rollback immediately. • **timer**—Resets the configuration revert timer. • Use the *minutes* argument with the **timer** keyword to specify a new revert time in minutes. • Use the **idle** keyword along with a time in minutes to set the maximum allowable time period of no activity before reverting to the saved configuration.
Step 4	**configure confirm** **Example:** `Device# configure confirm`	(Optional) Confirms replacement of the current running configuration file with a saved configuration file. **Note** Use this command only if the **time** *minutes* keyword and argument of the **configure replace** command are specified.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device# exit`	Exits to user EXEC mode.

Configuration Examples for Configuration Rollback Confirmed Change

Example: Performing a Configuration Replace Operation with the configure confirm Command

The following example shows the use of the **configure replace** command with the **time** *minutes* keyword and argument. You must enter the **configure confirm** command within the specified time limit to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command).

```
Device# configure replace nvram:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
Device# configure confirm
```

The following example shows the use of the **configure revert** command with the **timer** keyword. You must enter the **configure revert** command to cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback.

```
Device# configure revert timer 100
```

Additional References

Related Documents

Related Topic	Document Title
Information about managing configuration files	"Managing Configuration Files" module in the *Managing Configuration Files Configuration Guide*
Commands for managing configuration files	Cisco IOS Configuration Fundamentals Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

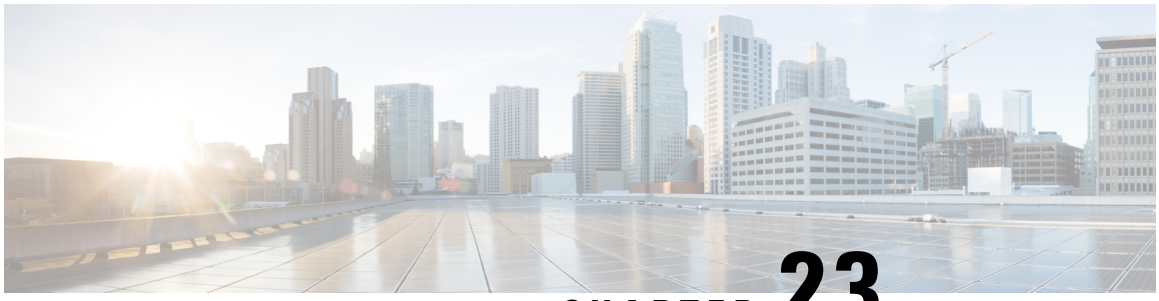
Feature Information for Configuration Rollback Confirmed Change

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 29: Feature Information for Configuration Rollback Confirmed Change

Feature Name	Releases	Feature Information
Configuration Rollback Confirmed Change	12.2(33)SB 12.2(33)SRC 12.2(33)SXI 12.4(20)T Cisco IOS XE Release 2.1	The Configuration Rollback Confirmed Change feature allows configuration changes to be performed with an optional requirement that they be confirmed. This mechanism provides a safeguard against inadvertent loss of connectivity between a network device and the user or management application due to configuration changes. The following commands were introduced or modified: **configure confirm**, **configure replace**, **configure revert**, **configure terminal**.



CHAPTER 23

Configuration Logger Persistency

The Configuration Logger Persistency feature increases the operational robustness of Cisco IOS configuration and provisioning actions by implementing a "quick-save" functionality. When the Configuration Logger Persistency feature is configured, Cisco IOS software saves just the commands entered since the last startup-config file was generated, rather than saving the entire startup configuration.

Prerequisites for Configuration Logger Persistency

To enable the Configuration Logger Persistency feature, you must have disk0: configured and an external flash card inserted on the router.

To achieve optimum results from the Configuration Logger Persistency feature, you must have Cisco IOS Release 12.2(33)SRA, Release 12.4(11)T, Release 12.2(33)SXH, or Release 12.2(33)SB installed on your system.

Information About Configuration Logger Persistency

Use of Configuration Logger Persistency to Save Configuration Files

Cisco IOS software uses the startup-config file to save router configuration commands across reloads. This single file contains all the commands that need to be applied when the router reboots. The startup-config file gets updated every time a **writememory** command or **copy***url* **startup-config** command is entered. As the size of the running-config file grows, the time to save the startup-config file to the NVRAM file system increases as well. Startup-config files can be 1 MB and larger. For files of this size, making a single-line change to the startup-config file requires that the entire startup-config file is saved again even though most of the configuration has not changed.

The Configuration Logger Persistency feature implements a “quick-save” functionality. The aim is to provide a “configuration save” mechanism where the time to save changes from the startup-config file is proportional to the size of the incremental changes (with respect to the startup-config file) that need to be saved.

The Cisco IOS configuration logger logs all changes that are manually entered at the command-line prompt. This feature also notifies the registered clients when changes to the log occur. The contents of the configuration log are stored in the run-time memory--the contents of the log are not persisted after reboots.

The Configuration Logger Persistency feature provides a mechanism to persist the configuration commands entered by users across reloads. Only the commands entered at the command-line interface (CLI) (that is, the commands entered in configuration mode) are persisted across reload. This feature uses the Cisco IOS secure file system to persist the configuration commands that are generated.

Note The Cisco IOS configuration logger is different from the system message logging (syslog) facility. Syslog is a general logging facility for tracking system messages. The configuration logger tracks information about configuration commands entered at the CLI.

Persisted Commands

The persisted commands from the Cisco IOS configuration logger are used as an extension to the startup configuration. These saved commands provide a quick-save capability. Rather than saving the entire startup-config file, Cisco IOS software saves just the commands entered since the last startup-config file was generated.

Only the logged commands are persisted. The following additional data from the configuration logger are *not* persisted:

- User who logged the command
- IP address from which the user logged in
- Session and log indexes for the logged command
- Time when the command was entered
- Pre- and post-NVGEN output associated with the entered command
- Parser return code output for the entered command

The persisted commands’ primary purpose is for use as a quick-save extension to the startup-config file. The additional information associated with a configuration command is not useful for quick-save purposes. If you need the additional information to be persisted across reboots (for auditing purposes), complete the following steps:

1. Enable configuration logger notification to syslog
2. Enable the syslog persistence feature

Alternatively, Cisco Networking Services, CiscoView, or other Network Management systems that manage Cisco IOS devices to keep track of configuration changes in an off-the-box storage solution can be used.

By default, upon reload, the persisted commands are appended to the startup-config file. These commands are applied only when you explicitly configure this behavior using a CLI configuration command.

How to Configure the Configuration Logger Persistency Feature

Enabling the Configuration Logger Persistency Feature

The Configuration Logger Persistency feature implements a quick-save mechanism so that the time to save changes from the startup configuration is proportional to the size of the incremental changes (with respect to the startup configuration) that need to be saved. The persisted commands from the Cisco IOS configuration logger will be used as an extension to the startup configuration. The saved commands, which are used as an extension to the startup configuration, provide a quick-save ability. Rather than saving the entire startup-config file, Cisco IOS software saves just the commands entered since the last startup-config file was generated.

To enable the Configuration Logger Persistency feature, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging persistent auto manual**
6. **logging persistent reload**
7. **logging persistent** *size threshold*
8. **logging size** *entries*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Router(config)# archive`	Enters archive configuration mode.
Step 4	**log config** **Example:** `Router(config-archive)# log config`	Enters archive configuration-log configuration mode.

	Command or Action	Purpose
Step 5	**logging persistent auto manual** **Example:** `Router(config-archive-log-cfg)# logging persistent auto`	Enables the Configuration Logger Persistency feature: • The **auto** keyword specifies that each configuration command will be saved automatically to the Cisco IOS secure file system. • The **manual** keyword specifies that you can save the configuration commands to the Cisco IOS secure file system on-demand. To do this, you must use the **archivelogconfigpersistentsave** command. **Note** To enable the **loggingpersistentauto** command, you must have disk0: configured and an external flash card inserted on the router.
Step 6	**logging persistent reload** **Example:** `Router(config-archive-log-cfg)# logging persistent reload`	Sequentially applies the configuration commands saved in the configuration logger database (since the last **writememory**command) to the running-config file after a reload.
Step 7	**logging persistent** *size threshold* **Example:** `Router(config-archive-log-cfg)# logging persistent size threshold`	Specifies the disk space size for writing log messages in the configuration logger database; triggers an alert on the console or syslog server when the log size exceeds the threshold (specified in percentage).
Step 8	**logging size** *entries* **Example:** `Router(config-archive-log-cfg)# logging size 10`	Specifies the maximum number of entries retained in the configuration log. • Valid values range from 1 to 1000. • The default value is 100 entries.

Verifying and Troubleshooting the Configuration Logger Persistency Feature

Three commands can be used to verify, archive, and clear the contents of the configuration log. For troubleshooting purposes, the command in Step 4 turns on debugging.

SUMMARY STEPS

1. **show archive log config persistent**
2. **clear archive log config persistent**
3. **archive log config persistent save**
4. **debug archive log config persistent**

DETAILED STEPS

Step 1 **show archive log config persistent**

This command displays the persisted commands in the configuration log. The commands appear in a configlet format. The following is sample output from this command:

Example:

```
Router# show archive log config persistent
!Configuration logger persistentarchive
 log config
 logging persistent auto
 logging persistent reload
archive
 log config
logging size 10
logging console
interface loop 101
 ip address 10.1.1.1 255.255.255.0
 ip address 10.2.2.2 255.255.255.0
 no shutdown
```

Step 2 **clear archive log config persistent**

This command clears the configuration logging persistent database entries. Only the entries in the configuration logging database file are deleted. The file itself is not deleted because it will be used to log new entries. After this command is entered, a message is returned to indicate that the archive log is cleared.

Example:

```
Router# clear archive log config persistent
Purged the config log persist database entries successfully
Router#
```

Step 3 **archive log config persistent save**

This command saves the configuration log to the Cisco IOS secure file system. For this command to work, the **archivelogconfigpersistentsave**command must be configured.

Step 4 **debug archive log config persistent**

This command turns on the debugging function. A message is returned to indicate that debugging is turned on.

Example:

```
Router# debug archive log config persistent
debug archive log config persistent debugging is on
```

Configuration Examples for the Configuration Logger Persistency Feature

Configuration Logger Persistency Configuration on a Cisco 7200 Series Router Example

In this example, each configuration command is saved automatically to the Cisco IOS secure file system, configuration commands saved in the configuration logger database (since the last **writememory**command) are applied sequentially to the running-config file, and the maximum number of entries retained in the configuration log is set to 10:

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# archive
Router(config-archive)# log config
Router(config-archive-log-config)# logging persistent auto
configuration log persistency feature enabled. Building configuration... [OK]
Router(config-archive-log-config)# logging persistent reload
Router(config-archive-log-config)# logging persistent size 16384 threshold 10
Router(config-archive-log-config)# logging size 10
Router(config-archive-log-config)# archive log config persistent save
Router(config-archive-log-config)# end
Router#
```

Additional References

The following sections provide references related to the Configuration Logger Persistency feature.

Related Documents

Related Topic	Document Title
Comprehensive command-reference information	*Cisco IOS Configuration Fundamentals Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Configuration Logger Persistency

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 30: Feature Information for Configuration Logger Persistency

Feature Name	Releases	Feature Information
Configuration Logger Persistency	12.2(33)SRA 12.4(11)T 12.2(33)SXH 12.2(33)SB Cisco IOS XE Release 3.9S	The Configuration Logger Persistency feature increases the operational robustness of Cisco configuration and provisioning actions by implementing a “quick-save” functionality. Effective with Cisco IOS Release 12.2(33)SRA, Release 12.4(11)T, Release 12.2(33)SXH, and Release 12.2(33)SB, Cisco software saves just the commands entered since the last startup-config file was generated, rather than saving the entire startup configuration. This feature was integrated into Cisco IOS XE Release 3.9S.

Glossary

API --application programming interface.

CAF --command action function.

CDP --Cisco Discovery Protocol.

CSB --Command Status Block.

HA --high-availability architecture.

MIB --Management Information Base.

NAF --NVGEN action function.

NVGEN --nonvolatile generation.

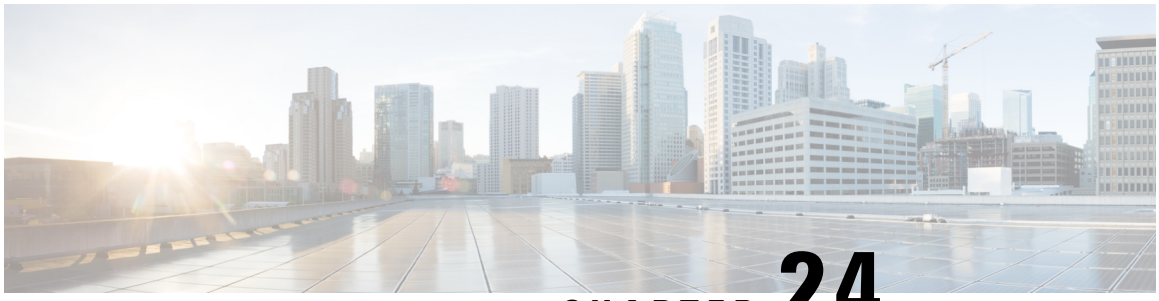
NVRAM --nonvolatile Random Access Memory.

parse chain --A sequence of C language macros defining the syntax of a Cisco IOS command.

RP --Route Processor.

SNMP --Simple Network Management Protocol.

XML --eXtensible Markup Language.



CHAPTER 24

Software Maintenance Upgrade

The Software Maintenance Upgrade (SMU) is a package that can be installed on a system to provide a patch fix or security resolution to a released image.

Information About Software Maintenance Upgrade

Software Maintenance Upgrade

The Software Maintenance Upgrade (SMU) is a package that can be installed on a system to provide a patch fix or security resolution to a released image.

An SMU package is provided on a per release and per component basis and is specific to the platform.

An SMU provides a significant benefit over classic IOS software as it allows you to address the network issue quickly while reducing the time and scope of the testing required. The Cisco IOS XE platform internally validates the SMU compatibility and does not allow you to install non-compatible SMUs.

All SMUs are integrated into the subsequent Cisco IOS XE software maintenance releases. An SMU is an independent and self-sufficient package and it does not have any prerequisites or dependencies. You can choose which SMUs to install or uninstall in any order.

Starting from Cisco IOS XE Everest 16.6.1, SMUs are supported only on Extended Maintenance releases and for the full lifecycle of the underlying software release.

The following are three basic steps to install an SMU:

- Addition of the SMU to the filesystem
- Activation of the SMU onto the system
- Committing the SMU change so it is persistent across reloads

Supported Platforms

From Cisco IOS XE Everest 16.6.1, the following platforms are supported for software maintenance upgrade:

- Cisco ASR 1000 Series Aggregate Services Routers (ASR1001-X, ASR1002-X, ASR1001-HX, ASR1002-HX, ASR1000-RP2, ASR1000-RP3)
- Cisco ISR 4000 Series Integrated Services Routers (ISR4351, ISR4331, ISR4431, ISR4321, ISR4451)
- Cisco CSR 1000v Series Cloud Services Routers
- Cisco Catalyst 9300 Series Switches
- Cisco Catalyst 9400 Series Switches

From Cisco IOS XE Dublin 17.10.1a, the following platforms are supported for software maintenance upgrade:

- Cisco Catalyst 8200 Series Edge Platforms
- Cisco Catalyst 8300 Series Edge Platforms

Software Maintenance Upgrade Package

An SMU package contains metadata and the fix for the reported issue that the SMU is requested for.

Software Maintenance Upgrade Workflow

The SMU process is initiated with a request to the SMU committee. Contact your customer support to raise an SMU request.

At release time, SMU packages are posted to the Cisco Software Download page along with the following information:

- Defect addressed
- Type of defect -PSIRT and so on

SMU Reload

The SMU type describes the effect to the system after installing the SMU. SMUs can be non-traffic affecting or can result in device reload or switchover.

A cold reload of the system requires the complete reload of the operating system. This action effects traffic flow for the duration of the reload (~5 min currently). Reload ensures that all processes are started with the correct libraries and files that are installed as part of the SMU.

How to Configure Software Maintenance Upgrade

Adding, Activating, Committing an SMU

To install an SMU package, copy the downloaded SMU package to the relevant directory on your device. Use the following commands to add, activate, and commit an SMU:

- **install add**: Runs base compatibility checks on a file to ensure that the SMU package is supported on the platform. It also adds an entry in the package/SMU .sta file, so that its status can be monitored/maintained from here on. The install add command takes the following input: package file location and downloading method (tftp, ftp, and so on)
- **install active**: Runs compatibility checks, installs the package, updates package status details. For restartable package it either triggers the appropriate post-install scripts to restart necessary processes or for non-restartable packages it will trigger a reload.
- **install commit**: Commits the activation changes to be persistent across all reloads. The commit can be done after activation while the system is up, or after the first reload. If a package was activated but not committed, it will remain active after the first reload, but not after the second reload.

Perform the following configuration to add, activate, and commit an SMU:

```
enable
install add file bootflash:isr4300-universalk9.BLD_
SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

show install summary  // Shows the installed SMU package as inactive package in the command
 output

install activate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

show version // Shows the image version tagged with the  "SMU Patched" phrase

show install summary // Shows the installed SMU package as an active package in the command
 output

install commit

show install summary // Shows the installed SMU package as a committed package in the command
 output.
```

Rolling Back, Deactivating, or Removing an SMU

Use the following commands to roll back, deactivate, and remove an SMU:

- **install rollback**: Returns the device to the previous installation state. This rollback requires a reload.
- **install deactivate**: Deactivates an active package, updates the package status, and triggers a process restart or a reload.
- **install remove**: Removes all or specified inactive SMU packages from the file system.

Perform the following tasks to roll back, deactivate, or remove an SMU:

```
enable
install rollback to committed

install deactivate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin


install remove file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin
```

Configuration Examples for Software Maintenance Upgrade

Example: Adding, Activating, and Committing an SMU

Adding, Activating, and Committing an SMU

The following example shows the workflow for adding, activating, and committing an SMU

```
Device# install add file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin
install_add: START Tue Aug  1 04:22:48 UTC 2017
install_add: Adding SMU


*Aug  1 04:22:54.492: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:22:54 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install add
bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin--- Starting SMU Add operation
 ---
Performing SMU_ADD on Active/Standby
  [R0] SMU_ADD package(s) on R0
  [R0] Finished SMU_ADD on R0
Checking status of SMU_ADD on [R0]
SMU_ADD: Passed on [R0]
Finished SMU Add operation

SUCCESS: install_add  Tue Aug  1 04:23:10 UTC 2017


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install activate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

install_activate: START Tue Aug  1 04:24:42 UTC 2017
install_activate: Activating SMU

*Aug  1 04:24:48.682: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:24:48 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install activate
```

```
bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Activate operation ---
Performing SMU_ACTIVATE on Active/Standby
  [R0] SMU_ACTIVATE package(s) on R0
    DMP package.
  [R0] Finished SMU_ACTIVATE on R0
Checking status of SMU_ACTIVATE on [R0]
SMU_ACTIVATE: Passed on [R0]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!
Aug  1 04:25:36
Aug  1 04:25:45.742 RP0/0: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install activate
SMU bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin

<after reload>

Device# show version
Cisco IOS XE Software, Version 16.06.01 - SMU-PATCHED
Cisco IOS Software [Everest], ISR Software (X86_64_LINUX_IOSD-UNIVERSALK9-M), Version 16.6.1,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Sat 22-Jul-17 05:55 by mcpre

...

Active SMU Information:
  State (St):  C - Committed, U - Uncommitted
--------------------------------------------------------------------------------
Type  Defect_ID    Version   St Filename
--------------------------------------------------------------------------------
SMU   CSCxxXXXXX     16.6.1.0. U  isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu
--------------------------------------------------------------------------------
cisco ISR4351/K9 (2RU) processor with 7941107K/6147K bytes of memory.
Processor board ID FLM2007W0MJ
3 Gigabit Ethernet interfaces
32768K bytes of non-volatile configuration memory.
16777216K bytes of physical memory.
14659583K bytes of flash memory at bootflash:.
0K bytes of WebUI ODM Files at webui:.

Configuration register is 0x0

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# show install active
 R0 ] Active Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
```

```
SMU   U    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install commit
install_commit: START Tue Aug  1 04:48:03 UTC 2017
install_commit: Committing SMU


*Aug  1 04:48:10.042: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:48:10 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install commit--- Starting SMU Commit operation ---
Performing SMU_COMMIT on Active/Standby
  [R0] SMU_COMMIT package(s) on R0
  [R0] Finished SMU_COMMIT on R0
Checking status of SMU_COMMIT on [R0]
SMU_COMMIT: Passed on [R0]
Finished SMU Commit operation
SUCCESS: install_commit  Tue Aug  1 04:48:33 UTC 2017

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0
```

Example: Rolling Back, Deactivating, or Removing an SMU

Example: Rolling back, Deactivating, or Removing an SMU

```
Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#show install rollback
ID      Label          Description
-----------------------------------------------
4       No Label        No Description

Device# install rollback to committed

install_rollback: START Tue Aug  1 05:00:37 UTC 2017

*Aug  1 05:00:44.038: %IOSXE-5-PLATFORM:  SIP2: Aug  1 05:00:44 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install rollbackinstall_rollback: Rolling back SMU

This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Rollback operation ---
Performing SMU_ROLLBACK on Active/Standby
  [R0] SMU_ROLLBACK package(s) on R0
```

```
  [R0] Finished SMU_ROLLBACK on R0
Checking status of SMU_ROLLBACK on [R0]
SMU_ROLLBACK: Passed on [R0]
Finished SMU Rollback operation

install_rollback will reload the system now!
Aug  1 05:01:40.43
Aug  1 05:01:53.558 RP0/0: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install rollback
SMU

<after reload>


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    16.6.1.0


//install deactivate: Deactivates an active package and triggers a process restart or a
reload.

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install deactivate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin
install_deactivate: START Tue Aug  1 05:28:47 UTC 2017
install_deactivate: Deactivating SMU

This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Deactivate operation ---
Performing SMU_DEACTIVATE on Active/Standby
  [R0] SMU_DEACTIVATE package(s) on R0
    DMP package.
  [R0] Finished SMU_DEACTIVATE on R0
Checking status of SMU_DEACTIVATE on [R0]
SMU_DEACTIVATE: Passed on [R0]
Finished SMU Deactivate operation

install_deactivate: Reloading the box to complete activation of the SMU...
install_deactivate will reload the system now!

<after reload>


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
```

```
--------------------------------------------------------------------------------
SMU   D    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#install commit
install_commit: START Tue Aug  1 05:39:29 UTC 2017
install_commit: Committing SMU


*Aug  1 05:39:35.222: %IOSXE-5-PLATFORM:  SIP2: Aug  1 05:39:35 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install commit--- Starting SMU Commit operation ---
Performing SMU_COMMIT on Active/Standby
  [R0] SMU_COMMIT package(s) on R0
  [R0] Finished SMU_COMMIT on R0
Checking status of SMU_COMMIT on [R0]
SMU_COMMIT: Passed on [R0]
Finished SMU Commit operation

SUCCESS: install_commit  Tue Aug  1 05:39:58 UTC 2017
Completed install commit SMU

Device#show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0


//install remove: Deletes the inactive SMU file from the file system.

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#install remove file bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
install_remove: START Tue Aug  1 05:43:22 UTC 2017
install_remove: Removing SMU

--- Starting SMU Remove operation ---
Performing SMU_REMOVE on Active/Standby
  [R0] SMU_REMOVE package(s) on R0
  [R0] Finished SMU_REMOVE on R0
Checking status of SMU_REMOVE on [R0]
SMU_REMOVE: Passed on [R0]
Finished SMU Remove operation

SUCCESS: install_remove  Tue Aug  1 05:43:43 UTC 2017

//Remove inactive: Deletes all inactive packages from the file system


Device#show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
```

```
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0


Device#install remove inactive
install_remove: START Tue Aug  1 05:52:31 UTC 2017
Cleaning up unnecessary package files
  Scanning boot directory for packages ... done.
  Preparing packages list to delete ...
    isr4300-universalk9.16.06.01.SPA.bin
      File is in use, will not delete.
    packages.conf
      File is in use, will not delete.
  done.

The following files will be deleted:
[R0]:
/bootflash/isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin

Do you want to remove the above files? [y/n]y
[R0]:
Deleting file bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin ... done.
SUCCESS: Files deleted.
--- Starting Post_Remove_Cleanup ---
Performing Post_Remove_Cleanup on Active/Standby
  [R0] Post_Remove_Cleanup package(s) on R0
  [R0] Finished Post_Remove_Cleanup on R0
Checking status of Post_Remove_Cleanup on [R0]
Post_Remove_Cleanup: Passed on [R0]
Finished Post_Remove_Cleanup

SUCCESS: install_remove  Tue Aug  1 05:53:19 UTC 2017


///Show install package

Device#show install package bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
Name: isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
Version: 16.6.1.0.202.1500742946..Everest
Platform: ISR4300
Package Type: SMU
Defect ID: CSCxxXXXXX
Package State: Not Installed
Supersedes List: {}
SMU ID: 0
SMU Type: reload
SMU Compatible with Version: 16.6.1.0.202


//Show install log
Device#show install log
[0|install_op_boot]: START Tue Aug  1 05:34:59 Universal 2017
[0|install_op_boot(INFO, )]: SMU
/bootflash/isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin will be activated upon reload.
[0|install_op_boot]: END SUCCESS  Tue Aug  1 05:35:06 Universal 2017
```

Feature Information for Software Maintenance Upgrade

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

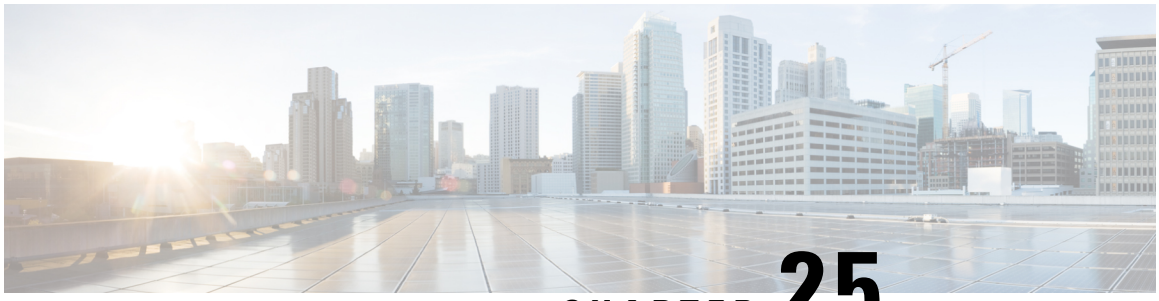
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 31: Feature Information for Software Maintenance Upgrade

Feature Name	Releases	Feature Information
Software Maintenance Upgrade	Cisco IOS XE Everest 16.6.1.	The Software Maintenance Upgrade (SMU) is a package that can be installed on a system in order to provide a patch fix or security resolution to a released image The following commands were introduced or modified: **install**, **show install**

PART IV

The Integrated File System Cisco IOS



CHAPTER 25

Configuring Basic File Transfer Services

Using basic file transfer services, you can configure a router as a Trivial File Transfer Protocol (TFTP) or Reverse Address Resolution Protocol (RARP) server, configure the router to forward extended BOOTP requests over asynchronous interfaces, and configure rcp, rsh, and FTP.

Prerequisites for Basic File Transfer Services

- You should have at least a basic familiarity with the Cisco IOS environment and the command-line interface.
- You should have at least a minimal configuration running on your system.

Restrictions for Basic File Transfer Services

- You must have your network up and running, with Cisco IOS Release 12.2 or a later release installed.
- Some of the Cisco IOS configuration commands are only available on certain router platforms, and the command syntax may vary on different platforms.

Information About Basic File Transfer Services

Use of a Router as a TFTP or RARP Server

It is too costly and inefficient to have a machine that acts only as server on every network segment. However, when you do not have a server on every segment, your network operations can incur substantial time delays across network segments. You can configure a router to serve as a RARP or TFTP server to reduce costs and time delays in your network while allowing you to use your router for its regular functions.

Typically, a router that is configured as a TFTP or RARP server provides other routers with system image or router configuration files from its Flash memory. You can also configure the router to respond to other types of service requests, such as requests.

Use of a Router as a TFTP Server

As a TFTP server host, the router responds to TFTP Read Request messages by sending a copy of the system image contained in ROM or one of the system images contained in Flash memory to the requesting host. The TFTP Read Request message must use one of the filenames that are specified in the configuration.

Note For the Cisco 7000 family, the filename used must represent a software image that is present in Flash memory. If no image resides in Flash memory, the client router will boot the server's ROM image as a default.

Flash memory can be used as a TFTP file server for other routers on the network. This feature allows you to boot a remote router with an image that resides in the Flash server memory.

Some Cisco devices allow you to specify one of the different Flash memory locations (**bootflash:**, **slot0:**, **slot1:**, **slavebootflash:**, **slaveslot0:**, or **slaveslot1:**) as the TFTP server.

Use of a Router as a RARP Server

Reverse Address Resolution Protocol (RARP) is a protocol in the TCP/IP stack that provides a method for finding IP addresses based on MAC (physical) addresses. This functionality is the reverse of broadcasting Address Resolution Protocols (ARPs), through which a host can dynamically discover the MAC-layer address corresponding to a particular IP network-layer address. RARP makes diskless booting of various systems possible (for example, diskless workstations that do not know their IP addresses when they boot, such as Sun workstations or PCs on networks where the client and server are on separate subnets). RARP relies on the presence of a RARP server with cached table entries of MAC-layer-to-IP address mappings.

You can configure a Cisco router as a RARP server. This feature enables the Cisco IOS software to answer RARP requests.

Use of a Router for rsh and rcp

Remote shell (rsh) gives users the ability to execute commands remotely. Remote copy (rcp) allows users to copy files to and from a file system residing on a remote host or server on the network. Cisco's implementation of rsh and rcp interoperates with the industry standard implementations. Cisco uses the abbreviation RCMD (Remote Command) to indicate both rsh and rcp.

Source Interface for Outgoing RCMD Communications

You can specify the source interface for RCMD (rsh and rcp) communications. For example, the router can be configured so that RCMD connections use the loopback interface as the source address of all packets leaving the router. Specifying the source-interface is most commonly used to specify a loopback interface. This allows you to associate a permanent IP address with RCMD communications. Having a permanent IP address is useful for session identification (remote device can consistently idendify the origin of packets for the session). A "well-known" IP address can also be used for security purposes, as you can then create access lists on remote devices which include the address.

About DNS Reverse Lookup for rcmd

As a basic security check, the Cisco IOS software does a reverse lookup of the client IP address using DNS for the remote command (rcmd) applications (rsh and rcp). This check is performed using a host authentication process.

When enabled, the system records the address of the requesting client. That address is mapped to a host name using DNS. Then a DNS request is made for the IP address for that host name. The IP address received is then checked against the original requesting address. If the address does not match with any of the addresses received from DNS, the rcmd request will not be serviced.

This reverse lookup is intended to help protect against "spoofing." However, please note that the process only confirms that the IP address is a valid routable address; it is still possible for a hacker to spoof the valid IP address of a known host.

Implementation of rsh

You can use rsh (remote shell) to execute commands on remote systems to which you have access. When you issue the **rsh** command, a shell is started on the remote system. The shell allows you to execute commands on the remote system without having to log in to the target host.

You do not need to connect to the system, router, or access server and then disconnect after you execute a command if you use rsh. For example, you can use rsh to remotely look at the status of other devices *without* connecting to the target device, executing the command, and then disconnecting. This capability is useful for looking at statistics on many different routers. Configuration commands for enabling rsh use the acronym "rcmd", which is short for "remote command".

Maintaining rsh Security

To gain access to a remote system running rsh, such as a UNIX host, an entry must exist in the system's *.rhosts* file or its equivalent identifying you as a user who is authorized to execute commands remotely on the system. On UNIX systems, the *.rhosts* file identifies users who can remotely execute commands on the system.

You can enable rsh support on a router to allow users on remote systems to execute commands. However, our implementation of rsh does not support an *.rhosts* file. Instead, you must configure a local authentication database to control access to the router by users attempting to execute commands remotely using rsh. A local authentication database is similar to a UNIX *.rhosts* file. Each entry that you configure in the authentication database identifies the local user, the remote host, and the remote user.

Implementation of rcp

The remote copy (rcp) commands rely on the rsh server (or daemon) on the remote system. To copy files using rcp, you do not need to create a server for file distribution, as you do with TFTP. You need only to have access to a server that supports the remote shell (rsh). (Most UNIX systems support rsh.) Because you are copying a file from one place to another, you must have read permission on the source file and write permission in the destination directory. If the destination file does not exist, rcp creates it for you.

Although Cisco's rcp implementation emulates the functions of the UNIX rcp implementation--copying files among systems on the network--Cisco's command syntax differs from the UNIX rcp command syntax. The Cisco IOS software offers a set of copy commands that use rcp as the transport mechanism. These rcp copy commands are similar in style to the Cisco IOS TFTP copy commands, but they offer an alternative that provides faster performance and reliable delivery of data. These improvements are possible because the rcp transport mechanism is built on and uses the Transmission Control Protocol/Internet Protocol (TCP/IP) stack, which is connection-oriented. You can use rcp commands to copy system images and configuration files from the router to a network server and vice versa.

You can also enable rcp support to allow users on remote systems to copy files to and from the router.

If you do not specify the**/user** keyword and argument, the Cisco IOS software sends a default remote username. As the default value of the remote username, the software sends the remote username associated with the current tty process, if that name is valid. If the tty remote username is invalid, the software uses the router host name as the both the remote and local usernames.

Configure the Remote Client to Send rcp Requests

The rcp protocol requires a client to send a remote username on each rcp request to a server. When you copy a configuration file from a server to the router using rcp, the Cisco IOS software sends the first valid username in the following list:

1. The username set by the **iprcmdremote-username** command, if the command is configured.
2. The remote username associated with the current tty (terminal) process. For example, if the user is connected to the router through Telnet and was authenticated through the **username** command, the router software sends the Telnet username as the remote username.

Note In Cisco products, ttys are commonly used in access servers. The concept of tty originated with UNIX. For UNIX systems, each physical device is represented in the file system. Terminals are called *tty devices* , which stands for *teletype* , the original UNIX terminal.

1. The router host name.

For **boot**commands using rcp, the software sends the router host name; you cannot explicitly configure the remote username.

For the rcp copy request to execute successfully, an account must be defined on the network server for the remote username.

If you are writing to the server, the rcp server must be properly configured to accept the rcp write request from the user on the router. For UNIX systems, you must add an entry to the *.rhosts* file for the remote user on the rcp server. For example, if the router contains the following configuration lines.

```
hostname Rtr1
ip rcmd remote-username User0
```

and the router's IP address translates to Router1.company.com, then the *.rhosts* file for User0 on the rcp server should contain the following line:

```
Router1.company.com Rtr1
```

Refer to the documentation for your rcp server for more details.

If the server has a directory structure, the configuration file or image is written or copied relative to the directory associated with the remote username on the server. Use the **iprcmdremote-username** command to specify which directory on the server to use. For example, if the system image resides in the home directory of a user on the server, you can specify that user's name as the remote username.

If you copy the configuration file to a personalcomputer used as a file server, the computer must support rsh.

Use of a Router for FTP Connections

You can configure a router to transfer files between systems on the network using the File Transfer Protocol (FTP). With the Cisco IOS implementation of FTP, you can set the following FTP characteristics:

- Passive-mode FTP
- User name
- Password
- IP address

How to Configure Basic File Transfer Services

Configuring the Router for Use as a TFTP Server

To configure your router for use as a TFTP server, complete the tasks in this section.

Before you begin

The server and client router must be able to reach each other before the TFTP function can be implemented. Verify this connection by testing the connection between the server and client router (in either direction) using the **ping***a.b.c.d* command (where *a.b.c.d* is the address of the client device). After the **ping** command is issued, connectivity is indicated by a series of exclamation points (!), while a series of periods (.) plus [timed out] or [failed] indicates that the connection attempt failed. If the connection fails, reconfigure the interface, check the physical connection between the Flash server and client router, andping again.

After you verify the connection, ensure that a TFTP-bootable image is present on the server. This is the system software image the client router will boot. Note the name of this software image so you can verify it after the first client boot.

Caution For full functionality, the software image sent to the client must be the same type as the ROM software installed on the client router. For example, if the server has X.25 software, and the client does not have X.25 software in ROM, the client will not have X.25 capabilities after booting from the server's image in Flash memory.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **tftp-server flash** [*partition-number***:**]*filename1* [**alias***filename2*] [*access-list-number*]
   - **tftp-server flash** *device* **:** *filename* (Cisco 7000 family only)
   - **tftp-server flash** [*device***:**][*partition-number***:**]*filename* (Cisco 1600 series and Cisco 3600 series only)
   - **tftp-server rom alias** *filename1* [*access-list-number*]
4. **end**

5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	Do one of the following: • **tftp-server flash** [*partition-number***:**]*filename1* [**alias***filename2*] [*access-list-number*] • **tftp-server flash** *device* **:** *filename* (Cisco 7000 family only) • **tftp-server flash** [*device***:**][*partition-number***:**]*filename* (Cisco 1600 series and Cisco 3600 series only) • **tftp-server rom alias** *filename1* [*access-list-number*] **Example:** `Device(config)# tftp-server flash version-10.3 22`	Specifies the system image to send in response to Read Requests. You can enter multiple lines to specify multiple images.
Step 4	**end** **Example:** `Device(config)# end`	Ends the configuration session and returns you to privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the running configuration to the startup configuration file.

Examples

In the following example, the system can use TFTP to send copies of the Flash memory file *version-10.3* in response to a TFTP Read Request for that file. The requesting host is checked against access list 22.

```
tftp-server flash version-10.3 22
```

In the following example, the system can use TFTP to send a copy of the ROM image *gs3-k.101*in response to a TFTP Read Request for the *gs3-k.101* file:

```
tftp-server rom alias gs3-k.101
```

In the following example, a router sends a copy of the file *gs7-k.9.17* in Flash memory in response to a TFTP Read Request. The client router must reside on a network specified by access list 1. Thus, in the example, the any clients on network 172.16.101.0 are permitted access to the file.

```
Server# configure terminal

Enter configuration commands, one per line. End with CTRL/Z
Server(config)# tftp-server flash gs7-k.9.17 1

Server(config)# access-list 1 permit 172.16.101.0 0.0.0.255

Server(config)# end

Server# copy running-config startup-config

[ok]
Server#
```

Troubleshooting

The TFTP session can sometimes fail. TFTP generates the following special characters to help you determine why a TFTP session fails:

- An “E” character indicates that the TFTP server received an erroneous packet.
- An “O” character indicates that the TFTP server received an out-of-sequence packet.
- A period (.) indicates a timeout.

For diagnosing any undue delay in the transfer, the output is useful. For troubleshooting procedures, refer to the *Internetwork Troubleshooting Guide* publication.

Configuring the Client Router

To configure the client router to first load a system image from the server, and as a backup, to configure the client router to load its own ROM image if the load from a server fails, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no boot system**
4. **boot system** [**tftp**] *filename* [*ip-address*]
5. **boot system rom**
6. **config-register** *value*
7. **end**
8. **copy running-config startup-config**
9. **reload**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no boot system** **Example:** `Device(config)# no boot system`	(Optional) Removes all previous **bootsystem** statements from the configuration file.
Step 4	**boot system** [**tftp**] *filename* [*ip-address*] **Example:** `Device(config)# boot system c5300-js-mz.121-5.T.bin 172.16.1.1`	Specifies that the client router load a system image from the server.
Step 5	**boot system rom** **Example:** `Device(config)# boot system rom`	Specifies that the client router loads its own ROM image if the load from a server fails.
Step 6	**config-register** *value* **Example:** `Device(config)# config-register 0x010F`	Sets the configuration register to enable the client router to load a system image from a network server.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the configuration file to your startup configuration.
Step 9	**reload** **Example:** `Device# reload`	(Optional) Reloads the router to make your changes take effect.

Examples

In the following example, the router is configured to boot from a specified TFTP server:

```
Client# configure terminal

Enter configuration commands, one per line. End with CTRL/Z
Client(config)# no boot system

Client(config)# boot system c5300-js-mz.121-5.T.bin 172.16.1.1

Client(config)# boot system rom

Client(config)# config-register 0x010F

Client(config)# end

Client# copy running-config startup-config

[ok]
Client# reload
```

In this example, the **nobootsystem** command invalidates all other**bootsystem** commands currently in the configuration memory, and any **bootsystem** commands entered after this command will be executed first. The second command, **bootsystem***filename address* , tells the client router to look for the file c5300-js-mz.121-5.T.bin on the TFTP server with an IP address of 172.16.111.111. Failing this, the client router will boot from its system ROM in response to the **bootsystemrom** command, which is included as a backup in case of a network problem. The **copyrunning-configstartup-config** command copies the configuration to the startup configuration, and the**reload**command boots the system.

Note The system software to be booted from the server must reside in Flash memory on the server. If it is not in Flash memory, the client router will boot the server's system ROM.

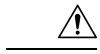
The following example shows sample output of the **showversion** command after the router has rebooted:

```
Device> show version
Cisco Internetwork Operating System Software
Cisco IOS (tm) 5300 Software (C5300-JS-M), Version 12.1(5)T,  RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2000 by Cisco Systems, Inc.
Compiled Sat 11-Nov-00 03:03 by joe
Image text-base: 0x60008958, data-base: 0x611C6000
ROM: System Bootstrap, Version 11.2(9)XA, RELEASE SOFTWARE (fc2)
BOOTFLASH: 5300 Software (C5300-BOOT-M), Version 12.0(7)T,  RELEASE SOFTWARE (f)
Router uptime is 8 weeks, 4 days, 22 hours, 36 minutes
System returned to ROM by power-on
System restarted at 00:37:38 UTC Thu Feb 22 2001
System image file is "flash:c5300-js-mz.121-5.T.bin"
.
.
.
Configuration register is 0x010F
```

The important information in this example is contained in the first line "Cisco IOS (tm).." and in the line that begins "System image file...." The "Cisco IOS (tm)..." line shows the version of the operating system in NVRAM. The "System image file...." line show the filename of the system image loaded from the TFTP server.

What to Do Next

After the system reloads, you should use the **showversion** EXEC mode command to verify that the system booted the desired image.



Caution Using the **nobootsystem** command, as in the following example, will invalidate *all* other boot system commands currently in the client router system configuration. Before proceeding, determine whether the system configuration stored in the client router should first be saved (uploaded) to a TFTP file server so you have a backup copy.

Configuring the Router as a RARP Server

To configure the router as a RARP server, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type* [*slot*/]*port*
4. **ip rarp-server** *ip-address*

DETAILED STEPS

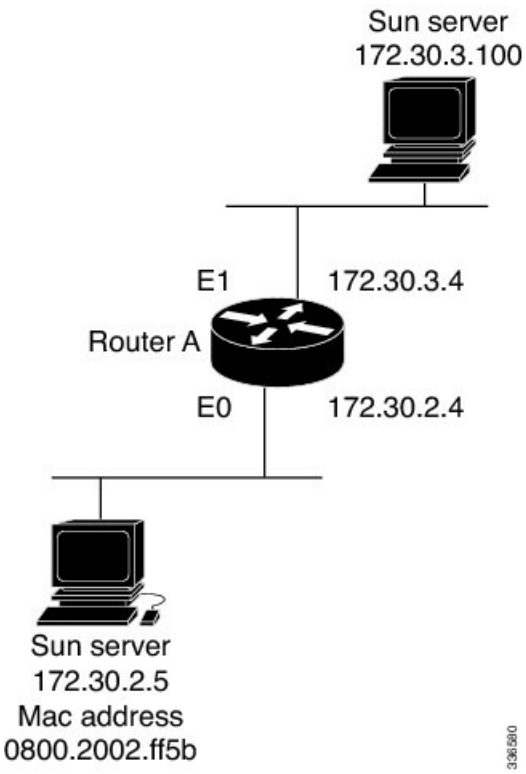
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type* [*slot*/]*port* **Example:** `Device(config)# interface Gigabitethernet 0/0`	Specifies the interface that you will be configuring the RARP service on and enters interface configuration mode for the specified interface.
Step 4	**ip rarp-server** *ip-address* **Example:**	Enables the RARP service on the router.

	Command or Action	Purpose
	`Device(config-if)# ip rarp-server 172.30.3.100`	

Examples

The figure below illustrates a network configuration in which a router is configured to act as a RARP server for a diskless workstation. In this example, the Sun workstation attempts to resolve its MAC (hardware) address to an IP address by sending a SLARP request, which is forwarded by the router to the Sun server.

Figure 9: Configuring a Router As a RARP Server



Router A has the following configuration:

```
! Allow the router to forward broadcast portmapper requests
ip forward-protocol udp 111
! Provide the router with the IP address of the diskless sun
arp 172.30.2.5 0800.2002.ff5b arpa
interface Gigabitethernet 0/0
! Configure the router to act as a RARP server, using the Sun Server's IP
! address in the RARP response packet.
ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
ip helper-address 172.30.3.100
```

The Sun client and server's IP addresses must use the same major network number because of a limitation with the current SunOS *rpc.bootparamd* daemon.

In the following example, an access server is configured to act as a RARP server.

```
! Allow the access server to forward broadcast portmapper requests
 ip forward-protocol udp 111
! Provide the access server with the IP address of the diskless sun
 arp 172.30.2.5 0800.2002.ff5b arpa
 interface Gigabitethernet 0/0
! Configure the access server to act as a RARP server, using the Sun Server's
! IP address in the RARP response packet.
 ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
 ip helper-address 172.30.3.100
```

Configuring a Router to Use rsh and rcp

Specifying the Source Interface for Outgoing RCMD Communications

To configure the router so that RCMD connections use the loopback interface as the source address of all packets leaving the router, specify the interface associated with RCMD communications by completing the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd source-interface** *interface-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd source-interface** *interface-id* **Example:** `Device(config)# ip rcmd source-interface`	Specifies the interface address that will be used to label all outgoing rsh and rcp traffic.

Disabling DNS Reverse Lookup for rcmd

DNS Reverse Lookup for rcmd is enabled by default. You can disable the DNS check for RCMD (rsh and rcp) access by completing the task in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip rcmd domain-lookup**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ip rcmd domain-lookup** **Example:** `Device(config)# no ip rcmd domain-lookup`	Disables the Domain Name Service (DNS) reverse lookup function for remote command (rcmp) applications (rsh and rcp).

Configuring the Router to Allow Remote Users to Execute Commands Using rsh

To configure the router to allow remote user to execute commands using rsh, complete the tasks in this section:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]]
4. **ip rcmd rsh-enable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]] **Example:** `Device(config)# ip rcmd remote-host Router1 172.16.101.101 netadmin4 enable`	Creates an entry in the local authentication database for each remote user who is allowed to execute rsh commands.
Step 4	**ip rcmd rsh-enable** **Example:** `Device(config)# ip rcmd rsh-enable`	Enables the software to support incoming rsh commands. **Note** To disable the software from supporting incoming rsh commands , use the **noiprcmdrsh-enable** command. **Note** When support of incoming rsh commands is disabled, you can still issue an rsh command to be executed on other routers that support the remote shell protocol and on UNIX hosts on the network.

Examples

The following example shows how to add two entries for remote users to the authentication database, and enable a router to support rsh commands from remote users:

```
ip rcmd remote-host Router1 172.16.101.101 rmtnetad1
ip rcmd remote-host Router1 172.16.101.101 netadmin4 enable
ip rcmd rsh-enable
```

The users, named *rmtnetad1* and *netadmin4* , are both on the remote host at IP address 172.16.101.101. Although both users are on the same remote host, you must include a unique entry for each user. Both users are allowed to connect to the router and remotely execute rsh commands on it after the router is enabled for rsh. The user named *netadmin4* is allowed to execute privileged EXEC mode commands on the router. Both authentication database entries give the router's host name *Router1* as the local username. The last command enables the router for to support rsh commands issued by remote users.

Executing Commands Remotely Using rsh

To execute a command remotely on a network server using rsh, use the following commands in user EXEC mode:

SUMMARY STEPS

1. **enable**

2. **rsh** {*ip-address* | *host* } [/**user***username*] *remote-command*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**rsh** {*ip-address* | *host* } [/**user***username*] *remote-command* **Example:** `Device# rsh mysys.cisco.com /user sharon ls -a`	Executes a command remotely using rsh.

Examples

The following example executes the “ls -a” command in the home directory of the user sharon on mysys.cisco.com using rsh:

```
Device# enable
Device# rsh mysys.cisco.com /user sharon ls -a
.
..
.alias
.cshrc
.emacs
.exrc
.history
.login
.mailrc
.newsrc
.oldnewsrc
.rhosts
.twmrc
.xsession
jazz
Device#
```

Configuring the Router to Accept rcp Requests from Remote Users

To configure the Cisco IOS software to support incoming rcp requests, use the following commands in global configuration mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]]
4. **ip rcmd rcp-enable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]] **Example:** `Device(config)# ip rcmd remote-host Router1 172.16.101.101 netadmin3`	Create an entry in the local authentication database for each remote user who is allowed to execute rcp commands. **Note** To disable the software from supporting incoming rcp requests, use the **noiprcmdrcp-enable** command. **Note** When support for incoming rcp requests is disabled, you can still use the rcp commands to copy images from remote servers. The support for incoming rcp requests is distinct from its ability to handle outgoing rcp requests.
Step 4	**ip rcmd rcp-enable** **Example:** `Device(config)# ip rcmd rcp-enable`	Enable the software to support incoming rcp requests.

Examples

The following example shows how to add two entries for remote users to the authentication database and then enable the software to support remote copy requests from remote users. The users, named *netadmin1* on the remote host at IP address 172.16.15.55 and *netadmin3* on the remote host at IP address 172.16.101.101, are both allowed to connect to the router and remotely execute rcp commands on it after the router is enabled to support rcp. Both authentication database entries give the host name *Router1* as the local username. The last command enables the router to support for rcp requests from remote users.

```
ip rcmd remote-host Router1 172.16.15.55 netadmin1
ip rcmd remote-host Router1 172.16.101.101 netadmin3
ip rcmd rcp-enable
```

Configuring the Remote to Send rcp Requests

To override the default remote username sent on rcp requests, use the following command in global configuration mode:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-username** *username*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username sharon`	Specifies the remote username. **Note** To remove the remote username and return to the default value, use the **noiprcmdremote-username** command.

Configuring a Router to Use FTP Connections

To configure a router to transfer files between systems on the network using the File Transfer Protocol (FTP), complete the tasks in this section to configure the FTP characteristics:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *string*
4. **ip ftp password** [*type*] *password*
5. Do one of the following:
   - **ip ftp passive**
   -
   -
   - **no ip ftp passive**
6. **ip ftp source-interface** *interface*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

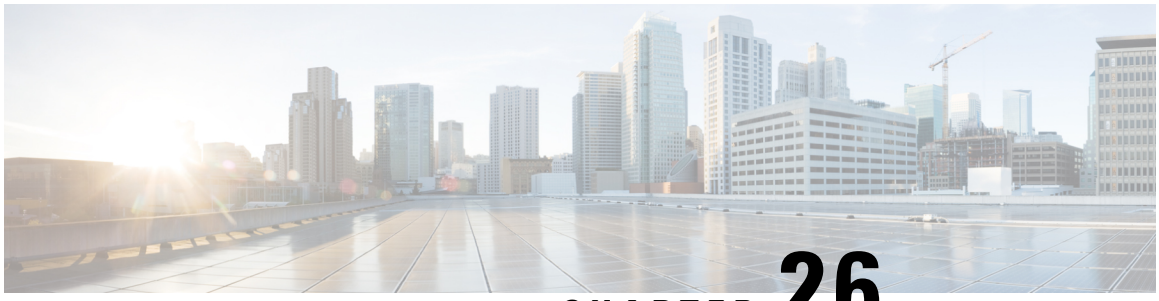
	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ftp username** *string* **Example:** `Device(config)# ip ftp username zorro`	Specifies the user name to be used for the FTP connection.
Step 4	**ip ftp password** [*type*] *password* **Example:** `Device(config)# ip ftp password sword`	Specifies the password to be used for the FTP connection.
Step 5	Do one of the following: • **ip ftp passive** • • • **no ip ftp passive** **Example:** `Device(config)# ip ftp passive`	Configures the router to only use passive-mode FTP connections. or Allows all types of FTP connections (default).
Step 6	**ip ftp source-interface** *interface* **Example:** `Device(config)# ip ftp source-interface to1`	Specifies the source IP address for FTP connections.

Examples

The following example demonstrates how to capture a core dump using the Cisco IOS FTP feature. The router accesses a server at IP address 192.168.10.3 with login name zorro and password sword. The default passive-mode FTP is used, and the server is accessed using Token Ring interface to1 on the router where the core dump will occur:

```
ip ftp username zorro
ip ftp password sword
ip ftp passive
ip ftp source-interface to1
! The following command allows the core-dump code to use FTP rather than TFTP or RCP
exception protocol ftp
! The following command identifies the FTP server
```

```
! 192.168.10.3 crashes
exception dump 192.168.10.3
```



CHAPTER 26

Transferring Files Using HTTP or HTTPS

Cisco IOS Release 12.4 provides the ability to transfer files between your Cisco IOS software-based device and a remote HTTP server using the HTTP or HTTP Secure (HTTPS) protocol. HTTP and HTTPS can now be specified as the targets and source locations in Cisco IOS command-line interface (CLI) commands that use file system prefixes such as the **copy** command.

Prerequisites for Transferring Files Using HTTP or HTTPs

To copy files to or from a remote HTTP server, your system must support the HTTP client feature, which is integrated in most Cisco IOS software images. The HTTP client is enabled by default. To determine if the HTTP client is supported on your system, issue the **show ip http client all**command. If you are able to execute the command, the HTTP client is supported.

Commands exist for the optional configuration of the embedded HTTP client and for the HTTPS client, but the default configuration is sufficient for using the File Transfer Using HTTP or HTTPS feature. For information on configuring optional HTTP or HTTPS client characteristics, see the “Related Documents” section.

Restrictions for Transferring Files Using HTTP or HTTPs

- Existing limitations to the **copy** command, such as no network-to-network copies, are in effect for the File Transfer Using HTTP or HTTPS feature.

Note The **copy** command in Cisco IOS Release 12.4T does not work in conjunction with older versions of the Apache server software. The Apache server software must be upgraded to version 2.0.49 or later in order to use the copy command.

- From Cisco Release 17.3.1, a TLS connection is only established when the hostname matches with the Subject Alternative Name (SAN) or Common Name (CN) in a certificate. If a server doesn't meet these expectations and sends invalid attributes, then the SSL handshake is denied because there a TLS connection is not established. Hence, the HTTPS copy won't be successful.
- From Cisco IOS XE 17.15.1a, you must provide the absolute file path for the copy to succeed. See section HTTP or HTTPs File Transfer Using Absolute File Path for sample configurations. Providing the relative file path will result in error "No such file or directory".

Note Only the absolute path is used in the file transfer. The file path provided in the **ip http path** <> command is not used in the file transfer.

Information About File Transfers Using HTTP or HTTPs

To transfer files using HTTP or HTTPS, you should understand the following concept:

The File Transfer Using HTTP or HTTPs feature provides the capability to copy files, such as Cisco IOS image files, core files, configuration files, log files, scripts, and so on, to and from a remote server and your local routing device using the Cisco IOS **copy** command and command-line interface. The HTTP copy operation works in the same way as copying from other remote file systems, such as FTP or TFTP.

The HTTP copy operation can use the embedded HTTPS client for HTTP Secure transfers, providing secure and authenticated file transfers within the context of a public key infrastructure (PKI).

How to Transfer Files Using HTTP or HTTPs

This section contains the following procedures:

Note To use the File Transfer Using HTTP feature, you may need to specify a username and password for the HTTP connections for those servers that require a username and password to connect. Commands are also available to specify custom connection characteristics, although default settings can be used. The feature also offers commands to monitor and maintain connections and files.

Configuring HTTP Connection Characteristics for File Transfers

Default values are provided for HTTP File transfers. The following task is used to customize the connection characteristics for your network to specify a username and password, connection preferences, a remote proxy server, and the source interface to be used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http client connection** {**forceclose** | **idletimeout***seconds* | **timeout***seconds*}

4. **ip http client username** *username*
5. **ip http client password** *password*
6. **ip http client proxy-server** {*proxy-name* | *ip-address*} [**proxy-port***port-number*]
7. **ip http client source-interface** *interface-id*
8. **do copy running-config startup-config**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip http client connection** {**forceclose** | **idletimeout***seconds* | **timeout***seconds*} **Example:** `Router(config)# ip http client connection timeout 15`	Configures characteristics for HTTP client connections to a remote HTTP server for all file transfers: • **forceclose** --Disables the default persistent connection. • **idle timeout** *seconds* --Sets the period of time allowed for an idle connection, in a range from 1 to 60 seconds. Default timeout is 30 seconds. • **timeout** *seconds* --Sets the maximum time the HTTP client waits for a connection, in a range from 1 to 60 seconds. Default is 10 seconds.
Step 4	**ip http client username** *username* **Example:** `Router(config)# ip http client username user1`	Specifies the username to be used for HTTP client connections that require user authentication. **Note** You can also specify the username on the CLI when you issue the **copy** command, in which case the username entered overrides the username entered with this command. See the "Downloading a File from a Remote Server Using HTTP or HTTPs: Example" section for an example.
Step 5	**ip http client password** *password* **Example:**	Specifies the password to be used for HTTP client connections that require user authentication.

	Command or Action	Purpose
	`Router(config)# ip http client password letmein`	**Note** You can also specify the password on the CLI when you issue the **copy** command, in which case the password entered overrides the password entered with this command. See the "Downloading a File from a Remote Server Using HTTP or HTTPs: Example" section for an example.
Step 6	**ip http client proxy-server** {*proxy-name* | *ip-address*} [**proxy-port***port-number*] **Example:** `Router(config)# ip http client proxy-server edge2 proxy-port 29`	Configures the HTTP client to connect to a remote proxy server for HTTP file system client connections. • The optional**proxy-port***port-number* keyword and argument specify the proxy port number on the remote proxy server.
Step 7	**ip http client source-interface** *interface-id* **Example:** `Router(config)# ip http client source-interface Ethernet 0/1`	Specifies the interface for the source address in all HTTP client connections.
Step 8	**do copy running-config startup-config** **Example:** `Router(config)# do copy running-config startup-config`	(Optional) Saves the running configuration as the startup configuration file. • The **do** command allows you to execute privileged EXEC mode commands from global configuration mode.
Step 9	**end** **Example:** `Router(config)# end` **Example:** `Router#`	Ends your configuration session and returns the CLI to user EXEC mode.

Downloading a File from a Remote Server Using HTTP or HTTPs

Perform this task to download a file from a remote HTTP server using HTTP or HTTPs. The **copy** command helps you to copy any file from a source to a destination.

SUMMARY STEPS

1. **enable**
2. Do one of the following:
   - **copy** [**/erase**] [**/noverify**] **http://***remote-source-urllocal-destination-url*
   - **copy https://** *remote-source-url local-destination-url*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable` **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Do one of the following: • **copy** [**/erase**] [**/noverify**] **http://***remote-source-urllocal-destination-url* • **copy https://** *remote-source-url local-destination-url* **Example:** `Router# copy http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx flash:c7200-i-mx` **Example:** `Router# copy` **Example:** `copy https://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx flash:c7200-i-mx`	Copies a file from a remote web server to a local file system using HTTP or HTTPS. • **/erase** --Erases the local destination file system before copying. This option is provided on Class B file system platforms with limited memory to allow an easy way to clear local flash memory space. • **/noverify** --If the file being copied is an image file, this keyword disables the automatic image verification that occurs after an image is copied. • The *remote-source-url* argument is the location URL (or alias) from which to get the file to be copied, in standard Cisco IOS file system HTTP syntax as follows: **http://** [[*username*:*password*]**@**] {*hostname* | *host-ip*}[*/filepath*]*/filename* **Note** The optional *username* and *password* arguments can be used to log in to an HTTP server that requires user authentication, in place of configuring the **iphttpclientusername** and **iphttpclientpassword** global configuration commands to specify these authentication strings. • The *local-destination-url* is the location URL (or alias) to put the copied file, in standard Cisco IOS file system syntax as follows: *filesystem* **:** [*/filepath*][*/filename*] **Note** For more information on URL syntax when you use the **copy** command, see the "Additional References" section.

Troubleshooting Tips

If file transfers from a remote web server fail, verify the following:

- Your router has an active connection to the Internet.
- The correct path and filename have been specified.

- The remote server requires a username and password.
- The remote server has a nonstandard communications port configured. (The default port for HTTP is 80; the default port for HTTPS is 443.)

The CLI returns error messages to help you determine the cause of a failed copy request. Additional information on the copy process can be displayed with the **debugiphttpclientall** command.

Uploading a File to a Remote Server Using HTTP or HTTPS

Perform this task to upload a file to a remote HTTP server using HTTP or HTTPS.

SUMMARY STEPS

1. **enable**
2. Do one of the following:
   - **copy** [**/erase**] [**/noverify**] *local-source-url***http://***remote-destination-url*
   - **copy** *local-source-url* **https://** *remote-destination-url*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Do one of the following: • **copy** [**/erase**] [**/noverify**] *local-source-url***http://***remote-destination-url* • **copy** *local-source-url* **https://** *remote-destination-url* **Example:** `Router# http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx_backup` **Example:** `Router# copy flash:c7200-i-mx http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx_backup` **Example:**	Copies a file from a local file system to a remote web server using HTTP or HTTPS. • **/erase** --Erases the local destination file system before copying. This option is provided on Class B file system platforms with limited memory to allow an easy way to clear local flash memory space. • **/noverify** --If the file being copied is an image file, this keyword disables the automatic image verification that occurs after an image is copied. • The *local-source-url* argument is the location URL (or alias) from which to get the file to be copied, in standard Cisco IOS file system syntax as follows: **http://** [[*username*:*password*]**@**] {*hostname* | *host-ip*}[*/filepath*]*/filename*

	Command or Action	Purpose
		Note The optional *username* and *password* arguments can be used to log in to an HTTP server that requires user authentication, in place of configuring the **iphttpclientusername** and **iphttpclientpassword** global configuration commands to specify these authentication strings. • The *remote-destination-url* is the URL (or alias) to put the copied file, in standard Cisco IOS file system syntax, as follows: *filesystem* **:** [*/filepath*][*/filename*] **Note** For more information on URL syntax when you use the **copy** command, see the "Additional References" section.

Troubleshooting Tips

If file transfers from a remote web server fail, verify the following:

- Your router has an active connection to the Internet.
- The correct path and filename have been specified.
- The remote server requires a username and password.
- The remote server has a nonstandard communications port configured. (The default port for HTTP is 80; the default port for HTTPS is 443.)

The CLI returns error messages to help you determine the cause of a failed copy request. Additional information on the copy process can be displayed with the **debugiphttpclientall** command.

Maintaining and Monitoring File Transfers Using HTTP

Perform this task to maintain and monitor HTTP connections. Steps 2 through 4 can be performed in any order.

SUMMARY STEPS

1. **enable**
2. **show ip http client connection**
3. **show ip http client history**
4. **show ip http client session-module**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**show ip http client connection** **Example:** `Router# show ip http client connection`	Displays details about active HTTP client connections.
Step 3	**show ip http client history** **Example:** `Router# show ip http client history`	Displays the last 20 URLs accessed by the HTTP client.
Step 4	**show ip http client session-module** **Example:** `Router# show ip http client session-module`	Displays details about about sessions (applications) that have registered with the HTTP client.

Configuration Examples for the File Transfer Using HTTP or HTTPs

Configuring HTTP Connection Characteristics for File Transfers Example

The following example shows how to configure the HTTP password and username for connection to a remote server that authenticates all users. The example also shows how to configure the connection for a 20-second idle connection period. The maximum time the HTTP client waits for a connection remains at the default 10 seconds.

```
Router(config)# ip http client connection idle timeout 20
Router(config)# ip http client password Secret
Router(config)# ip http client username User1
Router(config)# do show running-config | include ip http client
```

Downloading a File from a Remote Server Using HTTP or HTTPs Example

The following example shows how to configure the file c7200-i-mx is copied from a remote server to flash memory using HTTP. This example also shows how to enter a username and password from the command line for an HTTP server that authenticates users.

```
Router# copy http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx
flash:c7200-i-mx
```

Uploading a File from Flash to the Remote HTTP Server Example

The following example shows how to copy a file from flash memory to the remote HTTP server. The example shows the prompts and displays that can be expected from transferring a file using the**copy** privileged EXEC command.

```
Router# copy flash:c7200-js-mz.ELL2 http://172.19.209.190/user1/c7200-js-mz.ELL2
Address or name of remote host [172.19.209.190]?
Destination filename [user1/c7200-js-mz.ELL2]?
Storing http://172.19.209.190/user1/c7200-js-mz.ELL2 !!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
17571956 bytes copied in 57.144 secs (307503 bytes/sec)
```

Downloading a File from the Remote HTTP Server to Flash Memory Example

The following example shows how to copy a file from the remote HTTP server to the flash memory. The example shows the prompts and displays that can be expected from transferring a file using the **copy** privileged EXEC command.

```
Router# copy http://172.19.209.190/user1/c7200-i-mz.test  flash:c7200-i-mz.test
Destination filename [c7200-i-mz.test]?
Loading http://172.19.209.190/user1/c7200-i-mz.test
.
.
.
```

11272788 bytes copied in 527.104 secs (21386 bytes/sec)

Uploading a File to a Remote Server Using HTTP or HTTPs

The following example shows how to copy a file to the remote server using HTTP or HTTPs.

```
router#copy flash
: http:
Source filename []? running-config
Address or name of remote host []? 10.1.102.1 Destination filename [pilot-confg]?file1 ...
```

HTTP or HTTPs File Transfer Using Absolute File Path

From Cisco IOS XE 17.15.1a, providing the relative file path results in an error. You must provide the absolute file path for the transfer to succeed.

Example	Old Behavior (Cisco IOS XE 17.14.x and lower)	New Behavior (Cisco IOS XE 17.15.1a and higher)
Relative File Path Configuration: `Router(config)#ip http path <path e.g. flash:>` Example 1: `Router#copy http(s)://<server ip>/common.tar flash:`	System picks the absolute file path from the configuration, transfer is successful.	System does not pick the absolute file path from the configuration. `Error:(No such file or directory)`
Example 2: `wget -N http://<device ip>/common.tar -t 2 --no-check-certificate --no-proxy --http-user=<username> --http-password=<password>`	System picks the absolute file path from the configuration, transfer is successful.	System does not pick the absolute file path from the configuration. `Error:(No such file or directory)`
Absolute File Path Configuration: `Router(config)#ip http path <path e.g. flash:>` Example 1: `Router#copy http(s)://<server ip>/flash/common.tar flash:`	Transfer is successful.	Transfer is successful.
Example 2: `wget -N http://<device ip>/flash/common.tar -t 2 --no-check-certificate --no-proxy --http-user=<username> --http-password=<password>`	Transfer is successful.	Transfer is successful.

Additional References

The following sections provide information related to transferring files using HTTP or HTTPS.

Related Documents

Related Topic	Document Title
Secure HTTP communications	*HTTPS --HTTP Server and Client with SSL 3.0*
Cisco IOS embedded web server	*HTTP 1.1 Web Server and Client*
Cisco IOS embedded web client	*HTTP 1.1 Client*

Related Topic	Document Title
Network Management Commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Network Management Command Reference*
Configuration Fundamentals Commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Configuration Fundamentals Command Reference*

Standards

Standards	Title
No new of modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2616	*Hypertext Transfer Protocol -- HTTP/1.1*, R. Fielding, et al.
RFC 2617	*HTTP Authentication: Basic and Digest Access Authentication*, J. Franks, et al.

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Transferring Files Using HTTP or HTTPS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

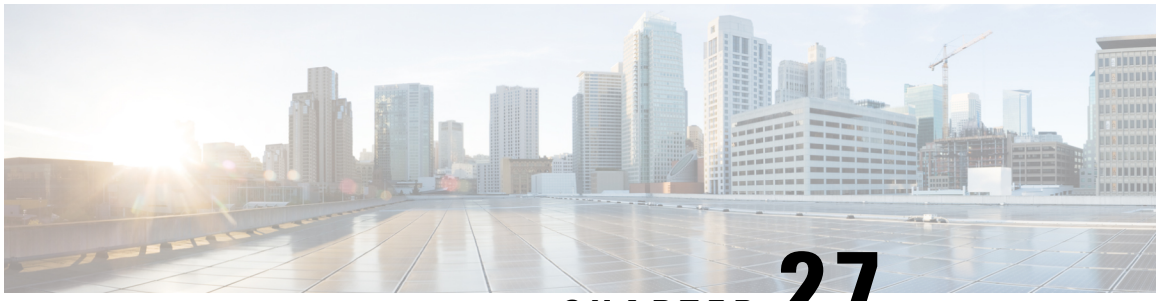
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 32: Feature Information for Transferring Files Using HTTP or HTTPS

Feature Name	Releases	Feature Information
File Download Using HTTP	12.3(2)T	The File Download Using HTTP feature allows you to copy files from an HTTP server to a Cisco IOS software-based platform.
File Upload Using HTTP	12.3(7)T	
File Transfer Using HTTP	12.3(7)T	The File Transfer Using HTTP feature provides the capability to copy files, such as Cisco IOS image files, core files, configuration files, log files, and scripts to and from a remote server and your local routing device using the Cisco IOS **copy** command and command-line interface. The HTTP copy operation works in the same way as copying from other remote file systems, such as FTP or TFTP. This feature provides support for copying files from a Cisco IOS software-based platform to an HTTP server, using either HTTP or HTTPs.

PART V

Loading and Managing System Images



CHAPTER 27

Digitally Signed Cisco Software

The Digitally Signed Cisco Software feature describes how to identify digitally signed Cisco software, gather software authentication information related to digitally signed images, and perform key revocation. Digitally Signed Cisco software is software that is digitally signed using secure asymmetrical (public-key) cryptography.

The purpose of digitally signed Cisco software is to ensure that customers are confident that the software running within their systems is secure and has not been tampered with, and that the software running in those systems originated from the trusted source as claimed.

For customers concerned about software updates involving digitally signed Cisco software--no action is necessary for customers to take advantage of the increased protection. The system operation is largely transparent to existing practices. Some minor changes in system displays reflect the use of digitally signed Cisco software.

Restrictions for Digitally Signed Cisco Software

The Cisco Catalyst 4500 E+Series switches running Cisco IOS XE software include the functionality described in this document, except for Digitally Signed Software Key Revocation and Replacement.

Information About Digitally Signed Cisco Software

Features and Benefits of Digitally Signed Cisco Software

Three main factors drive digitally signed Cisco software and software integrity verification:

- The U.S. government is introducing a new version of the Federal Information Processing Standard (FIPS) 140. FIPS-140-3 is the latest draft and is scheduled for ratification in 2010 and to be effective in 2011. This standard requires software to be digitally signed and to be verified for authenticity and integrity prior to load and execution.

- The focus on product security provides increased protection from attacks and threats to Cisco products. Digitally signed Cisco software offers increased protection from the installation and loading of software that has been corrupted or modified.
- Digitally signed Cisco software provides counterfeit protection, which provides further assurance for customers that the equipment they purchase is as claimed.

Digitally Signed Cisco Software Identification

Digitally signed Cisco IOS software is identified by a three-character extension in the image name. The Cisco software build process creates a Cisco IOS image file that contains a file extension based on the signing key that was used to sign images. These file extensions are:

- .SPA
- .SSA

The significance of each character in the file extension is explained in the table below.

Table 33: Digitally Signed Cisco Software Images File Extension Character Meanings

File Extension Character	Character Meaning
S (first character)	Stands for digitally signed software.
P or S (second character)	P and S stand for a production and special (development) image, respectively. A production image is Cisco software approved for general release; a special image is development software provided under special conditions for limited use.
A (third character)	Indicates the key version used to digitally sign the image. A key version is identified by an alphabetical character - for example, A,B,C...

Digitally Signed Cisco Software Key Types and Versions

Digitally signed Cisco software keys are identified by the type and version of the key. A key can be a special, production, or rollover key type. Special and production keys can be revoked. A rollover key is used to revoke a production or special key. The second character in the file extension indicates whether the key type is a special or production key. The key type can be “P” for a production key or an “S” for a special key.

Production and special key types have an associated key version. The key version is defined by the third character in the file extension, in the form of an alphabetical character; for example A, B or C. When a key is replaced, the key version is incremented alphabetically. For example, after a key revocation of a key type “P” (production key) with a key version of “A”, the new image will be signed with key version “B”. Key type and key version are stored as part of the key record in the key storage of the device.

Digitally Signed Cisco Software Key Revocation and Replacement

Note Key revocation and replacement is not supported on Catalyst 4500 E+Series switches running IOS XE software.

Key Revocation

Key revocation is the process of removing a key from operational use in digitally signed Cisco software.

Key revocation takes place when a key becomes compromised or is no longer used. Key revocation and replacement is only necessary in the event of a certain type of vulnerability or catastrophic loss to Cisco's secure key infrastructure. Operational steps to remedy the situation would only be necessary if notified and directed by Cisco. Notification and direction would occur through posting of advisories or field notices on www.cisco.com.

There are two different key revocation processes depending on the type of key to be revoked:

- Production key replacement uses a revocation image and a production image
- Special key replacement uses a production image

Key Replacement

Key replacement is the process of providing a new key to replace a compromised key. The new key is added before the compromised key is revoked. Key replacement is a two-step process:

1. A new key is added to the key storage to replace the revoked key.
2. After the image is verified as operating correctly with the new key, the compromised key is revoked from the key storage.

Key Revocation Image

A revocation image is a basic version of the normal image whose function is to add a new production key to the key storage area. A revocation image has no other capabilities. When a key is to be revoked and replaced, one revocation image per key is provided.

A revocation image contains a new production key bundled within it.

A rollover key stored on the platform is used to verify the signature of the revocation image--a valid revocation image is signed using the same rollover key.

Note A revocation image can be used only in production key revocation.

Important Tasks Concerning the Revocation Image

There are two important tasks concerning the revocation image:

- Adding the new production key to the key storage area.
- Performing a production key upgrade check. For more information, see Step 2 in the “Production Key Revocation”.

Adding the New Production Key to the Key Storage Area:

The revocation image adds the bundled production key to the key storage. The key is written to the primary and backup key storage areas after the revocation image checks that the key is already not part of the existing set of keys in the key storage.

Performing a Key Upgrade Check:

After the new key is added and the customer has upgraded the software (Cisco IOS and ROMmon), the show software authenticity upgrade-status command should be run. The user can review the command output to determine if the production key is successfully upgraded, and can be selected for the next boot.

Production Key Revocation

A production key (also called the release key) is revoked and replaced using a revocation image signed with a rollover key, because the images signed using the compromised production key cannot be trusted. The ROMmon can boot any image signed using a rollover key. The production key revocation and replacement process involves four steps:

1. Add the new production key to the key storage. The new production key is bundled within the revocation image.
2. Perform a software upgrade check using the show software authenticity upgrade-status command to verify the following:
   - The new production key version is installed.
   - The new production key is added to the primary key storage (if not, issue the software authenticity key add production command again with the existing revocation image).
   - The new production key is added to the backup key storage (if not, issue the software authenticity key add production command again with the existing revocation image).
   - The image is configured for autoboot (with the boot system command) signed with the new production key (if not, make sure the new production image is copied into the box and modify the boot system command to point to the new image).
   - The upgradable ROMmon is signed with the new production key (if not, upgrade the ROMmon to the one signed with the new production key).
3. Once everything is verified, the user may load the production image signed with the new production key by using the reload command.
4. Once the new production image is loaded, the user may revoke the compromised key using the software authenticity key revoke production command.

Steps 1 and 2 are done using the special revocation image. It is important for the user to do verifications in Step 2 because after a reboot (in Step 3), an old key will not be revoked if any of the software is still using the old key. The verifications help to ensure that the new key is fully installed and the next reboot (in Step 3) will use the new release software and new ROMmon. Revoking the old production key (Step 4) can be done only after the new key and the new software are installed to the system.

Special Key Revocation

A special key is revoked using a production image signed with a production key. Each production image used for special key revocation has a bundled special key that is the latest at the time of building the production image. The special key revocation and replacement process involves three steps:

1. Add the bundled new special key to the key storage area.
2. Upgrade the ROMmon that is signed using the compromise special key, to the new ROMmon signed with the new special key.
3. Revoke the compromised key from the key storage.

Note that Step 3 does not require any reboot and will be done using the production image itself. This is because the customer is already running a production image and invalidation itself happens from the running production image. Special images do not have the capability to add or invalidate any key.

How to Work with Digitally Signed Cisco Software Images

Identifying Digitally Signed Cisco Software

Perform this task to identify digitally signed Cisco software by examining the image filename in the command output from the show version command, and judging it on the criteria described in the "Digitally Signed Cisco Software Identification" section.

Note If the image file has been renamed by the user, it may not be possible to identify the image because the user may have overwritten the criteria used to indicate that the image is digitally signed.

SUMMARY STEPS

1. **enable**
2. **show version**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show version** **Example:** `Device# show version`	Displays information about the Cisco IOS software version running on a routing device, the ROM Monitor and Bootflash software versions, and the hardware configuration, including the amount of system memory.

Displaying Digitally Signed Cisco Software Signature Information

Perform this task to display information related to software authentication for the current ROMmon and the Cisco IOS image file used for booting. The display includes image credential information, the key type used for verification, signature information, and other attributes in the signature envelope.

SUMMARY STEPS

1. **enable**
2. **show software authenticity running**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show software authenticity running** **Example:** `Device# show software authenticity running`	Displays software authenticity-related information for the current ROMmon and the Cisco IOS image file used for booting.

Displaying Digital Signature Information for a Specific Image File

Perform this task to display the digital signature information related to software authentication for a specific image file.

SUMMARY STEPS

1. **enable**
2. **show software authenticity file {flash0:filename | flash1:filename | flash:filename | nvram:filename | flash0:filename | flash1:filename}**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show software authenticity file {flash0:filename | flash1:filename | flash:filename | nvram:filename | flash0:filename | flash1:filename}** **Example:** `Device# show software authenticity file flash0:c3900-universalk9-mz.SPA`	Displays digital signature and software authenticity-related information for a specific image file.

Displaying Digitally Signed Cisco Software Key Information

Perform this task to display digitally signed Cisco software key information. The information details the software public keys that are in storage with the key types.

SUMMARY STEPS

1. **enable**
2. **show software authenticity keys**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show software authenticity keys** **Example:** `Device# show software authenticity keys`	Displays the software public keys that are in storage with the key types for digitally signed Cisco software.

Troubleshooting Digitally Signed Cisco Software Images

Perform this task to troubleshoot digitally signed Cisco software images.

SUMMARY STEPS

1. **enable**
2. **debug software- authenticity errors {envelope | errors | key | revocation | show | verbose}**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug software- authenticity errors {envelope | errors | key | revocation | show | verbose}** **Example:** `Device# debug software-authenticity errors`	Enables the display of debug messages for digitally signed Cisco software.

Configuration Examples for Digitally Signed Cisco Software

Identifying Digitally Signed Cisco Software Example

The following example displays the digitally signed Cisco software image filename and allows a user to identify it based on the digitally signed Cisco software identification criteria:

```
Device# show version
Cisco IOS Software, C3900 Software (C3900-UNIVERSALK9-M),
12.4(20090904:044027) [i12 577]
Copyright (c) 1986-2009 by Cisco Systems, Inc.
Compiled Fri 04-Sep-09 09:22 by xxx
ROM: System Bootstrap, Version 12.4(20090303:092436)
C3900-2 uptime is 8 hours, 41 minutes
System returned to ROM by reload at 08:40:40 UTC Tue May 21 1901!
System image file is "xxx.SPA"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Cisco xxx (revision 1.0) with CISCxxx with 987136K/61440K bytes of memory.
Processor board ID xxx
3 Gigabit Ethernet interfaces
1 terminal line
1 Virtual Private Network (VPN) Module
1 cisco Integrated Service Engine(s)
DRAM configuration is 72 bits wide with parity enabled.
255K bytes of non-volatile configuration memory.
1020584K bytes of USB Flash usbflash0 (Read/Write)
1020584K bytes of USB Flash usbflash1 (Read/Write)
500472K bytes of ATA System CompactFlash 0 (Read/Write)
License Info:
License UDI:
-------------------------------------------------
Device#   PID                   SN
-------------------------------------------------
xx        xxx                   xxxx
Technology Package License Information for Module:'xxx'
-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      securityk9    Evaluation    securityk9
uc            None          None          None
data          None          None          None
Configuration register is 0x2102
```

Note the digitally signed image file is identified in the following line:

```
System image file is "xxx.SPA"
```

The image has a three-character extension in the filename (.SPA) characteristic of digitally signed Cisco software. Based on the guidelines in the “Digitally Signed Cisco Software Identification” section the first character in the file extension “S” indicates that the image is a digitally signed software image, the second character “P” indicates that the image is digitally signed using a production key, and the third character “A” indicates that the key version is version A.

Displaying Digitally Signed Cisco Software Signature Information Example

The following example shows how to display information related to software authentication for the current ROMmon and Cisco IOS image file used for booting:

```
Device# show software authenticity running
SYSTEM IMAGE
--------------------
Image type                    : Development
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : xxx
    Signature Algorithm       : 2048-bit RSA
    Key Version               : xxx

    Verifier Information
        Verifier Name         : ROMMON 2
        Verifier Version      :  System Bootstrap, Version 12.4(20090409:084310)
ROMMON 2
---------------
Image type                    : xxx
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : xxx
    Signature Algorithm       : 2048-bit RSA
    Key Version               : xx

    Verifier Information
        Verifier Name         : ROMMON 2
        Verifier Version      :  System Bootstrap, Version 12.4(20090409:084310) [
```

The table below describes the significant fields shown in the display.

Table 34: show software authenticity running Field Descriptions

Field	Description
SYSTEM IMAGE	Section of the output displaying the system image information.
Image type	Displays the type of image.
Common Name	Displays the name of the software manufacturer.
Organization Unit	Displays the hardware the software image is deployed on.

Field	Description
Organization Name	Displays the owner of the software image.
Certificate Serial Number	Displays the certificate serial number for the digital signature.
Hash Algorithm	Displays the type of hash algorithm used in digital signature verification.
Signature Algorithm	Displays the type of signature algorithm used in digital signature verification.
Key Version	Displays the key version used for verification.
Verifier Name	Name of the program responsible for performing the digital signature verification.
Verifier Version	Version of the program responsible for performing the digital signature verification.
ROMMON 2	Section of the output displaying the current ROMmon information.

Displaying the Digital Signature Information for a Specific Image File Example

The following example shows how to display the digital signature information related to software authentication for a specific image file:

Device# **show software authenticity file flash0:**c3900-universalk9-mz.SSA

```
File Name                    : flash0:c3900-universalk9-mz.SSA
Image type                   : Development
    Signer Information
        Common Name          : xxx
        Organization Unit    : xxx
        Organization Name    : xxx
    Certificate Serial Number : xxx
    Hash Algorithm           : SHA512
    Signature Algorithm      : 2048-bit RSA
    Key Version              : A

The table below describes the significant fields shown in the display.
```

Table 35: show software authenticity file Field Descriptions

Field	Description
File Name	Name of the filename in the memory. For example, flash0:c3900-universalk9-mz.SSA refers to filename c3900-universalk9-mz.SSA in flash memory (flash0:).
Image type	Displays the type of image.
Signer Information	Signature information.
Common Name	Displays the name of the software manufacturer.
Organization Unit	Displays the hardware the software image is deployed on.
Organization Name	Displays the owner of the software image.

Field	Description
Certificate Serial Number	Displays the certificate serial number for the digital signature.
Hash Algorithm	Displays the type of hash algorithm used in digital signature verification.
Signature Algorithm	Displays the type of signature algorithm used in digital signature verification.
Key Version	Displays the key version used for verification.

Displaying Digitally Signed Cisco Software Key Information Example

The following example displays digitally signed Cisco software key information. The information details the software public keys that are in storage, including their key types.

```
Device# show software authenticity keys
Public Key #1 Information
-------------------------
Key Type             : Release  (Primary)
Public Key Algorithm : RSA
Modulus :
        CC:CA:40:55:8C:71:E2:4A:3A:B6:9D:5C:94:1D:02:BA:
       ...
        26:04:6B:33:EB:70:2B:18:24:C7:D9:31:3E:77:24:85
Exponent : xxx
Key Version          : A
Public Key #2 Information
-------------------------
Key Type             : Development  (Primary)
Public Key Algorithm : RSA
Modulus :
        CC:CA:40:55:8C:71:E2:4A:3A:B6:9D:5C:94:1D:02:BA:
       ....
        26:04:6B:33:EB:70:2B:18:24:C7:D9:31:3E:77:24:85
Exponent : xxx
Key Version          : A

The table below describes the significant fields shown in the display.
```

Table 36: show software authenticity keys Field Descriptions

Field	Description
Public Key #	Public key number.
Key Type	Displays the key type used for image verification.
Public Key Algorithm	Displays the name of the algorithm used for public key cryptography.
Modulus	Modulus of the public key algorithm.
Exponent	Exponent of the public key algorithm
Key Version	Displays the key version used for verification.

Enabling Debugging of Digitally Signed Cisco Software Image Key Information Example

The following example shows how to enable debugging of software authentication events relating to key information for digitally signed Cisco software:

```
Device# debug software authenticity key
```

Additional References

The following sections provide references related to the Digitally Signed Cisco Software feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
System Management Command Reference	http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst3850/software/release/3se/system_management/c

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Digitally Signed Cisco Software

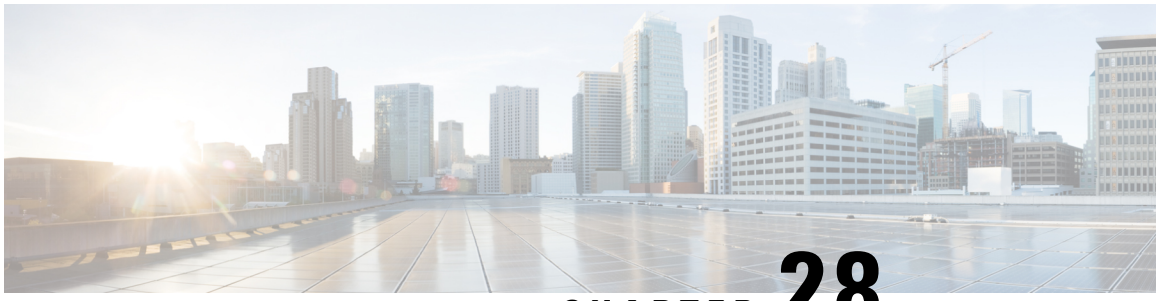
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 37: Feature Information for Digitally Signed Cisco Software

Feature Name	Releases	Feature Information
Digitally Signed Cisco Software		The Digitally Signed Cisco Software feature describes how to identify digitally signed Cisco software, gather software authentication information related to digitally signed images, and perform key revocation. Digitally Signed Cisco software is software that is digitally signed using secure asymmetrical (public-key) cryptography. The following commands were introduced or modified: **debug software authenticity, show software authenticity file, show software authenticity keys, show software authenticity running.**

Feature Name	Releases	Feature Information
Key Revocation Feature Support		Key revocation feature support was added. Key revocation removes a key from a platform's key storage. A platform can host a production or special image, and a production key (from a production image) or special key (from a special image) may be revoked during key revocation. The following section provides information about this feature: • Digitally Signed Cisco Software Key Revocation and Replacement The following commands were introduced or modified: **debug software authenticity, show software authenticity upgrade-status, software authenticity key add, software authenticity key revoke, upgrade rom-monitor file.**



CHAPTER 28

Using FTP to Manage System Images

This module contains information about using FTP to manage Cisco system images.

Image Copying from Flash Memory to an FTP Server

The FTP protocol requires a client to send a remote username and password on each FTP request to a server. When you copy a configuration file from the router to a server using FTP, the Cisco IOS software sends the first valid username it encounters in the following list:

1. The username specified in the **copy** privileged EXEC command, if a username is specified.
2. The username set by the **ipftpusername** global configuration command, if the command is configured.
3. Anonymous.

The router sends the first valid password it encounters in the following list:

1. The password specified in the **copy** privileged EXEC command, if a password is specified.
2. The password set by the **ipftppassword** global configuration command, if the command is configured.

The router forms a password *username* @*routername* .*domain* . The variable *username* is the username associated with the current session, *routername* is the configured hostname, and *domain* is the domain of the router.

The username and password must be associated with an account on the FTP server. If you are writing to the server, the FTP server must be properly configured to accept the FTP write request from the user on the router.

If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user's name as the remote username.

Refer to the documentation for your FTP server for more information.

Use the **ipftpusername** and **ipftppassword** commands to specify a username and password for all copies. Include the username in the **copy** command if you want to specify a username for that copy operation only.

Image Copy from an FTP Server to a Flash Memory File System

You can copy a system image from an FTP server to a flash memory file system.

FTP Username and Password

The FTP protocol requires a client to send a remote username and password on each FTP request to a server. When you copy a configuration file from the router to a server using FTP, the Cisco IOS software sends the first valid username it encounters in the following list:

1. The username specified in the **copy** privileged EXEC command, if a username is specified.
2. The username set by the **ipftpusername** global configuration command, if the command is configured.
3. Anonymous.

The router sends the first valid password it encounters in the following list:

1. The password specified in the **copy** privileged EXEC command, if a password is specified.
2. The password set by the **ip ftp password** command, if the command is configured.

The router forms a password *username @routername .domain* . The variable *username* is the username associated with the current session, *routername* is the configured host name, and *domain* is the domain of the router.

The username and password must be associated with an account on the FTP server. If you are writing to the server, the FTP server must be properly configured to accept the FTP write request from the user on the router.

If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user's name as the remote username.

Refer to the documentation for your FTP server for more information.

Use the **ip ftp username** and **ip ftp password** commands to specify a username and password for all copies. Include the username in the **copy** command if you want to specify a username for that copy operation only.

Copying an Image from Flash Memory to an FTP Server

To copy a system image to an FTP network server, complete the tasks in this section:

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).

Step 3 **ip ftp username** *username*

Example:

```
Router(config)# ip ftp username user1
```

(Optional) Changes the default remote username.

Step 4 **ip ftp password** *password*

Example:

```
Router(config)# ip ftp password guessme
```

(Optional) Changes the default password.

Step 5 **end**

Example:

```
Router(config)# end
```

(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).

Step 6 **show** *flash-filesystem* **:**

Example:

```
Router# show flash:
```

(Optional) Displays the system image file in the specified flash directory. If you do not already know it, note the exact spelling of the system image filename in flash memory.

Step 7 **copy** *flash-filesystem* **:** *filename* **ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*]

Example:

```
Router# copy slot0:1:your-ios ftp://myuser:mypass@172.23.1.129/dirt/sysadmin/your-ios
```

Copies the image to the FTP server.

Note After you have issued the **copy** privileged EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Examples

The following example uses the **showslot1:privileged**EXEC command to display the name of the system image file in the second PCMCIA slot, and copies the file (test) to an FTP server:

```
Router# show slot1:
-#- ED --type-- --crc--- -seek-- nlen -length- -----date/time------ name
1   .. 1        46A11866 2036C   4    746      May 16 1995 16:24:37 test
Router# copy slot1:test ftp://thisuser:thatpass@172.16.13.110/test
writing test!!!!...
successful ftp write.
```

In this example, the file named your-ios is copied from partition 1 of the flash memory PC card in slot 0 to the TFTP server at 172.23.1.129. The file will be saved with the name your-ios in the dirt/sysadmin directory relative to the directory of the remote username.

```
Router# show slot0: partition 1
PCMCIA Slot0 flash directory, partition 1:
File  Length   Name/status
  1   1711088 your-ios
[1711152 bytes used, 2483152 available, 4194304 total]
Router# copy slot0:1:your-ios ftp://myuser:mypass@172.23.1.129/dirt/sysadmin/your-ios
Verifying checksum for 'your-ios' (file # 1)...  OK
Copy 'your-ios' from Flash to server
  as 'dirt/sysadmin/ios-2'? [yes/no] yes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Upload to server done
Flash device copy took 00:00:23 [hh:mm:ss]
```

Copying from an FTP Server to Flash Memory

To copy a system image from an FTP server to a flash memory file system, complete the tasks in this section:

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show** *flash-filesystem* **:**

Example:

```
Router# show flash:
```

(Optional) Displays the system image filename in Flash memory. Use this command to verify the url-path of the file and the exact spelling of the system image filename for use in the next command.

Step 3 **copy** *flash-url* **tftp** :[[[*//location*]*/directory*]*/filename*]

Example:

```
Router# copy slot0:1:your-ios tftp://172.23.1.129/dirt/sysadmin/your-ios
```

Copies the system image from Flash memory to a TFTP server. Specify the file location and filename as the *flash-url* argument.

Note After you have issued the **copy** privileged EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

Step 4 **configure terminal**

Example:

```
Router# configure terminal
```

(Optional) Enters global configuration mode from the terminal. This step is required only if you want to override the default remote username or password (see Steps 3 and 4).

Step 5 **ip ftp username** *username*

Example:

```
Router(config)# ip ftp username netuser1
```

(Optional) Changes the default remote username.

Step 6 **ip ftp password** *password*

Example:

```
Router(config)# ip ftp password guessme
```

(Optional) Changes the default password.

Step 7 **end**

Example:

```
Router(config)# end
```

(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).

Step 8 **copy ftp:** [[[//[*username* [**:***password*]**@**]*location*] */directory*]/*filename*]*flash-filesystem***:**[*filename*]

Example:

```
Router# copy ftp://myuser:mypass@theserver/tftpboot/sub3/c7200-js-mz slot1:c7200-js-mz
```

Copies the configuration file from a network server to running memory or the startup configuration using rcp.

Note After you have issued the **copy** privileged EXEC command, you may be prompted for additional information or for confirmation of the action. The prompting will depend on how much information you provide in the **copy** command and the current setting of the **fileprompt** global configuration command.

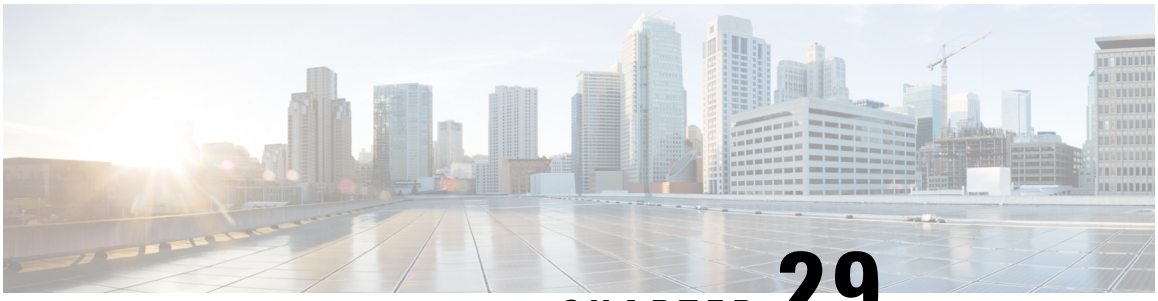
Examples

The following example illustrates how to use the **reload** command to reload the software on the router on the current day at 7:30 p.m.:

```
Router# reload at 19:30
Reload scheduled for 19:30:00 UTC Wed Jun 5 1996 (in 2 hours and 25 minutes)
Proceed with reload? [confirm]
```

The following example illustrates how to use the **reload** command to reload the software on the router at a future time:

```
Router# reload at 02:00 jun 20
Reload scheduled for 02:00:00 UTC Thu Jun 20 1996 (in 344 hours and 53 minutes)
Proceed with reload? [confirm]
```



CHAPTER 29

Configuring the Cisco IOS Auto-Upgrade Manager

The Cisco IOS Auto-Upgrade Manager (AUM) feature simplifies the software image upgrade process by providing a simple interface to specify, download, and upgrade a new Cisco IOS image.

You can upgrade to a new Cisco IOS image in interactive mode by allowing the Auto-Upgrade Manager to guide you through the process. Alternatively, you can perform the upgrade by issuing a single Cisco IOS command or a series of commands. All three methods utilize the Warm Upgrade functionality to perform the upgrade and minimize downtime.

Prerequisites for Cisco IOS Auto-Upgrade Manager

- You must configure the DNS server IP address on the router for a download from Cisco. For more details, refer to the “Configuring the DNS Server IP Address: Example” section and the “Related Documents” section.
- You must configure the Secure Socket Layer (SSL) certificate from the Cisco website (www.cisco.com) on the router for a download from Cisco. This configuration is not required for a download from a non-Cisco server. For more details, refer to the “Configuring the SSL Certificate for a Cisco Download” section and the “Related Documents” section.
- You must register with Cisco Systems for cryptographic software downloads if you want to download cryptographic Cisco IOS software images.

Restrictions for Cisco IOS Auto-Upgrade Manager

The Cisco IOS Auto-Upgrade Manager will not run to completion if the router does not have sufficient memory resource to load and store the requested Cisco IOS software image. The Cisco IOS software image can be downloaded from www.cisco.com only if the current Cisco IOS software image running in the router is a cryptographic image.

Information About Cisco IOS Auto-Upgrade Manager

Cisco IOS Auto-Upgrade Manager Overview

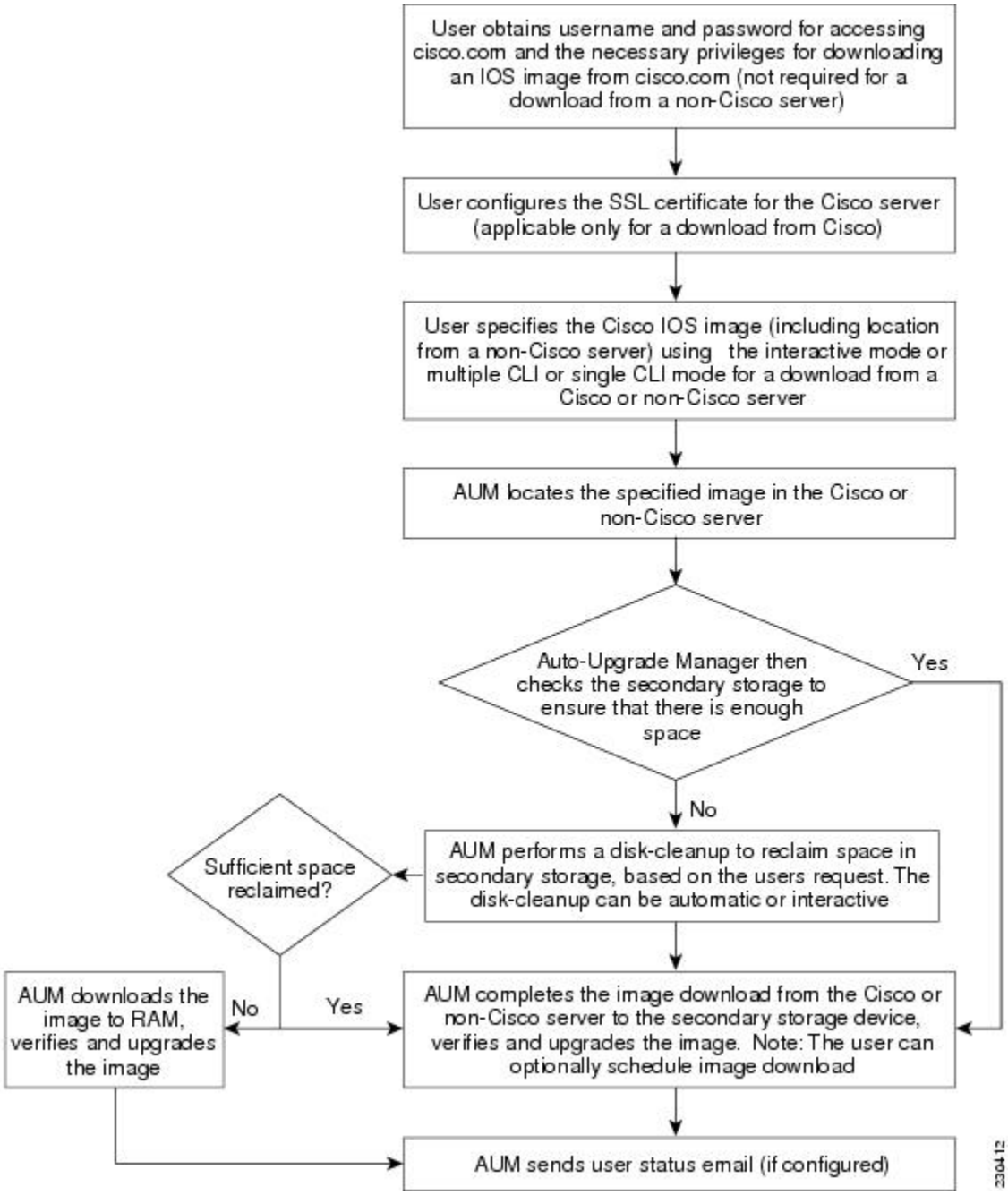
The Cisco IOS Auto-Upgrade Manager streamlines the process of upgrading to a new Cisco IOS software image. You can run the Cisco IOS Auto-Upgrade Manager through the command-line interface (CLI). AUM enables the router to connect to the Cisco website (www.cisco.com) and send the cisco.com username and password for authentication. After authentication, the router passes the name of the Cisco IOS software image that is specified by the user to the Cisco server. The Cisco server returns the complete URL of the Cisco IOS software image to the router.

The Cisco IOS Auto-Upgrade Manager configured on the router can then manage the entire process of upgrading to the Cisco IOS software image. AUM upgrades the router with the software image at the time specified by the user by performing the following tasks:

- Locating and downloading the Cisco IOS software image
- Checking all requirements
- Managing secondary storage space
- Validating the Cisco IOS software image
- Scheduling a warm-upgrade

The figure below illustrates the workflow of the Cisco IOS Auto-Upgrade Manager.

Figure 10: Cisco IOS Auto-Upgrade Manager Workflow



Note If the router fails to load the Cisco IOS software image that you have specified, it displays the error message in the console window and in the syslog buffers indicating the reason for the failure. If the user is not authorized to download encrypted software, an error message is generated requesting the user to register for this service. Similarly, if any CLI configuration statements are not understood by the parser at bootup, it generates an error message and stores the log of the invalid configuration lines in the nvram:invalid-config file. This error message indicates that the Cisco IOS software image that you have specified does not support the same feature set as the old Cisco IOS software image. If the router does not have sufficient secondary storage space to support both the images, but succeeds in the upgrade with the new image, it connects to the Cisco server again and downloads the Cisco IOS software image into a secondary storage. This process erases the existing image.

Specific Cisco IOS Software Image Download from the Cisco Website

You can download a specific Cisco IOS software image from www.cisco.com. AUM uses Secure Socket Layer (SSL) for a secure connection, requiring the user to configure the certificate. The router passes the name of the Cisco IOS software image along with your username and password to log in to the www.cisco.com server. The Cisco server returns the complete URL for the specific Cisco IOS software image to the router.

The Cisco IOS Auto-Upgrade Manager can then automatically download the Cisco IOS software image that you have specified from www.cisco.com, verify it, and upgrade the router with the downloaded image.

Note The Intelligent Download Application (IDA) is the Cisco interface to AUM and is sometimes used interchangeably with the term *Cisco server* in the context of AUM.

Additionally, the Cisco IOS Auto-Upgrade Manager provides the following optional services:

- Disk clean-up utility
- Scheduling of upgrade

These services are available for download from a Cisco or non-Cisco server, both in the interactive and command line modes.

Specific Cisco IOS Software Image Download from a Non-Cisco Server

You can download a Cisco IOS software image that is present on a local or non-Cisco TFTP or FTP server. You can provide an FTP username and password using the **ipftpusername** and **ipftppassword**global configuration commands for an FTP download. The Cisco IOS Auto-Upgrade Manager automates the process of downloading the specific Cisco IOS software image from a non-Cisco server and warm upgrade services. It also provides the disk clean-up utility to delete the files if the space required to download the new Cisco IOS software image is not sufficient.

Interactive and Single Command Line Mode

You can download a specific Cisco IOS software image from www.cisco.com using the CLI or through the following user interfaces:

Interactive Mode

The Auto-Upgrade Manager guides you through the process of upgrading to a new Cisco IOS image in the interactive mode. When you choose automatic upgrade, you are required to answer a few questions in the interactive mode to complete the device upgrade. You can initiate interactive mode by issuing the **upgradeautomatic** command without any options. For more details, refer to the *Cisco IOS Configuration Fundamentals Command Reference*.

Single Command Line Mode

The non-interactive single line CLI is for advanced users. You can download and upgrade to a new Cisco IOS software image from a Cisco or non-Cisco server by using the **upgradeautomaticgetversion** command and specifying all the required arguments. For more details, refer to the *Cisco IOS Configuration Fundamentals Command Reference*.

The interactive mode and single line CLI mode are applicable to downloads from Cisco and non-Cisco servers.

How to Upgrade a Cisco IOS Software Image Using the Cisco IOS Auto-Upgrade Manager

Configuring the SSL Certificate for a Cisco Download

Perform this task to configure the SSL certificate for a Cisco download.

Before you begin

The SSL certificate must be configured to download from cisco.com. The certificate is required for secure HTTP communication. You can obtain the SSL certificate from the Cisco website (www.cisco.com) to configure it on the router.

Perform the following task to obtain the SSL certificate from the Cisco website:

1. Pull down the Tools menu in Internet Explorer (IE) and select Internet Options.
2. Under the Advanced tab, select “Warn if changing between secure and not secure mode.”
3. Enter the URL https://www.cisco.com in IE. When a security alert pop-up box appears, click “No” for the question “You are about to leave a secure Internet connection. Do you want to continue?”.
4. Double-click the lock icon on the status bar of IE. This action opens a dialog box showing the details of the certificate.
5. Click the Certification Path tab. This tab displays the certification chain.
6. Select each CA certificate and click View Certificate. This action opens a details window for the certificate.
7. Select the Details tab of the certificate window displayed, and click Copy to File. This action opens the certificate export wizard.
8. Save the certificate in the Base-64 encoded format to a file (such as cisco.cert).
9. Open the cisco.cert file in a Notepad to get the certificate data that you need to configure on your router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal**
5. **revocation-check none**
6. **exit**
7. **crypto ca authenticate** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint cisco_ssl_cert`	Declares the certification authority (CA) and enters ca-trustpoint configuration mode.
Step 4	**enrollment terminal** **Example:** `Device(ca-trustpoint)# enrollment terminal`	Displays the certificate request on the console terminal and allows you to enter the issued certificate data on the terminal.
Step 5	**revocation-check none** **Example:** `Device(ca-trustpoint)# revocation-check none`	Specifies that certificate checking is not required.
Step 6	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 7	**crypto ca authenticate** *name* **Example:** `Device(config)# crypto ca authenticate cisco_ssl_cert`	Authenticates the CA to your router by obtaining the self-signed certificate of the CA.

Configuring the Cisco IOS Auto-Upgrade Manager

Perform this task to configure the Cisco IOS Auto-Upgrade Manager.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **autoupgrade disk-cleanup** {**crashinfo** | **core** | **image** | **irrecoverable**}
4. **autoupgrade ida url** *url*
5. **autoupgrade status email** {**recipient***email-address* | **smtp-server***name-address*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**autoupgrade disk-cleanup** {**crashinfo** | **core** | **image** | **irrecoverable**} **Example:** `Device(config)# autoupgrade disk-cleanup crashinfo`	Configures the Cisco IOS Auto-Upgrade Manager disk cleanup utility.
Step 4	**autoupgrade ida url** *url* **Example:** `Device(config)# autoupgrade ida url https://www.cisco.com/cgi-bin/new-ida/locator/locator.pl`	Configures the URL of the Cisco server running on www.cisco.com where the image download requests will be sent by Cisco IOS Auto-Upgrade Manager. **Note** This step is required only if the default URL has changed.
Step 5	**autoupgrade status email** {**recipient***email-address* | **smtp-server***name-address*} **Example:** `Device(config)# autoupgrade status email smtp-server smtpserver.abc.com`	Configures the email address and outgoing email server to which the router sends the status email.

Downloading the Cisco IOS Software Image

Perform this task to download the Cisco IOS software image from the Cisco website (www.cisco.com) or from a non-Cisco server.

SUMMARY STEPS

1. **enable**
2. **upgrade automatic getversion** {**ciscousername***username***password***password***image***image* | *url*} [**at***hh***:***mm* | **now** | **in***hh***:***mm*] [**disk-management** {**auto** | **confirm** | **no**}]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**upgrade automatic getversion** {**ciscousername***username***password***password***image***image*	*url*} [**at***hh***:***mm*	**now**	**in***hh***:***mm*] [**disk-management** {**auto**	**confirm**	**no**}] **Example:** `Device# upgrade automatic getversion tftp://abc/tom/c3825-adventerprisek9-mz.124-2.XA.bin at now disk-management auto`	Downloads the image directly from www.cisco.com or a non-Cisco server.

Reloading the Router with the New Cisco IOS software Image

Perform this task to reload the router with the new Cisco IOS software image.

SUMMARY STEPS

1. **enable**
2. **upgrade automatic runversion** [**at***hh***:***mm* | **now** | **in***hh***:***mm*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**upgrade automatic runversion** [**at***hh***:***mm*	**now**	**in***hh***:***mm*]	Reloads the router with the new image.

	Command or Action	Purpose
	Example: `Device# upgrade automatic runversion at 7:30`	**Note** You can also use the **upgradeautomaticgetversion** command to reload the router with the new Cisco IOS software image. But, if you have already downloaded the Cisco IOS software image using the **upgradeautomaticgetversion** command, you must use the **upgradeautomaticrunversion** command to reload the router.

Canceling the Cisco IOS Software Image Reload

Perform this task to cancel a scheduled reload of a specific Cisco IOS software image.

You can cancel an image reload under the following conditions:

- When the scheduled time to reload the router is not sufficient.
- When you do not want to upgrade the router to the new image.

SUMMARY STEPS

1. **enable**
2. **upgrade automatic abortversion**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**upgrade automatic abortversion** **Example:** `Device# upgrade automatic abortversion`	Cancels the Cisco IOS software image upgrade.

Configuration Examples for Cisco IOS Auto-Upgrade Manager

Configuring the DNS Server IP Address Example

You should configure the DNS server IP address on the router before configuring the Cisco IOS Auto-Upgrade Manager. This sequence of events enables the router to use the **ping** command with a hostname rather than an IP address. You can successfully ping the Cisco website (www.cisco.com) after configuring the DNS server IP address on the router. This action also ensures that the router is connected to the Internet.

The following example shows how to configure the DNS server IP address on your router. After configuring the DNS server IP address, you should be able to ping www.cisco.com successfully.

```
configure terminal
 ip domain name mycompany.com
 ip name-server 10.2.203.1
 end
  ping www.cisco.com
```

Configuring the SSL Certificate for a Cisco Download Example

You should configure the SSL certificate of the Cisco server on the router before using the Cisco IOS Auto-Upgrade Manager to download an image from the Cisco website.

The following example shows how to configure the SSL certificate:

```
configure terminal
 crypto pki trustpoint cisco_ssl_cert
  enrollment terminal
  revocation-check none
  exit
 crypto ca authenticate cisco_ssl_cert
!Enter the base 64 encoded CA certificate and end this with a blank line or the word quit
. !The console waits for the user input. Paste the SSL certificate text and press Return.

-----BEGIN CERTIFICATE-----


<The content of the certificate>

-----END CERTIFICATE-----


!Trustpoint 'cisco_ssl_cert' is a subordinate CA and holds a non self signed cert
!Trustpoint 'cisco_ssl_cert' is a subordinate CA.
!but certificate is not a CA certificate.
!Manual verification required
!Certificate has the following attributes:
      ! Fingerprint MD5: 49CE9018 C0CC41BA 1D2FBEA7 AD3011EF
      ! Fingerprint SHA1: A88EAA5D 73D63CB7 BF25197B 9C35ED97 023BB57B

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Configuring the Cisco IOS Auto-Upgrade Manager Example

The following example shows how to configure the Cisco IOS Auto-Upgrade Manager on the router:

```
configure terminal
autoupgrade disk-cleanup crashinfo
autoupgrade ida url https://www.cisco.com/cgi-bin/new-ida/locator/locator.pl
autoupgrade status status email smtp-server
```

Additional References

The following sections provide references related to the Cisco IOS Auto-Upgrade Manager.

Related Documents

Related Topic	Document Title
Cisco IOS Auto-Upgrade Manager commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples.	Cisco IOS Configuration Fundamentals Command Reference
Configuring DNS on Cisco routers	Configuring DNS on Cisco Routers technical note
Warm Upgrade	Warm Upgrade feature module

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Cisco IOS Auto-Upgrade Manager

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 38: Feature Information for Cisco IOS Auto-Upgrade Manager

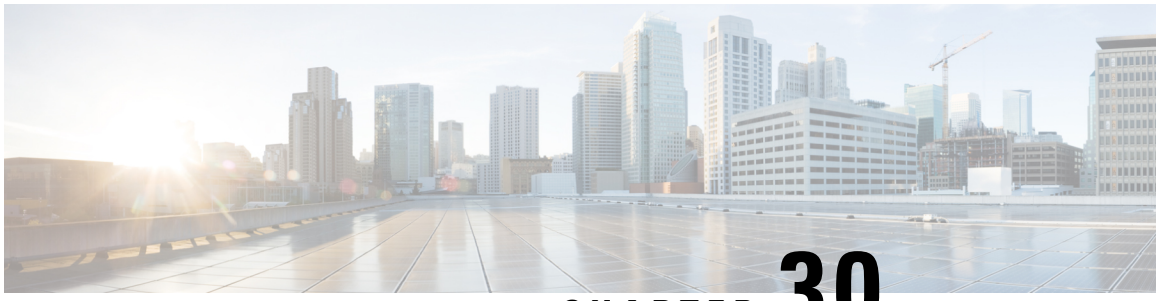
Feature Name	Releases	Feature Information
Cisco IOS Auto-Upgrade Manager	12.4(15)T Cisco IOS XE Release 3.9S	The Cisco IOS Auto-Upgrade Manager simplifies the software image upgrade process by providing a simple interface to specify, download, and upgrade a new Cisco IOS image. In 12.4(15)T, this feature was introduced on the Cisco 1800, Cisco 2800, and Cisco 3800 series routers. This feature was integrated into Cisco IOS XE Release 3.9S. The following commands were introduced or modified by this feature: **autoupgrade disk-cleanup**,**autoupgrade ida url**, **autoupgrade status email**, **debug autoupgrade**, **show autoupgrade configuration unknown**, **upgrade automatic abortversion**, **upgrade automatic getversion**, **upgrade automatic runversion**.

Glossary

CLI --command-line interface

IDA or Cisco server --Intelligent Download Application

Cisco IOS --Cisco Internetworking Operating System



CHAPTER 30

Information About Boot Integrity Visibility

Boot Iintegrity Visibility allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity. Software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code.

During the boot process, the software creates a checksum record of each stage of the bootloader activities.

You can retrieve this record and compare it with a Cisco-certified record to verify if your software image is genuine. If the checksum values do not match, you may be running a software image that is either not certified by Cisco or has been altered by an unauthorized party.

Note Boot Integrity Visibility is supported only on the active supervisor. It does not support high availability scenarios.

- Verifying the Software Image and Hardware, on page 337
- Verifying Platform Identity and Software Integrity, on page 338

Verifying the Software Image and Hardware

This task describes how to retrieve the checksum record that was created during a router bootup. Enter the following commands in privileged EXEC mode.

Note On executing the following commands, you might see the message % Please Try After Few Seconds displayed on the CLI. This does not indicate a CLI failure, but indicates setting up of underlying infrastructure required to get the required output. We recommend waiting for a few minutes and then try the command again.

The messages % Error retrieving SUDI certificate and % Error retrieving integrity data signify a real CLI failure.

1. show platform sudi certificate [sign [nonce nonce]]
2. show platform integrity [sign [nonce nonce]]

Verifying Platform Identity and Software Integrity

Verifying Platform Identity

```
Store-4451# show platform integrity
Platform: ISR4451-X/K9
Boot 0 Version: F01001R06.03c1d3d202013-01-18
Boot 0 Hash: 82597CE130610B8016A6A0FF2851919279857C86966540170E1132C6872A6274
Boot Loader Version: 16.7(4r)
Boot Loader Hash:
5F44054A51B69312283CE03255929D38D938351FDBE7F26A45DCEF6CB7F39C3078C65CB966D71DCF984865D30880AB8D65DD70DB31910B94B0AE290E8DA675E3
OS Version: BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127
OS Hashes:
isr4400-universalk9.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.bin:
8448067652482B991F562E7CB99FC1B1C1437BA7FC968A22C717AD1B5D36D1EE1331B6CCF5C5427FF9D88847D3B849DF482D92D0F631D00BD9A853C065DABEA1
isr4400-firmware_sm_dsp_sp2700.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

A667AFCD2B9819CE88725B90399131BDA06A0B9BFC0DC4835F02E6EFC23347C717DDB6A4659A8C33692344191931D32407EEAA1604F0C152222DE243D5E21D29
isr4400-firmware_nim_shdsl.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

A6E5D11706801FEF7B87B67B71A591176B05955CBA031FAABA23CC41AC715970819F06BD9A85AF945A338E99400211A5061D919C85FA3EC428457F0E498C06C0
isr4400-firmware_nim_ssd.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
F6F6418037171A6C941830BE8481A768C7CFB205F6A807B0196A54B8A2A607C78E6CA26F34BFFAB0C04D0CCA05A1AA5E8AECB6BC8CF7659E826A2F2DC39888DE
isr4400-firmware_nim_ge.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
97752B79EE8AE4925B74A94603CE5FEE5BF899945310 74C55935BF1C79065C474D21F3CF35A9F755110A6875ED425C0A14CA3400D3FB76C47CBEEA1B2A7E3216
isr4400-firmware_sm_async.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
0044338EE6A3E8A8AE61DAA599EB9A2A1B1BD78FBCB2880459FCCF9E750FD5852396777 55CC6ACE4BFDA1CED40A0F63D8A0DF5BAB4DF34DE11D4A42D8FCCFBCC
isr4400-firmware_sm_1t3e3.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
54DA6469D00FF20596FDAD7A2687ED6424180E73DA95A87848CE61143BAB5101186675 9B7CD21F4C77BFCB2219ECE6918A5F60F245E68BA2E22DFB3831CB1B2B
isr4400-firmware_dsp_analogbri.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

9B7B92DBF5B9E2574FC3668A6E2E4F1A0C20D4C895EF99016F51055E56D6195BA41DE31596E8F2D31B5C4B409207F3C04104304E9AEBC4461606B3614CD57F8C
isr4400-firmware_nim_xdsl.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
A2957CD3005499316638B0AE943F77B02882F1B490899EB43E0D052ED57E299AD3FD82D58589AEC97275DB9AB6D12382C99DF41FD3722D40E01AB7E0201B739D
isr4400-firmware_dsp_sp2700.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

4A6422975EDDBD6367F40A0FB6C20888414BCD2D9C78A615F8C853584CE360079533B63B2AE9D10C1C4BCBE2F46F409525927A416E7275A34B2D513635486F54
isr4400-firmware_ngwic_t1e1.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

001B48A89716B10B6B506AAF562495DFD67E8DBF5BF4385A870E8A8B08BAB7A4F7D67230084A344AA9E40B037974E425A8CE289CDB47D06FDF759F56B5D30DB6
isr4400-firmware_sm_10g.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
1B9D88DC2E708015D65A913B42CEF7D42981D2E09EF9B9CEBDC94714F23C6D19D66B9CD5C72F51434A719BDDC640D9F88972BE9CAC742C894A69EBE55694FF67
isr4400-firmware_prince.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
E82220CCB45DD66C2A7A99DFA10758FE5AB8C217624BA623A83D1ADAF87FD08E4FC533C028D8C86B093184479BB064E36DB6255AA15A91381AE287070C1226E4
isr4400-firmware_dreamliner.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

50AE70E6C115E5339A1299E4ED8C123DE8BBC04CA9A45CA11B716C3013FFDCACD73D53FF043D6EFA36655A56F687247AF2D57176BE2142E0ACC506E64BD2A7DD
isr4400-firmware_nim_bri_st_fw.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

338BDCB41132394919D045E6B957D485F3ACBD160C7561EEF0A8155036DC695F6300291E1444E240975D9D02B45F4DFD36F36C5973D4DD9091DF6F71D9BB4157
isr4400-firmware_nim_cwan.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
AF6FD9A79D92382994FFA292E3129C47024E907E1AC05E13BA44F519D1B95863E7BC2E0BF9A2DD82D153A0D0159131CE034253ADC8ECA8E4662787834BC8DA5E
isr4400-firmware_nim_async.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

DE69E388865BC5B0144FBC96996F35143CA1E3920D84EDA1D97A08281289575B1FA0664CC7B81FC834B4FFA8C91DCF177CD5CA8323BD078B85374374F63DFD16
isr4400-mono-universalk9.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:
6E9CCDCA9AD205B2713C0097A0B90B95B61FF267E3BC231916E8E1DE1650131F8168188E7F1CEE4F17A412B83C73D890A9EDC0409AB6BB6F5AA687E043FE154
isr4400-firmware_dsp_tilegx.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

BA452BCB66B279A97519397D6B90C8CF9C4CDF3BF74F41900EEDF0000D711FF03CE62C3B9878C314B5A339C16E0C963FD41C4DE86C3A36BDBD2481C49467B485
```

```
PCR0: C0F992411527603FE21E89331F95A1B9427B396C3210CFE47CD75B144A8A950E
PCR8: D767C72CEC698669B4A909423C56CA5527CF232217CF23B503B60D5C89275B20
```

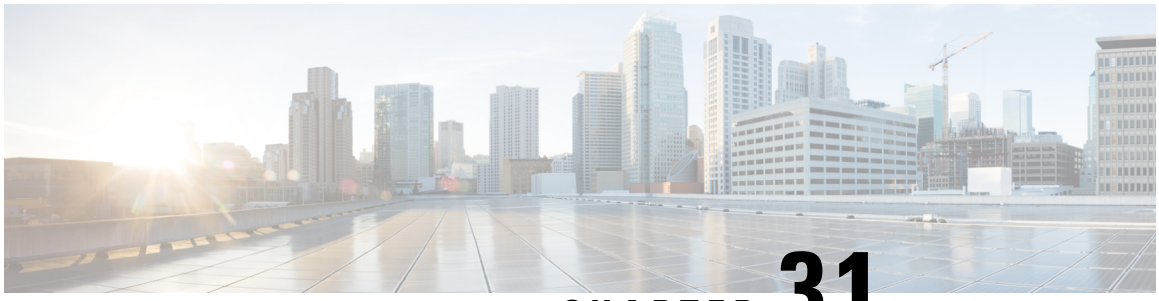
Verifying Software Integrity

The following example displays the checksum record for the boot stages. The hash measurements are displayed for each of the three stages of software successively booted. These hashes can be compared against Cisco-provided reference values. An option to sign the output gives a verifier the ability to ensure the output is genuine and is not altered. A nonce can be provided to protect against replay attacks.

```
show platform integrity sign nonce 123
Platform: C9300-24U
Boot 0 Version: F01144R16.216e68ad62019-02-13
Boot 0 Hash: 523DD459C650AF0F5AB5396060605E412C1BE99AF51F4FA88AD26049612921FF
Boot Loader Version: System Bootstrap, Version 17.1.1r, RELEASE SOFTWARE (P)
Boot Loader Hash:
34A2070D9EAE97E4FC4315A9BAF0E31FFD285E09F0B7F621955607A0FBC1D134ACC0068D8918F15B01975187458F6A46DF0F3DF9EA1593A3CD7BB4DF12487473
OS Version: BLD_POLARIS_DEV_LATEST_20191023_070152
OS Hashes:
cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.bin:
8656F31DE26886F555B93258ADA7F354E083F1AFD22E676D3D83E956F6AA3307F9553E0D94FF752BD6E08DED5DAE067528CE44B16F3DD30A9FB4793E38BAE952
cat9k-wlc.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
33DDC53F932C9EC4CED2B402DA600511D2E2C5F4EF8037CE5D7D8E70B7050936D060467E7533FC7064073F6B3D9ED5AE53F756DD3493A38D564E96E7A49E25E5
cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
4F2057EC660DCDE8EAE08CBE932E035338C7DE0A482B12CB443B506FA2298DE3B8FA1F805A28C0BBBFCDA089AE280E695387016 1DD5E7F0C16C66A75FEB48546
cat9k-webui.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
45F3315C88E57A45F21A508C3771FADF0C8DB952F8848CA1C81F5588FFE466B9AF96295A8247DEFC47CD26A39D1802F0507109897297A4B5A86EFCADB3CFC261
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EE6B1B1920145F5C978374ECB8374917E4E2825B059B7C95D409312C2C19271317AB349F775D4E1860DD0B22E2F68A961566A00466259D93323972F98E8B17E9
cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EAF591B3945F14596A8C8AE8022722B6FC2073DFCEC4D24FE2518CAD7338F73A26F4AD29D00602A56E0B8EF6FAA4463239094BA8446D7B074AAF00930253C281
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
27155ECC5007A7A457C3E3263257613231 7EBF905972454C0305932B9A97591D37AFFC7AB40EC19E7B82DE042B31078309C38F4B81AA756F8D4180662D10F051
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EDC255EC04D267055BE433D60F8CB4CCC426773C12442A291B15838E0D742F99CD45FD01B7E03AC139FDCA3143D83630052B45CCCBC834A84778CFEFF938CBC9
cat9k-espbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
65B0C8305E572247AAFE188A8C0B5081697CDD60BD8501FC2C88A8101862A63FED8B4AEF276D008F03F28978175FC3C4BF0B8FB3C238CDB619952F46CCF19CF1
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
A297AA546323F63751F1CDA42558975D549E83A8D928A6CAFBD5A77AE19C6645620488E5A40E99FD8BE0F9726B12FF9591D3107825B885C9F7C7244FB31491F9
PCR0: 32E782AF9D75D12AC55BA5F67E9E8F375589CAF9C3558BC90E0EB969A84CDE95
PCR8: F0637823517D08D145F3E4DF207673D194FCB437E8B07170887E7AE279F88178
Signature version: 1
Signature:
[redacted]
```

PART VI

Cisco Discovery Protocol



CHAPTER 31

Cisco Discovery Protocol Version 2

Cisco Discovery Protocol (formerly known as CDP) is a Layer 2, media-independent, and network-independent protocol that runs on Cisco devices and enables networking applications to learn about directly connected devices nearby. This protocol facilitates the management of Cisco devices by discovering these devices, determining how they are configured, and allowing systems using different network-layer protocols to learn about each other.

This module describes Cisco Discovery Protocol Version 2 and how it functions with Simple Network Management Protocol (SNMP).

Prerequisites for Using Cisco Discovery Protocol

- Interfaces must support Subnetwork Access Protocol (SNAP) headers.

Restrictions for Using Cisco Discovery Protocol

- Cisco Discovery Protocol functions only on Cisco devices.
- Cisco Discovery Protocol is not supported on Frame Relay multipoint subinterfaces.
- If a neighbor has no IP address on an interface enabled with Cisco Discovery Protocol, the IP address of another interface will be updated as IP address for the non-IP address interface.
- Cisco Discovery Protocol is *not* supported on encapsulation default interface.

Information About Using Cisco Discovery Protocol

VLAN Trunking Protocol

VLAN Trunking Protocol (VTP) is a discovery technique used by switches. A switch advertises its management domain, configuration revision number, VLANs, and their specific parameters on its trunk ports. A VTP domain is made up of a single device or interconnected devices that share the same VTP domain name. A switch can belong to only one VTP domain.

Type-Length-Value Fields

Type-Length-Value (TLV) fields are blocks of information embedded in Cisco Discovery Protocol advertisements. Information in advertisements varies, and the TLV frame format allows for extending advertisements when needed. The table below summarizes the TLV definitions.

Table 39: Type-Length-Value Definitions for Cisco Discovery Protocol Version 2

TLV	Definition
Address TLV	Contains network addresses of both receiving and sending devices.
Application TLV	Provides a mechanism to send an application-specific TLV through Cisco Discovery Protocol.
Capabilities TLV	Identifies the device type, which indicates the functional capability of the device. For example, a switch.
Device-ID TLV	Identifies the device name in the form of a character string.
Full or Half Duplex TLV	Indicates the duplex configuration of the Cisco Discovery Protocol broadcast interface. This information is used by network operators to diagnose connectivity problems between adjacent network devices.
IP Network Prefix TLV	Contains a list of network prefixes to which a sending device can forward IP packets. A prefix includes the interface protocol and the port number. For example, Ethernet 1/0.

TLV	Definition
Location TLV	Delivers location-based information to endpoint devices through access devices (switches or routers) by using Cisco Discovery Protocol. The location TLV can send the following types of information: • Civic location information—Provides the civic address information and the postal information. Examples include street address, road name, and postal community name. • ELIN location information—Provides the location information of a caller. The location is determined by the emergency location identifier number (ELIN), which is a phone number that routes an emergency call to the local public safety answering point (PSAP) and which the PSAP can use to call back the emergency caller. You must configure the location TLV on the device before Cisco Discovery Protocol can deliver location-based information to endpoint devices. For more information about configuring location TLV, see the *Using Link Layer Discovery Protocol in Multivendor Networks* module.
Location-Server TLV	Provides a mechanism for location servers to transfer the required information to neighboring devices.
Native VLAN TLV	Indicates, per interface, the assumed VLAN for untagged packets on the interface. Cisco Discovery Protocol learns the native VLAN for an interface. This field is implemented only for interfaces that support the IEEE 802.1Q protocol.
Platform TLV	Identifies the hardware platform of the device. For example, Cisco 4500.
Port-ID TLV	Identifies the port on which a Cisco Discovery Protocol packet is sent.
Version TLV	Contains the device software release information.
VTP Management Domain TLV	Advertises the configured VLAN Trunking Protocol (VTP)-management-domain name of the system. This name is used by network operators to verify VTP-domain configuration in adjacent network nodes.

Cisco Discovery Protocol

Cisco Discovery Protocol is a Layer 2, media-independent, and network-independent protocol that networking applications use to learn about nearby, directly connected devices. Cisco Discovery Protocol is enabled by default. Each device configured for Cisco Discovery Protocol advertises at least one address at which the device can receive messages and sends periodic advertisements (messages) to the well-known multicast address

01:00:0C:CC:CC:CC. Devices discover each other by listening at that address. They also listen to messages to learn when interfaces on other devices are up or go down.

Advertisements contain time-to-live information, which indicates the length of time a receiving device should hold Cisco Discovery Protocol information before discarding it. Advertisements supported and configured in Cisco software are sent, by default, every 60 seconds on interfaces that support Subnetwork Access Protocol (SNAP) headers. Cisco devices never forward Cisco Discovery Protocol packets. Cisco devices that support Cisco Discovery Protocol store the information received in a table. Information in this table is refreshed every time an advertisement is received, and information about a device is discarded after three advertisements from that device are missed.

The information contained in Cisco Discovery Protocol advertisements varies based on the type of device and the installed version of the operating system. Some of the information that Cisco Discovery Protocol can learn includes:

- Cisco IOS version running on Cisco devices
- Hardware platform of devices
- IP addresses of interfaces on devices
- Locally connected devices advertising Cisco Discovery Protocol
- Interfaces active on Cisco devices, including encapsulation type
- Hostname
- Duplex setting
- VLAN Trunking Protocol (VTP) domain
- Native VLAN

Cisco Discovery Protocol Version 2 provides more intelligent, device-tracking features than those available in Version 1. One of the features available is an enhanced reporting mechanism for more rapid error tracking, which helps to reduce network downtime. Errors reported include mismatched native VLAN IDs (IEEE 802.1Q) on connected ports and mismatched port-duplex states between connected devices. Messages about reported errors can be sent to the console or to a logging server.

You can use **show** commands for getting detailed output on VTP management domains and duplex modes of neighboring devices, counters related to Cisco Discovery Protocol, and VLAN IDs of connecting ports.

Using Cisco Discovery Protocol with SNMP

Using Cisco Discovery Protocol with Simple Network Management Protocol (SNMP) allows network management applications to learn the type of device and the SNMP agent address of neighboring devices. Applications can also send SNMP queries to these neighboring devices.

The SNMP management application learns protocol addresses and types of neighboring devices by retrieving the Cisco Discovery Protocol tables from SNMP agents on those devices. When enabled, the network management module (NMM) SNMP agent discovers neighboring devices and builds its local cache with information about these devices. A management workstation can retrieve this cache by sending SNMP requests to access CISCO-CDP-MIB.

Cisco Discovery Protocol and On-Demand Routing Support for ATM PVCs

Cisco Discovery Protocol and On-Demand Routing (ODR) support exists for ATM point-to-point permanent virtual circuits (PVCs). ODR uses Cisco Discovery Protocol to propagate IP address information in hub-and-spoke topologies. When ODR is enabled, spoke routers automatically advertise their subnets by using Cisco Discovery Protocol.

Cisco Discovery Protocol is disabled by default on ATM PVC interfaces. To enable Cisco Discovery Protocol, use the **cdp run** command in global configuration mode and the **cdp enable** command in interface configuration mode on both ends of the PVC. To enable ODR, use the **router odr** command in global configuration mode on the hub router, and turn off all dynamic-routing protocols on the spoke routers. For details about configuring ODR, see the "Configuring On-Demand Routing" section in the *IP Routing: ODR Configuration Guide*.

Cisco Discovery Protocol Support in IPv6

Cisco Discovery Protocol in IPv6 functions in the same way as in IPv4 and offers the same benefits. The IPv6 enhancement allows Cisco Discovery Protocol to exchange IPv6 and neighbor addressing information. The enhancement also provides IPv6 information to network management products and troubleshooting tools.

Benefits of Cisco Discovery Protocol

Cisco Discovery Protocol provides the following benefits:

- Allows systems using different network layer protocols to learn about one another.
- Facilitates management of Cisco devices by discovering them and discovering how they are configured.
- Assists with troubleshooting Type-Length-Value Fields (TLV) fields.
- Works with SNMP by learning SNMP agent addresses and sending SNMP queries.

How to Use Cisco Discovery Protocol Version 2

Disabling and Enabling Cisco Discovery Protocol on a Cisco Device

Note Upon system restart, CDP will be disabled by default. If CDP is enabled, then the CDP TLV application will also be enabled by default. Should CDP be disabled, the CDP app TLV will be disabled as well. However, if CDP is re-enabled, "no CDP app TLV" will be displayed, which diverges from the default behavior.

Disabling Cisco Discovery Protocol on a Supported Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no cdp run**

4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no cdp run** **Example:** `Device(config)# no cdp run`	Disables Cisco Discovery Protocol on a supported device.
Step 4	**end** **Example:** `Device(config)# end`	Returns the CLI to privileged EXEC mode.

Enabling Cisco Discovery Protocol on a Supported Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp run**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**cdp run** **Example:** `Device(config)# cdp run`	Enables Cisco Discovery Protocol on a supported device.
Step 4	**end** **Example:** `Device(config)# end`	Exits configuration mode and returns to privileged EXEC mode.

Disabling and Enabling Cisco Discovery Protocol on a Supported Interface

Disabling Cisco Discovery Protocol on a Supported Interface

If the encapsulation of an interface is changed, Cisco Discovery Protocol is reenabled on that interface even if Cisco Discovery Protocol was previously disabled. For example, when interface encapsulation changes from PPP to High-Level Data Link Control (HDLC), Cisco Discovery Protocol is reenabled on that interface even though it was explicitly disabled with the **no cdp run** command on that interface. This behavior is by design. The encapsulation changes the Layer 2 protocol configured for that interface and resets the interface configuration to the default Cisco Discovery Protocol state of being enabled, assuming that Cisco Discovery Protocol is enabled globally on the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **no cdp enable**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:**	Configures the specified interface and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**no cdp enable** **Example:** `Device(config-if)# no cdp enable`	Disables Cisco Discovery Protocol on the interface. **Note** If the encapsulation of an interface is changed, Cisco Discovery Protocol is reenabled on that interface even if Cisco Discovery Protocol was previously disabled.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits to privileged EXEC mode.

Example

In the following example, Cisco Discovery Protocol is first disabled on an interface:

```
Device(config)#
Device(config-if)# no ip address

Device(config-if)# shutdown
Device(config-if)# no cdp enable
! Cisco Discovery Protocol is disabled.
Device(config-if)# end
```

Enabling Cisco Discovery Protocol on a Supported Interface

Note If the encapsulation of an interface is changed, Cisco Discovery Protocol is reenabled on that interface, even if Cisco Discovery Protocol was previously disabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **cdp enable**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Device(config)# interface Gigabitethernet 1/0/1`	Configures the specified interface and places the CLI in interface configuration mode. **Note** If the encapsulation of an interface is changed, Cisco Discovery Protocol is reenabled on that interface, even if Cisco Discovery Protocol was previously disabled.
Step 4	**cdp enable** **Example:** `Device(config-if)# cdp enable`	Enables Cisco Discovery Protocol on the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns the CLI to privileged EXEC mode.

Setting the Transmission Timer and Hold Time

Perform this task to set the frequency of Cisco Discovery Protocol transmissions and the hold time for Cisco Discovery Protocol packets.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp timer** *seconds*
4. **cdp holdtime** *seconds*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**cdp timer** *seconds* **Example:** `Device(config)# cdp timer 30`	Specifies the frequency of transmission of Cisco Discovery Protocol packets.
Step 4	**cdp holdtime** *seconds* **Example:** `Device(config)# cdp holdtime 90`	Specifies the time for which a receiving device should hold information before discarding it.
Step 5	**end** **Example:** `Device(config)# end`	Enters privileged EXEC mode.

Disabling and Reenabling Cisco Discovery Protocol Version 2 Advertisements

The broadcasting of Cisco Discovery Protocol Version 2 advertisements is enabled by default on Cisco devices. To disable or reenable this broadcasting, perform these tasks.

Disabling Cisco Discovery Protocol Version 2 Advertisements

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no cdp advertise-v2**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**no cdp advertise-v2** **Example:** `Device(config)# no cdp advertise-v2`	Disables the broadcasting of Cisco Discovery Protocol Version 2 advertisements.
Step 4	**end** **Example:** `Device(config)# end`	Exits to privileged EXEC mode.

Enabling Cisco Discovery Protocol Version 2 Advertisements

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp advertise-v2**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cdp advertise-v2** **Example:** `Device(config)# cdp advertise-v2`	Enables the broadcasting of Cisco Discovery Protocol Version 2 advertisements.
Step 4	**end** **Example:** `Device(config)# end`	Exits to privileged EXEC mode.

Monitoring and Maintaining Cisco Discovery Protocol

Perform this task to monitor and maintain Cisco Discovery Protocol on a device. This task and all its steps are optional, and the steps can be performed in any sequence.

SUMMARY STEPS

1. **enable**
2. **clear cdp counters**
3. **clear cdp table**
4. **show cdp**
5. **show cdp entry** *device-name* [**protocol** | **version**]
6. **show cdp interface** [*type number*]
7. **show cdp neighbors** [*type number*] [**detail**]
8. **show cdp traffic**
9. **show debugging**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear cdp counters** **Example:** `Device# clear cdp counters`	Resets Cisco Discovery Protocol traffic counters to zero.
Step 3	**clear cdp table** **Example:** `Device# clear cdp table`	Clears the table that contains Cisco Discovery Protocol information about neighbors.
Step 4	**show cdp** **Example:** `Device# show cdp`	Displays the interval between advertisements, the number (in seconds) for which an advertisement is valid for a given port, and the version of the advertisement.
Step 5	**show cdp entry** *device-name* [**protocol**	**version**] **Example:** `Device# show cdp entry test-device protocol`	Displays information about a specific neighbor.
Step 6	**show cdp interface** [*type number*] **Example:**	Displays information about interfaces on which Cisco Discovery Protocol is enabled.

	Command or Action	Purpose
	`Device# show cdp interface`	
Step 7	**show cdp neighbors** [*type number*] [**detail**] **Example:** `Device# show cdp neighbors`	Displays the type of device that has been discovered, the name of the device, the number and type of the local interface (port), the time (in seconds) the Cisco Discovery Protocol advertisement is valid for the interface, the device type, the device product number, and the port ID. • The **detail** keyword displays information about the native VLAN ID, the duplex mode, and the VTP domain name associated with neighboring devices.
Step 8	**show cdp traffic** **Example:** `Device# show cdp traffic`	Displays information about Cisco Discovery Protocol traffic, including the number of packets sent and received and checksum errors.
Step 9	**show debugging** **Example:** `Device# show debugging`	Displays information about the types of debugging that are enabled for the device.

Configuration Examples for Cisco Discovery Protocol Version 2

Example: Setting the Transmission Timer and Hold Time

In the following example, the timer is set to send updates every 30 seconds and a **show cdp interface** command is used to verify that the update is effective:

```
Device(config)# cdp timer 30
Device(config)# end
Device# show cdp interface

Serial0 is up, line protocol is up
Encapsulation is HDLC
Sending CDP packets every 30 seconds
Holdtime is 180 seconds
```

In the following example, the hold time is set to 90 seconds and a **show cdp interface** command is used to verify that the update is effective:

```
Device(config)# cdp holdtime 90
Device(config)# end
Device# show cdp interface

Serial0 is up, line protocol is up
```

```
Encapsulation is HDLC
Sending CDP packets every 30 seconds
Holdtime is 90 seconds
```

Example: Monitoring and Maintaining Cisco Discovery Protocol

The following example shows a series of commands that you can use to view Cisco Discovery Protocol information:

Additional References for Cisco Discovery Protocol Version 2

Related Documents

Related Topic	Document Title
Cisco Discovery Protocol commands	Cisco IOS Cisco Discovery Protocol Command Reference
SNMP support configuration tasks	"Configuring SNMP Support" module
On-Demand Routing configuration tasks	"Configuring On-Demand Routing" module
Debug commands	Cisco IOS Debug Command Reference

Standards

Standard	Title
IEEE 802.1Q	*Virtual LANs*

MIBs

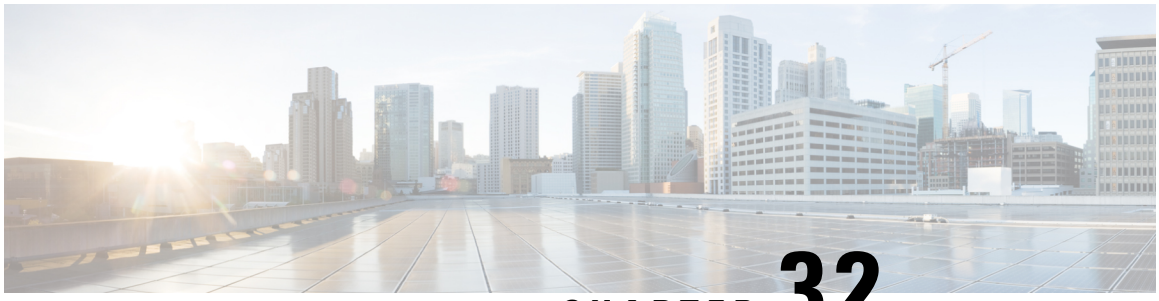
MIB	MIBs Link
CISCO-CDP-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

PART VII

Media Monitoring



CHAPTER 32

Configuring Cisco Mediatrace

This chapter contains information about and instructions for configuring Cisco Mediatrace.

Cisco Mediatrace enables you to isolate and troubleshoot network degradation problems for data streams. Although it can be used to monitor any type of flow, it is primary used with video flows. It can also be used for non-flow related monitoring along a media flow path.

Information About Configuring Cisco Mediatrace

Overview of Cisco Mediatrace

Note Mediatrace is no longer supported on M&T train. For performance monitoring, see Configuring Cisco Performance Monitor, on page 395 chapter.

Cisco Mediatrace helps to isolate and troubleshoot network degradation problems by enabling a network administrator to discover an IP flow's path, dynamically enable monitoring capabilities on the nodes along the path, and collect information on a hop-by-hop basis. This information includes, among other things, flow statistics, and utilization information for incoming and outgoing interfaces, CPUs, and memory, as well as any changes to IP routes or the Cisco Mediatrace monitoring state.

This information can be retrieved in either of two ways:

- By issuing an exec command to perform an on-demand collection of statistics from the hops along a media flow. During this one-shot operation, the hops along the media flow are discovered and shown to you, along with a set of other specified information.
- By configuring Cisco Mediatrace to start a recurring monitoring session at a specific time and on specific days. The session can be configured to specify which metrics to collect, and how frequently they are collected. The hops along the path are automatically discovered as part of the operation.

After collecting the metrics you specified, you can view a report on the metrics.

Cisco Mediatrace is part of the Cisco Medianet family of products. For more information about the design, configuration, and troubleshooting of Mediatrace when used in conjunction with other Cisco products, including a Quick Start Guide and Deployment Guide, see the Cisco Medianet Knowledge Base Portal, located at http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html.

Metrics That You Can Collect Using Cisco Mediatrace

You can collect the following categories of metrics using Mediatrace:

- Common Metrics for Each Responder
- System Metrics: TCP Profile
- System Metrics: RTP Profile
- System Metrics: INTF Profile
- System Metrics: CPU Profile
- System Metrics: MEMORY Profile
- App-Health Metrics: MEDIATRACE-HEALTH Profile
- Metrics for the Mediatrace Request Summary from Initiator

The individual metrics under each of these categories are listed the appropriate section below.

Metics for Mediatrace Request Summary from Initiator

- Request Timestamp
- Request Status
- Number of Hops Responded
- Number of Hops with Valid Data
- Number of Hops with Error
- Number of hops with no data record
- Last Route Change Timestamp
- Route Index

Common Metrics for Each Responder

- Metrics Collection Status
- Reachability address
- Ingress Interface
- Egress Interface
- Mediatrace IP TTL

- Hostname
- Mediatrace Hop Count

Perf-Monitor Metrics: TCP Profile

- Flow Sampling Start Timestamp
- Loss of measurement confidence
- Media Stop Event Occurred
- IP Packet Drop Count
- IP Byte Count
- IP Packet Count
- IP Byte Rate
- IP DSCP
- IP TTL
- IP Protocol
- Media Byte Count
- TCP Connect Round Trip Delay
- TCP Lost Event Count

Perf-Monitor Metrics: RTP Profile

- Flow Sampling Start Timestamp
- Loss of measurement confidence
- Media Stop Event Occurred
- IP Packet Drop Count
- IP Byte Count
- IP Packet Count
- IP Byte Rate
- Packet Drop Reason
- IP DSCP
- IP TTL
- IP Protocol
- Media Byte Rate Average
- Media Byte Count
- Media Packet Count

- RTP Interarrival Jitter Average
- RTP Packets Lost
- RTP Packets Expected (pkts):
- RTP Packet Lost Event Count:
- RTP Loss Percent

System Metrics: INTF Profile

- Collection timestamp
- Octet input at Ingress
- Octet output at Egress
- Packets received with errors at Ingress
- Packets with errors at Egress
- Packets discarded at Ingress
- Packets discarded at Egress
- Ingress interface speed
- Egress interface speed

System Metrics: CPU Profile

- CPU Utilization (1min)
- CPU Utilization (5min)
- Collection timestamp

System Metrics: MEMORY Profile

- Processor memory utilization %
- Collection timestamp

App-Health Metrics: MEDIATRACE-HEALTH Profile

- Requests Received
- Time Last Request Received
- Initiator of Last Request
- Requests Dropped
- Max Concurrent Sessions supported
- Sessions currently active

- Sessions Teared down
- Sessions Timed out
- Hop Info Requests Received
- Performance Monitor Requests Received
- Performance Monitor Requests failed
- Static Policy Requests Received
- Static Policy Requests Failed
- System Data Requests Received
- System Data Requests Failed
- Application Health Requests Received
- Local route change events
- Time of last route change event
- Number of unknown requests received

Overview of Configuring Cisco Mediatrace

Information can be retrieved from Mediatrace by using in either:

- A pre-scheduled, recurring monitoring session.
- An one-shot, on-demand collection of statistics, known as a Mediatrace poll.

Before you can implement a Mediatrace session or poll, you enable Mediatrace on each network node that you want to collect flow information from. You must enable the Mediatrace Initiator on the network node that you will use to configure, initiate, and control the Mediatrace sessions or polls. On each of the network nodes that you want top collect information from, you must enable the Mediatrace Responder.

To configure a Cisco Mediatrace session, you can set session parameters by associating either of two types of pre-packaged profiles with the session:

- video-monitoring profiles
- system-data profiles

You can also configure your own parameters for a Cisco Mediatrace session by configuring the following types of profiles and associating them with the session:

- Path-specifier profile
- Flow-specifier profile
- Sessions-parameters profile

Therefore, the next section describes how to perform the following tasks in order to configure a Cisco Mediatrace session:

1. Enable mediatrace

2. Setup a video-monitoring profile
3. Setup a system-data profile
4. Setup a path-specifier profile
5. Setup a flow-specifier profile
6. Setup a sessions-params profile
7. Associate profiles with a mediatrace session
8. Schedule a mediatrace session

The next section also describes how to execute a mediatrace poll, which is an on-demand fetch of data from the hops on a specific path.

In addition, the next section describes how to manage mediatrace sessions by performing the following tasks:

- Clear incomplete Cisco Mediatrace sessions
- Troubleshoot a Cisco Mediatrace session

Limitations

- Mediatrace does not support IPv6.
- Resource Reservation Protocol (RSVP) does not forward an incoming Path message on the same interface (i.e., through the interface from where it receives the path message). It displays an error some message on the console, “ingress interface = egress interface”. But the Path is sent out on the incoming interface in case of an Performance Routing (PfR) border router.

How to Configure Cisco Mediatrace

Enabling Cisco Mediatrace

For each node you want to monitor using Cisco Mediatrace, you must enable at least the Cisco Mediatrace Responder. You must also enable the Cisco Mediatrace Initiator for all nodes that you want to initiate Mediatrace sessions or polls.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace initiator** {**source-ip** ip-address | **source-interface** *interface-name*} [**force**] **[max-sessions** *number* **]**
4. **mediatrace responder [max-sessions** *number* **]**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace initiator** {**source-ip** ip-address | **source-interface** *interface-name*} [**force**] **[max-sessions** *number* **]** **Example:** `Router(config)# mediatrace initiator source-ip 10.10.1.1 max-sessions 4`	Enables the Cisco Mediatrace or initiator. You can also use the following keywords: • *ip-address* --Any reachable IP address. • *interface-name* --Any local interface that connects to the initiator. • **max-sessions** --Sets the number of Cisco Mediatrace sessions.
Step 4	**mediatrace responder [max-sessions** *number* **]** **Example:** `Router(config)# mediatrace responder max-sessions 4`	Enables the Cisco Mediatrace responder. You can also use the following keywords: • **max-sessions** --Sets the number of Cisco Mediatrace sessions.
Step 5	**end** **Example:** `Router(config)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace responder app-health**command to verify whether the responder is collecting events, requests, and other Cisco Mediatrace related statistics properly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace Video Profile on the Mediatrace Initiator

Cisco Mediatrace provides pre-packaged video-monitoring profiles that contain all of the parameter settings you need to start a video media monitoring session. You can also configure your own video-monitoring profiles on the Mediatrace Initiator.

To initiate a new video media monitoring session, you can associate one of these profiles with a Cisco Mediatrace session when you configure it.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace profile perf-monitor** *name*
4. admin-params
5. **sampling-interval** *seconds*
6. **exit**
7. **metric-list** {**tcp | rtp**}
8. **clock-rate** {*type-number* | *type-name*} *rate*
9. **max-dropout** *number*
10. **max-reorder** *number*
11. **min-sequential** *number*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace profile perf-monitor** *name* **Example:** `Router(config)# mediatrace profile perf-monitor vprofile-2`	Enters perf-prof configuration mode so that you can configure parameters for a Cisco Mediatrace pre-packaged video-monitoring profile.
Step 4	admin-params **Example:** `Router(config-mt-prof-perf)# admin-params`	Enters admin parameters configuration mode so that you can configure video-monitoring admin parameters.
Step 5	**sampling-interval** *seconds* **Example:** `Router(config-mt-prof-perf-params)# sampling-interval 40`	Specifies the interval, in seconds, between samples taken of video-monitoring metrics.
Step 6	**exit** **Example:**	Exits the current configuration mode and returns to perf-prof configuration mode.

	Command or Action	Purpose
	`Router(config-mt-prof-perf-params)# exit`	
Step 7	**metric-list** {**tcp** | **rtp**} **Example:** `Router(config-mt-prof-perf)# metric-list rtp`	Specifies whether the metrics being monitored are for TCP or RTP.
Step 8	**clock-rate** {*type-number* | *type-name*} *rate* **Example:** `Router(config-mt-prof-perf-rtp-params)# clock-rate 64`	(Optional) Specifies the clock rate used to sample RTP video-monitoring metrics. Each payload type has a specific clock rate associated with it and is can specified with either a type number or type name. For the available values of the payload type name, see the Cisco Media Monitoring Command Reference .
Step 9	**max-dropout** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# max-dropout 2`	(Optional) Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics. Dropouts are the number of packets to ignore ahead the current packet in terms of sequence number.
Step 10	**max-reorder** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# max-reorder 4`	(Optional) Specifies the maximum number of reorders allowed when sampling RTP video-monitoring metrics. Reorders are the number of packets to ignore behind the current packet in terms of sequence number.
Step 11	**min-sequential** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# min-sequential 2`	(Optional) Specifies the minimum number of packets in a sequence used to classify a RTP flow .
Step 12	**end** **Example:** `Router(config-mt-prof-perf-rtp-params)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace profile perf-monitor** command to verify that the parameter values for your pre-packaged video-monitoring profiles are set correctly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace System Profile

Cisco Mediatrace provides pre-packaged system-data monitoring profiles that contain all of the parameter settings you need to start a system-data monitoring session. You can also configure your own system-data monitoring profiles. To initiate a new system-data monitoring session, you can associate one of these profiles with a Cisco Mediatrace session when you configure it.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace profile system** *name*
4. **metric-list** {**intf** | **cpu** | **memory**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace profile system** *name* **Example:** `Router(config)# mediatrace profile system system-2`	Enters system profile configuration mode so that you can configure parameters for a Cisco Mediatrace system profile.
Step 4	**metric-list** {**intf**	**cpu**	**memory**} **Example:** `Router(config-sys)# metric-list memory`	Specifies whether the metrics being monitored are for interfaces, the CPU, or the memory.
Step 5	**end** **Example:** `Router(config-sys)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace profile system**commandto verify that the parameter values for your pre-packaged system-data profiles are set correctly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace Path-Specifier Profile

A Cisco Mediatrace session configuration requires a path-specifier profile which defines the parameters that are used to discover the network hops that will be monitored for troubleshooting. The RSVP transport protocol, specified by optional **disc-proto** keyword, is used to do this hop discovery. The parameter values for the flow-specifier should match the values for the media flow that will be traced.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace path-specifier** *name* [**disc-proto rsvp**] {**gsid** *gsid* | **destination ip** *ip-address* **port** *nnnn* }
4. **source ip** *ip-address* **port** *nnnn*
5. **l2-params gateway** *ip-address* **vlan** *vlan-id*
6. **gsid** *gsid*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace path-specifier** *name* [**disc-proto rsvp**] {**gsid** *gsid*	**destination ip** *ip-address* **port** *nnnn* } **Example:** `Router(config)# mediatrace path-specifier path-4 disc-proto rsvp destination ip 10.1.1.1 port 400`	Enters path-specifier configuration mode so that you can configure parameters for a Cisco Mediatrace path-specifier profile. This command requires the name, destination address, and port of the path.
Step 4	**source ip** *ip-address* **port** *nnnn* **Example:** `Router(config-mt-path)# source ip 10.1.1.2 port 600`	Specifies the IP address of the source of the metrics being monitored.

	Command or Action	Purpose
Step 5	**l2-params gateway** *ip-address* **vlan** *vlan-id* **Example:** `Router(config-mt-path)# l2-params gateway 10.10.10.4 vlan 22`	Specifies the IP address and ID of the virtual LAN of the level-2 gateway. **Note** This command is available only on Catalyst platforms.
Step 6	**gsid** *gsid* **Example:** `Router(config-mt-path)# gsid 60606060`	Specifies the metadata global session ID of the flow being monitored.
Step 7	**end** **Example:** `Router(config-mt-path)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace path-specifier** command to verify that the parameter values for your path-specifier profiles are set correctly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace Flow-Specifier Profile

A Cisco Mediatrace session configuration requires a flow-specifier profile which defines the source IP address, destination IP address, source port, destination port, and protocol that identifies a flow. You can associate a profile with an actual Cisco Mediatrace session later when you configure it

For RTP media flows, select UDP as protocol.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace flow-specifier** *name*
4. **source-ip** *ip-address* [**source-port** *port*]
5. **dest-ip** *ip-address* [**dest-port** *port*]
6. **gsid** *gsid*
7. **ip-protocol** {**tcp** | **udp**}
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace flow-specifier** *name* **Example:** `Router(config)# mediatrace flow-specifier flow-6`	Enters flow-specifier configuration mode so that you can configure parameters for a Cisco Mediatrace flow-specifier profile.
Step 4	**source-ip** *ip-address* [**source-port** *port*] **Example:** `Router(config-mt-flowspec)# source-ip 10.1.1.2 source-port 600`	(Optional) Specifies the IP address of the source of the metrics being monitored.
Step 5	**dest-ip** *ip-address* [**dest-port** *port*] **Example:** `Router(config-mt-flowspec)# dest-ip 10.1.1.2 dest-port 600`	Specifies the IP address of the destination of the metrics being monitored.
Step 6	**gsid** *gsid* **Example:** `Router(config-mt-flowspec)# gsid 60606060`	Specifies the metadata global session ID of the flow being monitored.
Step 7	**ip-protocol** {**tcp** | **udp**} **Example:** `Router(config-mt-flowspec)# ip-protocol tcp`	Specifies whether the metrics being monitored are for TCP or UDP.
Step 8	**end** **Example:** `Router(config-mt-flowspec)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace flow-specifier** command to verify that the parameter values for your flow-specifier profiles are set correctly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace Session Parameters Profile

A Cisco Mediatrace session configuration requires a session-params profile, which defines the characteristics of a Cisco Mediatrace session and help it to operate smoothly. You can associate a profile with an actual Cisco Mediatrace session later when you configure it

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace session-params** *name*
4. **response-timeout** *seconds*
5. **frequency** {*frequency* | **on-demand**} **inactivity-timeout** *seconds*
6. **history** *buckets*
7. **route-change reaction-time** *seconds*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace session-params** *name* **Example:** `Router(config-mt-sesparam)# mediatrace session-params qos-2`	Enters session-params configuration mode so that you can configure parameters for a Cisco Mediatrace session-params profile.
Step 4	**response-timeout** *seconds* **Example:** `Router(config-mt-sesparam)# response-timeout 8`	Specifies the amount of time, in seconds, the initiator will wait for a response from the responder.
Step 5	**frequency** {*frequency* | **on-demand**} **inactivity-timeout** *seconds* **Example:** `Router(config-mt-sesparam)# frequency 4 inactivity-timeout 2`	Specifies the interval, in seconds, between samples taken of session-params metrics and the amount of time, in seconds, the initiator will remain active without any activity from the responder.

	Command or Action	Purpose
Step 6	**history** *buckets* **Example:** `Router(config-mt-sesparam)# history 2`	Specifies the number of historical data sets kept, up to a maximum of ten.
Step 7	**route-change reaction-time** *seconds* **Example:** `Router(config-mt-sesparam)# route-change reaction-time 8`	Specifies the amount of time, in seconds, the initiator will wait for the responder to react to its additional route changes. The range is seconds.
Step 8	**end** **Example:** `Router(config-mt-sesparam)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace session-param**command to verify that the parameter values for your session-parameters profiles are set correctly.

For more information about this command, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Configuring a Cisco Mediatrace Session

The Cisco Mediatrace session configuration links the various profiles to a session. Only one of each type of profile can be associated with a Cisco Mediatrace session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace** *session-number*
4. **trace-route**
5. **path-specifier** {[**forward**] *path-name* | **reverse** *path-name* }
6. **session-params** *name*
7. **profile system** *name*
8. **profile perf-monitor** *name* **flow-specifier** *flow-specifier-name*
9. **profile snmp** *name*
10. **profile custom** *name*
11. **last-node** { **auto** | **address** *address* }
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace** *session-number* **Example:** `Router(config)# mediatrace 157`	Enters session configuration mode.
Step 4	**trace-route** **Example:** `Router(config-mt-session)# trace-route`	Enables the running of trace route for the Cisco Mediatrace session. By default trace route is enabled. To stop running trace route, use the **no** form of this command.
Step 5	**path-specifier** {[**forward**] *path-name* | **reverse** *path-name* } **Example:** `Router(config-mt-session)# path-specifier path-4`	Associates a path-specifier profile with the Cisco Mediatrace session.
Step 6	**session-params** *name* **Example:** `Router(config-mt-session)# session-params session-6`	Associates a session-parameters profile with the Cisco Mediatrace session.
Step 7	**profile system** *name* **Example:** `Router(config-mt-session)# profile system sys-2`	Associates a system profile with the Cisco Mediatrace session.
Step 8	**profile perf-monitor** *name* **flow-specifier** *flow-specifier-name* **Example:** `Router(config-mt-session)# profile perf-monitor monitor-6 flow-specifier flow-4`	Associates a perf-monitor profile and flow-specifier with the Cisco Mediatrace session.
Step 9	**profile snmp** *name* **Example:**	Associates an SNMP profile with the Cisco Mediatrace session.

	Command or Action	Purpose
	`Router(config-mt-session)# profile snmp snmp-2`	
Step 10	**profile custom** *name* **Example:** `Router(config-mt-session)# profile custom cp-2`	Associates an SNMP profile with the Cisco Mediatrace session.
Step 11	**last-node** { **auto** | **address** *address* } **Example:** `Router(config-mt-session)# last-node address 10.1.1.1`	Configures the last node for the Cisco Mediatrace session.
Step 12	**end** **Example:** `Router(config-mt-session)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace session**command to display the parameter settings for a specific session or all sessions.

Use the **show mediatrace responder app-health** command and the **show mediatrace responder sessions**command to determine the status of the nodes being monitored.

If Cisco Mediatrace is not collecting all of the data that you want, use the **debug mediatrace**command.

For more information about these commands, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Scheduling a Cisco Mediatrace Session

Once you have configured a Cisco Mediatrace session, you can schedule it to begin when you want to start collecting the data. If the Cisco Mediatrace session is designed to collect performance monitoring metrics, it goes out to enable the Performance Monitor when the session begins.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace schedule** *session ID* [*life* {**forever** | *secs*}] [**start-time** {*hh*:*mm*:[:*ss*][*month day*| *day month*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *secs*] [**recurring**]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mediatrace schedule** *session ID* [*life* {**forever** | *secs*}] [**start-time** {*hh*:*mm*:[:*ss*][*month day*| *day month*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *secs*] [**recurring**] **Example:** `Router(config)# mediatrace schedule 22 life 40 start-time 10:00:00 AUG 20 recurring`	Specifcies when the session will occur. Use these settings: • *session ID* --Which session to run. • **life** --Amount of time the session lasts, either the number of seconds or forever. • **start-time** --When the session starts, whether it is at a specified time and date, pending an event, immediately, or after a specified time and date. • **ageout** --Timeout before removing the session configuration on the initiator. • **recurring** --Session reoccurs at the specified time.
Step 4	**end** **Example:** `Router(config-mt-sched)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **show mediatrace session**command to verify that the intended values are set for the parameters for a specific session or all sessions.

Use the **show mediatrace responder app-health** command and the **show mediatrace responder sessions**command to determine the status of the nodes being monitored.

If Cisco Mediatrace is not collecting all of the data that you want, use the **debug mediatrace**command.

For more information about these commands, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Clearing a Cisco Mediatrace Session

You can clear incomplete mediatrace sessions on the Initiator by using the **clear mediatrace incomplete-sessions**command as described below. This coammand also cleans up all Performance Monitor settings that were configured by Cisco Mediatrace. For sessions created by the config commands, use the **no**

mediatrace schedulecommand. The cleanup triggers a "session teardown" message to RSVP followed by a cleanup of the local mediatrace sessions database.

SUMMARY STEPS

1. **enable**
2. **clear mediatrace incomplete-sessions**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear mediatrace incomplete-sessions** **Example:** `Router# clear mediatrace incomplete-sessions`	Clears incomplete mediatrace sessions.
Step 3	**end** **Example:** `Router# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the status of your Cisco Mediatrace session, use the **show mediatrace responder sessions**command.

For more information about these commands, see the How to Troubleshoot and Monitor a Cisco Mediatrace Session, on page 382.

Executing a Cisco Mediatrace Poll

Cisco Mediatrace polls are used to perform an on-demand fetch of data from the hops on a specific path. Some examples of how it can be used are:

- To retrieve data using a pre-configured session. In this case, no other parameters have to be specified inline. The pre-configured session must be have the frequency type set to on-demand.
- To retrieve the system data, hop or video monitoring information from hops along the specified path. You can specify the path as a pre-configured path-specifier or an inline path specification, in case you do not have config mode privileges. Note that by default, Cisco Mediatrace tries to configure nodes along the path to report passive monitoring metrics, and then waits for a configurable amount of time before going out again to collect the data.

- The **configless** keyword can be used to fetch data from the nodes along a media path, which already have Performance Monitor policies configured using the Performance Monitor commands. Some key things to keep in mind when fetching data using this method are that:
  - The default perf-monitor profile or associated perf-monitor profile will have a sampling interval. If the sampling interval of the static policy does not match the one in the associated perf-monitor profile, no data is returned.
  - If there is no Performance Monitor policy configured on a Responder node, the Cisco Mediatrace responder does not try to configure Performance Monitor and simply reports error to the initiator.

SUMMARY STEPS

1. **enable**
2. **mediatrace poll** {**no-traceroute** | **session** *number* | [**timeout** *value*] **path-specifier**{**name** *path-name* | **gsid** *gsid* | {[**disc-proto rsvp**] **destination ip** *ip-address* [**port** *nnnnn*] | **source ip** *ip-address* [**port** *nnnnn*] **destination ip** *ip-address* [**port** *nnnn*] [**ip-protocol**{**tcp** | **udp**}]} {**app-health** | **hops** | **l2-params gateway** *ip-address* | **system** [**profile** *system-profile-name*] | [**configless**] **perf-monitor** [**profile** *profile-name*]} {**flow-specifier** *name* | **source-ip** *ipaddress* [**source-port** *nnnnn*] **dest-ip** *ipaddress* [**dest-port** *nnnnn*] **ip-protocol** {**tcp** | **udp**}}}}
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**mediatrace poll** {**no-traceroute** | **session** *number* | [**timeout** *value*] **path-specifier**{**name** *path-name* | **gsid** *gsid* | {[**disc-proto rsvp**] **destination ip** *ip-address* [**port** *nnnnn*] | **source ip** *ip-address* [**port** *nnnnn*] **destination ip** *ip-address* [**port** *nnnn*] [**ip-protocol**{**tcp** | **udp**}]} {**app-health** | **hops** | **l2-params gateway** *ip-address* | **system** [**profile** *system-profile-name*] | [**configless**] **perf-monitor** [**profile** *profile-name*]} {**flow-specifier** *name* | **source-ip** *ipaddress* [**source-port** *nnnnn*] **dest-ip** *ipaddress* [**dest-port** *nnnnn*] **ip-protocol** {**tcp** | **udp**}}}} **Example:** **Example:** `Router# mediatrace poll session 22`	Performs an on-demand fetch of data from the hops on a specific path. You can specify the hops using one of the following types of information: • A session definition or its constituent parameters • A system profile definition or its constituent parameters • A combination of a path-specifier profile definition and a perf-monitor profile definition or their constituent parameters **Note** The **l2-params gateway** keyword is available only on Catalyst platforms.
Step 3	**end** **Example:** `Router# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If Cisco Mediatrace is not collecting all of the data that you want:

- Use the **show mediatrace session**command to verify that the intended values are set for the parameters for a specific session or all sessions.
- Use the **show mediatrace responder app-health** command and the **show mediatrace responder sessions**command to determine the status of the nodes being monitored.
- Use the **debug mediatrace**command to view error messages.

Examples

Tip For examples of poll output, see Configuration Examples for Cisco Mediatrace, on page 389.

The following example shows how to fetch the default system metrics when the source IP address, source port, and destination port are not known. Cisco Mediatrace uses the best local IP address as source IP address to find which hops are using RSVP.

mediatrace poll path dest *ip-address* **system**

The following example shows how to fetch the default system metrics when the source and destination port numbers are not known. RSVP finds the hop between the specified source and destination.

mediatrace poll path source *ip-address* **dest** *ip-address* **system**

The following example shows how to fetch the default system metrics when the source and destination port numbers are known. RSVP finds the hop using this information.

mediatrace poll path source-ip *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp system**

The following example shows how to fetch the default set of RTP metrics when the source and destination port numbers are not known. Cisco Mediatrace uses the path source and destination IP addresses to find the hops as well as filter the Performance Monitor data.

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor**

The following example shows how to fetch the default set of RTP metrics. Cisco Mediatrace uses the path parameters to discover hops and uses the inline flow specifier profile as a filter for Performance Monitor data.

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp**

The following example shows how to fetch the default set of TCP metrics. Cisco Mediatrace uses the path parameters to discover hops and uses the inline flow-specifier profile as a filter for Performance Monitor data.

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol tcp**

The following example shows how to fetch the default set of RTP metrics. Cisco Mediatrace uses the best local IP address as source IP address for finding hops on the path and uses the inline flow specifier profile as a filter for Performance Monitor data.

mediatrace poll path dest *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp**

The following example shows how to fetch the default set of TCP metrics. Cisco Mediatrace uses the best local IP address as source IP address for finding hops on the path and uses the inline flow-specifier profile as a filter for Performance Monitor data.

mediatrace poll path dest *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol tcp**

The following example shows how to fetch the default set of RTP metrics from the static policy that is already configured on the hops. The command does not configure the Performance Monitor. Cisco Mediatrace uses the path parameters to discover hops and use the inline flow specifier profile as a filter for Performance Monitor data.

mediatrace poll path source *ip-address* **dest** *ip-address* **configless perf-monitor flow-specifier source** *ip-address* **port** *nnnn* **dest** *ip-address* **port** *nnnn* **ip-protocol udp**

Poll Output Example

This example shows the output is produced by the following hops poll command:

```
mediatrace poll path-specifier source 10.10.130.2 destination 10.10.132.2 hops
Started the data fetch operation.
Waiting for data from hops.
This may take several seconds to complete...
Data received for hop 1
Data received for hop 2
Data fetch complete.
Results:
Data Collection Summary:
  Request Timestamp: 22:47:56.788 PST Fri Oct 29 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 2
  Number of hops with valid data report: 2
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
    Number of Mediatrace hops in the path: 2
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Reachability Address: 10.10.12.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Reachability Address: 10.10.34.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
```

How to Troubleshoot and Monitor a Cisco Mediatrace Session

Use the **show** commands described in this section to troubleshoot to monitor a Cisco Mediatrace session.

Tip For sample outputs, see the Examples section, in this chapter.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **show mediatrace profile perf-monitor** [*name*]
4. **show mediatrace profile system** [*name*]
5. **show mediatrace flow-specifier** [*name*]
6. **show mediatrace path-specifier** [*name*]
7. **show mediatrace initiator**
8. **show mediatrace session-params** [*name*]
9. **show mediatrace session** [**config**| **data**| **stats**| **hops**] [**brief**| *ID*]
10. **show mediatrace responder app-health**
11. **show mediatrace responder sessions [** *global-session-id* **| brief | details]**
12. **debug mediatrace** {**event** | **trace** | **error**} [**initiator** | **responder**| *session-id*]
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**show mediatrace profile perf-monitor** [*name*] **Example:** `Router(config)# show mediatrace profile perf-monitor vprofile-4`	Displays the parameters configured for all pre-packaged video-monitoring profiles or the specified profile.
Step 4	**show mediatrace profile system** [*name*] **Example:** `Router(config)# show mediatrace profile system system-8`	Displays the parameters configured for all pre-packaged system-data monitoring profiles or the specified profile.
Step 5	**show mediatrace flow-specifier** [*name*] **Example:** `Router(config)# show mediatrace flow-specifier flow-2`	Displays the parameters configured for all flow-specifier profiles or the specified flow-specifier profile.
Step 6	**show mediatrace path-specifier** [*name*] **Example:** `Router(config)# show mediatrace path-specifier path-6`	Displays the parameters configured for all path-specifier profiles or the specified path-specifier profile.

	Command or Action	Purpose
Step 7	**show mediatrace initiator** **Example:** `Router(config)# show mediatrace initiator`	Displays the parameters configured for the initiator profile.
Step 8	**show mediatrace session-params** [*name*] **Example:** `Router(config)# show mediatrace session-params sysparams-2`	Displays the monitoring parameters for the session like frequency, response timeout, ands so on. the parameters configured for all pre-packaged system-data monitoring profiles or the specified profile.
Step 9	**show mediatrace session** [**config**| **data**| **stats**| **hops**] [**brief**| *ID*] **Example:** `Router(config)# show mediatrace session data 1002`	Displays the parameters configured for all session profiles or the specified session profile. Use the following keywords to display the corresponding information: • **config** --Configuration of the session. • **data** --All data records collected and still cached at the Initiator. • **stats** --Statistics for this service path or session. • **hops** --Prior service paths (if available) and current service paths discovered. Also shows where and when the last route change happened. • **brief** -- Only a list of sessions with ID, destination/source address/port, and their role association as Initiator or Responder. • *ID* -- Session ID and some state information.
Step 10	**show mediatrace responder app-health** **Example:** `Router(config)# show mediatrace responder app-health`	Displays the current status of the responder.
Step 11	**show mediatrace responder sessions [** *global-session-id* **| brief | details]** **Example:** `Router(config)# show mediatrace responder sessions`	Displays the information about all or specific active sessions on local responder. Use the following keywords to display the corresponding information • *global-session-id* -- ID of the session for which information is displayed. • **brief** --Displays only the destination and source address/port of the path, their role as either Initiator or Responder, and some state information. • **details** --Displays all information.

	Command or Action	Purpose
Step 12	**debug mediatrace** {**event** | **trace** | **error**} [**initiator** | **responder**| *session-id*] **Example:** `Router(config)# debug mediatrace event 24`	Enables debugging for a particular path, or a particular session, or for all Initiator and Responder functions. You can use the following options: • **event** -- Displays only event information. • **trace** -- Displays only trace information. • **error** -- Displays only errors. • **initiator** -- Displays information for only the initiator. • **responder** -- Displays information for only the responder. • *session-id* -- Displays information for only the session.
Step 13	**end** **Example:** `Router(config)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Examples

Note For a complete description of the output for the following show commands, see the *Cisco Media Monitoring Command Reference*.

The following example displays video-monitoring profiles:

```
Router# show mediatrace profile perf-monitor
Perf-monitor Profile: vprof-4
Metric List: rtp
RTP Admin Parameter:
  Max Dropout: 5
  Max Reorder: 5
  Min Sequential: 5
Admin Parameter:
  Sampling Interval (sec): 30
```

The following example displays system-data profiles:

```
Router# show mediatrace profile
system

System Profile: sys-1
Metric List: intf
```

The following example displays flow-specifier profiles:

```
Router# show mediatrace
 flow-specifier flow-1
Flow Specifier: flow-1
    Source address/port:
    Destination address/port:
    Protocol: udp
```

The following example displays path-specifier profiles:

```
Router# show mediatrace
 path-specifier flow-1
Path Configuration: ps1
    Destination address/port: 10.10.10.1
    Source address/port: 10.10.10.4
    Gateway address/vlan:
    Discovery protocol: rsvp
```

The following example displays the initiator profile:

```
Router# show mediatrace
 initiator
Version: Mediatrace 1.0
Mediatrace Initiator status: enabled
Source IP: 1.1.1.1
Number of Maximum Allowed Active Session: 127
Number of Configured Session: 1
Number of Active Session    : 0
Number of Pending Session   : 0
Number of Inactive Session  : 1
Note: the number of active session may be higher than max active session
      because the max active session count was changed recently.
```

The following example displays session profiles:

```
Router# show mediatrace session-params
Session Parameters: s-1
    Response timeout (sec): 60
    Frequency: On Demand
    Inactivity timeout (sec): 300
    History statistics:
       Number of history buckets kept: 3
    Route change:
       Reaction time (sec): 5
```

The following example displays Mediatrace session statistics:

```
Router# show mediatrace session stats 2
Session Index: 2
Global Session Id: 86197709
Session Operation State: Active
Operation time to live: Forever
Data Collection Summary:
  Request Timestamp: 23:55:04.228 PST Fri Oct 29 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 2
  Number of Non Mediatrace hops responded: 0
  Number of hops with valid data report: 2
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
```

```
  Route Index: 0
    Number of Mediatrace hops in the path: 2
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Metrics Collection Status: Success
      Reachability Address: 10.10.12.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
      Traceroute data:
        Address List: 1.2.2.3
        Round Trip Time List (msec): 12 msec
```

Note The rest of the data for hop 1 is similar to the data for hop 2, as shown below.

```
Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Metrics Collection Status: Success
      Reachability Address: 10.10.34.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
      Metrics Collected:
        Collection timestamp: 23:55:04.237 PST Fri Oct 29 2010
        Octet input at Ingress (KB): 929381.572
        Octet output at Egress (MB): 1541.008502
        Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
        Pkts discarded at Egress (pkts): 0
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
```

The following example displays Mediatrace session configuration information:

```
Router# show mediatrace session config 2
Global Session Id: 93642270
---------------------------
Session Details:
    Path-Specifier: ps1
    Session Params: sp1
    Collectable Metrics Profile: intf1
    Flow Specifier:
Schedule:
   Operation frequency (seconds): 30  (not considered if randomly scheduled)
   Next Scheduled Start Time: Start Time already passed
   Group Scheduled : FALSE
   Randomly Scheduled : FALSE
   Life (seconds): Forever
   Entry Ageout (seconds): never
   Recurring (Starting Everyday): FALSE
   Status of entry (SNMP RowStatus): Active
History Statistics:
   Number of history Buckets kept: 10
```

The following example displays Mediatrace session hops:

```
show mediatrace session hops 2
Session Index: 2
Global Session Id: 93642270
Session Operation State: Active
Data Collection Summary:
  Request Timestamp: 13:40:32.515 PST Fri Jun 18 2010
```

```
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 3
  Number of hops with valid data report: 3
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
  Route Index: 0
    Number of Mediatrace hops in the path: 3
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
    Mediatrace Hop Number: 3 (host=responder3, ttl=252)
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
```

The following example displays Mediatrace session data:

```
Router# show mediatrace session data 2
Session Index: 2
Global Session Id: 35325453
Session Operation State: Active
Bucket index: 1
Data Collection Summary:
  Request Timestamp: 13:02:47.969 PST Fri Jun 18 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 3
  Number of hops with valid data report: 3
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
  Route Index: 0
    Number of Mediatrace hops in the path: 3
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Metrics Collection Status: Success
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
      Metrics Collected:
        Collection timestamp: 13:04:57.781 PST Fri Jun 18 2010
        Octet input at Ingress (KB): 10982.720
        Octet output at Egress (KB): 11189.176
    Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
        Pkts discarded at Egress (pkts): 0
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Metrics Collection Status: Success
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
      Metrics Collected:
        Collection timestamp: 13:04:57.792 PST Fri Jun 18 2010
        Octet input at Ingress (MB): 1805.552836
        Octet output at Egress (MB): 1788.468650
        Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
        Pkts discarded at Egress (pkts): 0
```

```
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
```

The following example displays application health information for the Mediatrace responder:

```
Router# show mediatrace responder app-health
Mediatrace App-Health Stats:
   Number of all requests received: 0
   Time of the last request received:
   Initiator ID of the last request received:  0
   Requests dropped due to queue full: 0
   Responder current max sessions: 45
   Responder current active sessions: 0
   Session down or tear down requests received: 0
   Session timed out and removed: 0
   HOPS requests received: 0
   VM dynamic polling requests received: 0
   VM dynamic polling failed: 0
   VM configless polling requests received: 0
   VM configless polling failed: 0
   SYSTEM data polling requests received: 0
   SYSTEM data polling requests failed: 0
   APP-HEALTH polling requests received: 0
   Route Change or Interface Change notices received: 0
   Last time Route Change or Interface Change:
   Unknown requests received: 0
```

The following example displays brief session information for the Mediatrace responder:

```
Router# show mediatrace responder sessions brief
Local Responder configured session list:
Current configured max sessions: 45
Current number of active sessions: 0
session-id initiator-name       src-ip          src-port   dst-ip          dst-port det-l
  2   host-18     10.10.10.2     200   10.10.10.8     200
```

Configuration Examples for Cisco Mediatrace

Example Basic Mediatrace Configuration

The topology for this example includes:

- One mediatrace initiator (10.10.12.2)
- Two mediatrace responders between:
    - A media source (10.10.130.2)
    - A destination (10.10.132.2)

In this example, there is an RTP traffic stream from the source (address=10.10.130.2, port=1000, to the destination (address=10.10.132.2, port=2000).

The basic configuration of the mediatrace responder is as follows:

```
mediatrace responder
snmp-server community public RO
```

The basic configuration of the mediatrace initiator is as follows:

```
mediatrace initiator source-ip 10.10.12.2
mediatrace profile system intf1
mediatrace profile perf-monitor rtp1
mediatrace path-specifier path1 destination ip 10.10.132.2 port 2000
 source ip 10.10.130.2 port 1000
mediatrace flow-specifier flow1
 source-ip 10.10.130.2 source-port 1000
 dest-ip 10.10.132.2 dest-port 2000
mediatrace session-params sp1
 response-timeout 10
 frequency 60 inactivity-timeout 180
mediatrace 1
 path-specifier path1
 session-params sp1
 profile perf-monitor rtp1 flow-specifier flow1
mediatrace schedule 1 life forever start-time now
mediatrace 2
 path-specifier path1
 session-params sp1
 profile system intf1
mediatrace schedule 2 life forever start-time now
```

A sample reverse mediatrace configuration is given below.

```
Device# show mediatrace initiator
Mediatrace Initiator Software Version: 3.0
Mediatrace Protocol Version: 1
Mediatrace Initiator status: enabled

Source IP: 10.10.1.1
Source IPv6:

Number of Maximum Allowed Active Session: 8
Number of Configured Session: 3
Number of Active Session    : 2
Number of Pending Session   : 0
Number of Inactive Session  : 1
Number of Total Proxy Session     : 1
Number of Active Proxy Session    : 1
Number of Pending Proxy Session   : 0
Number of Inactive Proxy Session  : 0

Note: the number of active session may be higher than max active session
      because the max active session count was changed recently.

Device# show run
Device# show running-config | show mediatrace
mediatrace responder
mediatrace initiator source-ip 10.10.1.1
mediatrace profile perf-monitor MT_PERF_RTP
mediatrace path-specifier MT_PATH destination ip 10.11.1.10 port 21064
 source ip 10.10.1.11 port 28938
mediatrace path-specifier MT_PATH2 destination ip 10.10.10.10 port 16514
 source ip 10.10.1.10 port 16558
mediatrace flow-specifier MT_FLOW
 source-ip 10.10.1.11 source-port 28938
 dest-ip 10.10.1.50 dest-port 21064
mediatrace flow-specifier MT_FLOW2
 source-ip 10.1.1.50 source-port 21064
 dest-ip 10.1.1.11 dest-port 28938
mediatrace session-params MT_PARAMS
```

```
 response-timeout 50
 frequency 60 inactivity-timeout 180
 history data-sets-kept 10
mediatrace reverse 155
 path-specifier forward/reverse MT_PATH/MT_PATH2
 session-params MT_PARAMS
 profile perf-monitor MT_PERF_RTP flow-specifier MT_FLOW2
mediatrace schedule 155 life forever start-time now
mediatrace 157
 path-specifier MT_PATH
 session-params MT_PARAMS
 profile perf-monitor MT_PERF_RTP flow-specifier MT_FLOW
mediatrace schedule 157 life forever start-time now
```

Where to Go Next

For more information about configuring the products in the Medianet product family, see the other chapter in this guide or see the *Cisco Media Monitoring Configuration Guide*.

Additional References

Related Documents

Related Topic	Document Title
Design, configuration, and troubleshooting resources for Cisco Mediatrace and other Cisco Medianet products, including a Quick Start Guide and Deployment Guide.	See the Cisco Medianet Knowledge Base Portal, located at http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html.
IP addressing commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco Media Montoring Command Reference*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified	--

RFCs

RFC[4]	Title
RFC 2205	*RSVP: Resource ReSerVation Protocol* http://www.ietf.org/rfc/rfc2205.txt

[4] These references are only a sample of the many RFCs available on subjects related to IP addressing and IP routing. Refer to the IETF RFC site at http://www.ietf.org/rfc.html for a full list of RFCs.

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

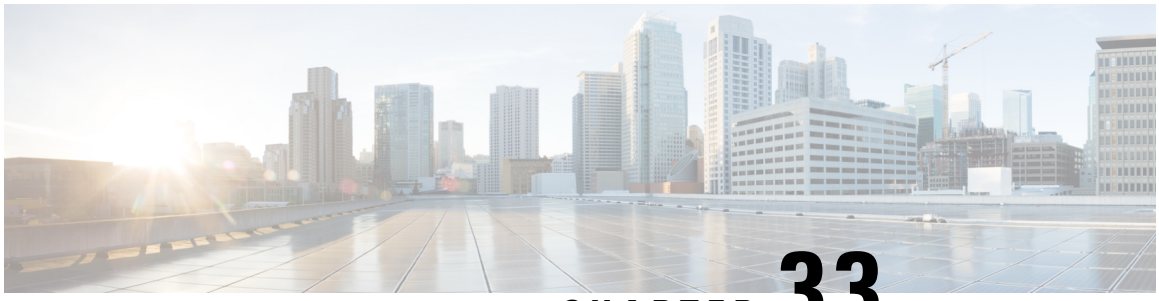
Feature Information for Cisco Mediatrace

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 40: Feature Information for Cisco Mediatrace

Feature Name	Releases	Feature Information
Cisco Mediatrace 1.0	15.1(3)T 12.2(58)SE 15.1(4)M1 15.0(1)SY 15.1(1)SY 15.1(1)SY1 15.2(1)S Cisco IOS XE Release 3.5S 15.1(2)SY	This feature enables you to isolate and troubleshoot network degradation problems for data streams. The following commands were introduced or modified by this feature: **admin-params**, **clear mediatrace**, **incomplete-sessions**, **clock-rate** (RTP parameters), **dest-ip** (flow), **frequency** (session parameters), **history** (session parameters), **ip-protocol** (flow), **max-dropout**, **max-reorder**, **mediatrace**, **mediatrace initiator**, **mediatrace responder**, **mediatrace path-specifier**, **mediatrace poll**, **mediatrace profile perf-monitor**, **mediatrace profile system**, **mediatrace schedule**, **mediatrace session-params**, **metric-list** (monitoring profile), **metric-list** (system profile), **min-sequential**, **path-specifier**, **profile perf-monitor**, **profile system**, **response-timeout** (session parameters), **route-change reaction-time**, **sampling-interval**, **session-params**, **show mediatrace flow-specifier**, **show mediatrace initiator**, **show mediatrace path-specifier**, **show mediatrace profile system**, **show mediatrace profile perf-monitor**, **show mediatrace responder app-health**, **show mediatrace responder sessions**, **show mediatrace session**, **show mediatrace session-params**, **source-ip** (flow), and **source ip** (path).



CHAPTER 33

Configuring Cisco Performance Monitor

This document contains information about and instructions for configuring Cisco Performance Monitor.

Information About Cisco Performance Monitor

Overview of Cisco Performance Monitor

Cisco Performance Monitor enables you to monitor the flow of packets in your network and become aware of any issues that might impact the flow before it starts to significantly impact the performance of the application in question. Performance monitoring is especially important for video traffic because high quality interactive video traffic is highly sensitive to network issues. Even minor issues that may not affect other applications can have dramatic effects on video quality.

Because Cisco Performance Monitor uses similar software components and commands as Cisco NetFlow and Cisco Flexible NetFlow, familiarity with these products will help you to understand how to configure Cisco Performance Monitor. These products provide statistics on packets flowing through a router and are the standard for acquiring IP operational data from IP networks. They provide data to support network and security monitoring, network planning, traffic analysis, and IP accounting. For more information about Cisco NetFlow and Cisco Flexible NetFlow, see the documents listed in the Additional References section.

For more information about the design, configuration, and troubleshooting of Performance Monitor and other Cisco Medianet products, including a Quick Start Guide and Deployment Guide, see the Cisco Medianet Knowledge Base Portal, located at http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html.

Prerequisites for Configuring Cisco Performance Monitor

The following prerequisites must be met before you can configure Cisco Performance Monitor:

IPv4 Traffic

- The networking device must be configured for IPv4 routing.
- One of the following must be enabled on your router and on any interfaces on which you want to enable Cisco Performance Monitor: Cisco Express Forwarding or distributed Cisco Express Forwarding.

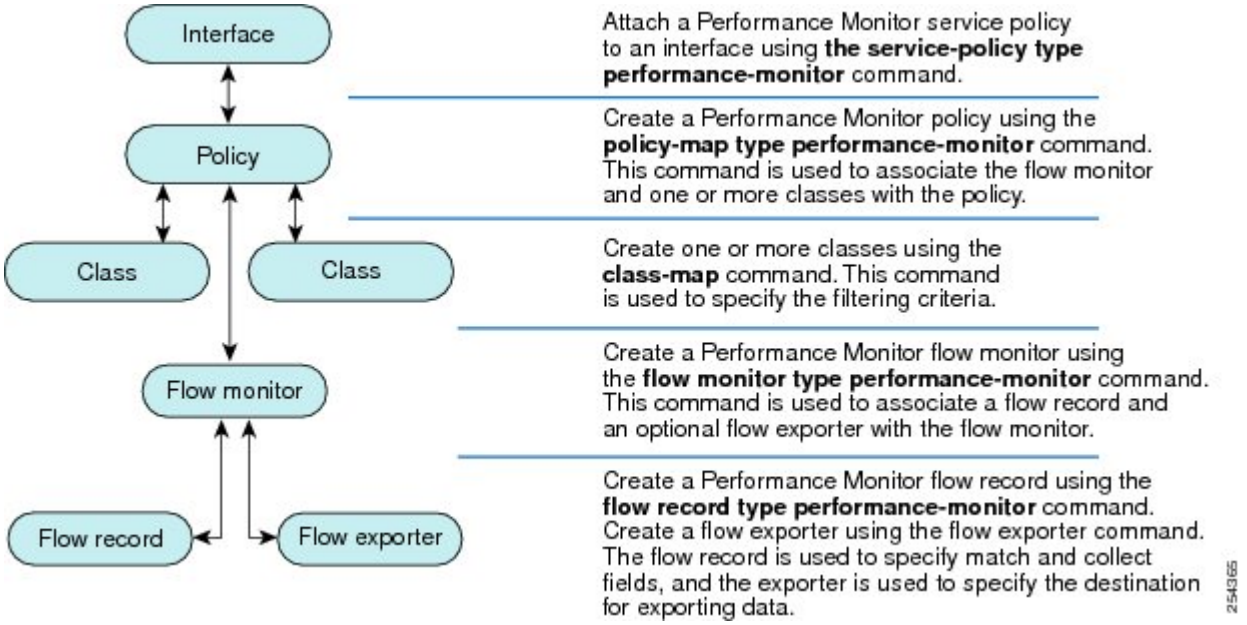
Configuration Components of Cisco Performance Monitor

To configure Cisco Performance Monitor, configure many of the same basic elements that you normally configure for Flexible NetFlow:

- Interface
- Policy
- Class
- Flow monitor
- Flow record
- Flow exporter

The figure below shows how these elements are related to each other. The elements at the bottom of the figure are configured first.

Figure 11: Cisco Performance Monitor Components



As shown above, a policy includes one or more classes. Each class has a flow monitor associated with it, and each flow monitor has a flow record and an optional flow exporter associated with it. These elements are configured in the following order:

1. Configure a flow record to specify the key and non-key fields that you want to monitor. This is configured using **match** and **collect** commands. You can also optimally configure a flow exporter to specify the

export destination. For Cisco Performance Monitor, you must configure a **performance-monitor** type flow record.

2. Configure a flow monitor that includes the flow record and flow exporter. For Cisco Performance Monitor, you must configure a **performance-monitor** type flow monitor.
3. Configure a class to specify the filtering criteria using the **class-map** command.
4. Configure a policy to include one or more classes and one or more **performance-monitor** type flow monitors using the **policy-map** command. For Cisco Performance Monitor, you must configure **performance-monitor** type policies.
5. Associate a **performance-monitor** type policy to the appropriate interface using the **service-policy type performance-monitor** command.

Data That You Can Monitor Using Cisco Performance Monitor

You can monitor the following information by configuring a flow record with **collect** or **match** commands for the corresponding non-key fields:

Tip For more information about these statistics, see the **show performance monitor status**command in the*Cisco Media Monitoring Command Reference.*

- IP Packet Count
- IP TTL
- IP TTL minimum
- IP TTL maximum
- Flow to Interface Mapping
- IP Flow destination address and port, source address and port, and protocol
- RTP Synchronization Source (SSRC)
- IP Octets Count
- Media Stream Packet Count
- Media Stream Octect Count
- Media Byte Rate
- Media Byte Count
- Media Packet Rate
- Media Packet Loss Count
- Media Packet Loss Rate
- Packets Expected Count
- Measured Rate

- Media Loss Event Count
- Round Trip Time (RTT)
- Interarrival Jitter (RFC3550) max
- Interarrival Jitter (RFC3550) min 2
- Interarrival Jitter (RFC3550) mean
- Media Rate Variation
- Monitor Event
- Media Error
- Media Stop
- IP Byte Count
- IP Byte Rate
- IP Source Mask
- IP Destination Mask
- Epoch of A Monitoring Interval
- Packet Forwarding Status
- Packet Drops
- DSCP and IPv6 Traffic Class
- TCP: Maximum Segment Size
- TCP: Window Size Maximum
- TCP: Window Size Maximum
- TCP: Window Size Average
- Out Of Order Bytes
- Out Of Order Packets

SNMP MIB Support for Cisco Performance Monitor

Cisco Performance Monitor provides support for the use of the industry-standard Simple Network Management Protocol (SNMP) to monitor media streams. This support is implemented with the addition of the following Cisco proprietary SNMP Management Information Base (MIB) modules:

- CISCO-FLOW-MONITOR-TC-MIB—Defines the textual conventions common to the following MIB modules.
- CISCO-FLOW-MONITOR-MIB—Defines the framework that describes the flow monitors supported by a system, the flows that it has learned, and the flow metrics collected for those flows.
- CISCO-RTP-METRICS-MIB—Defines objects that describe the quality metrics collected for RTP streams, similar to those described by an RTCP Receiver Report packet (RFC 3550).

- CISCO-IP-CBR-METRICS-MIB—Defines objects that describe the quality metrics collected for IP streams that have a Constant Bit Rate (CBR).

For detailed information about these MIBs, and to locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at http://www.cisco.com/go/mibs .

This feature also includes two new command-line interface (CLI) commands and one modified CLI command. The commands are as follows:

- **snmp-server host**—Enables the delivery of flow monitoring SNMP notifications to a recipient.
- **snmp-server enable traps flowmon**—Enables flow monitoring SNMP notifications. By default, flow monitoring SNMP notifications are disabled.
- **snmp mib flowmon alarm history**—Sets the maximum number of entries maintained by the flow monitor alarm history log.

Limitations for the Catalyst 6500 Platform

Cisco Performance Monitor has the following limitations on the Catalyst 6000 platform:

- There are some limitations on which types of interfaces can be monitored. The next two tables list which types of interfaces are supported for ingress and egress monitoring on the Catalyst 6500 platform.

Table 41: Support for Ingress Interfaces

Interface Type	Support
Layer 3 Routed Port	Yes
Layer 3 Sub-interface (a)	No
Layer 3 port channels	Yes
Layer 3 port-channel sub-interface (a)	No
Layer 3 SVI (b)	Partial (see the third bullet below)
L3 Tunnels	No
Layer 2 Physical (Switched) Ports	Yes
Layer 2 Port-channels	Yes
Layer 2 Vlans	Yes

Table 42: Support for Egress Interfaces

Interface Type	Support
Layer 3 Routed Port	Yes
Layer 3 Sub-interface (a)	Yes

Interface Type	Support
Layer 3 port channels	Yes
Layer 3 port-channel sub-interface (a)	Yes
Layer 3 SVI (b)	Yes
L3 Tunnels	No
Layer 2 Physical (Switched) Ports	No
Layer 2 Port-channels	No
Layer 2 Vlans	Yes

- Performance monitoring on VRFs is not supported.
- Performance Monitoring of multicast flows is not supported.
- Routed traffic from a trunk port on a VLAN interface cannot not be monitored because it is not possible to identify the source VLAN interface for the traffic. You will see the following syslog message: “Routed traffic from trunk ports will not be monitored by ingress policy on VLAN interface.”

  For a workaround, you can configure a performance monitoring policy on a trunk interface. This monitoring will result in additional CPU usage.
- You cannot use match all type Class maps. Only match any type of lookups are supported. If you configure performance monitoring to use match-all type class maps, it will result in the cloning of packet to the CPU. Packets will then again be classified in the CPU when match-all classes are properly applied and packet are dropped if required. This causes higher than expected CPU usage.
- Performance monitoring policy on the egress of a VLAN interface will not monitor traffic getting bridged within the VLAN. This is due to hardware limitation. Workaround is to apply the policy at the ingress of VLAN interface as well as egress. Policy on the ingress of the VLAN interface will monitor bridged packets.
- Cloned packets from Egress policies can only be software rate-limited. No hardware-based protection is available for these packets. Therefore, you might see high interrupt CPU usage during scenarios when many flows are being monitored.
- Egress performance monitoring makes use of a recirculation mechanism on the Catalyst 6500 platform. This introduces several microseconds of additional latency to the frame switching.
- Performance monitoring is not supported for the packets switched using the Fast (CEF) Path.
- Lawful intercept and performance monitoring makes use of the same mechanism for cloning the packets. The Lawful Intercept feature takes precedence over performance monitoring. Therefore, performance monitoring does not function when the Lawful Intercept feature is enabled. When this occurs, a syslog message is created.
- Performance monitoring makes use of same mechanism as other features, such as Optimized ACL logging, VACL Capture, IPv6 Copy, and so on. The feature that is enabled first takes precedence. The other features are blocked from being configured and a syslog message is created.

Limitations for IPv6 Support

Support for IPv6 with Performance Monitor has the following limitations:

- The following topologies are supported with IPv6: Non-MPLS, DMVPN (on most platforms), and dual stack.
- The following topologies are not supported with IPv6: MPLS/VRF (6PE and 6VPE), GETVPN and IPV6 over IPV4 tunnel.
- Mediatrace does not support IPv6.
- Exporting data to a IPv6 address is not supported on the ASR1K platform.
- Flexible NetFlow does not support IPv6 multicast.
- DMVPN is not supported with IPv6 on the ASR1K platform.

How to Configure Troubleshoot and Maintain Cisco Performance Monitor

Note Many of the Flexible NetFlow commands, keywords, and arguments used in used in these tasks are available in previous releases. For more information about these existing Flexible NetFlow commands, keywords, and arguments, refer to the *Cisco IOS Flexible NetFlow Command Reference*.

Configuring a Flow Exporter for Cisco Performance Monitor

Flow exporters are used to send the data that you collect with Cisco Performance Monitor to a remote system such as a NetFlow Collection Engine. Flow exporters use user datagram protocol (UDP) as the transport protocol and use the Version 9 export format.

To configure a flow exporter for the flow monitor, in order to export the data that is collected by Cisco Performance Monitor to a remote system for further analysis and storage, perform the following optional task. For Cisco Performance Monitor, flow exporters are configured the same way as they are configured for Cisco IOS Flexible NetFlow. For more information. see *Configuring Data Export for Cisco IOS Flexible NetFlow with Flow Exporters*.

Note You can export to a destination using either an IPv4 or IPv6 address.

Note Each flow exporter supports only one destination. If you want to export the data to multiple destinations, you must configure multiple flow exporters and assign them to the flow monitor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow exporter** *exporter-name*
4. **description** *description*
5. **destination** {*ip-address* | *hostname*} [**vrf** *vrf-name*]
6. **export-protocol** {**netflow-v5** | **netflow-v9** | **ipfix** }
7. **dscp** *dscp*
8. **source** *interface-type interface-number*
9. **option** {**application-attributes** | **application table** | **exporter-stats** | **interface-table** | **metadata-table** | **sampler-table** | **vrf-table**} [**timeout** *seconds*]
10. **output-features**
11. **template data timeout** *seconds*
12. **transport udp** *udp-port*
13. **ttl** *seconds*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow exporter** *exporter-name* **Example:** `Device(config)# flow exporter EXPORTER-1`	Creates the flow exporter and enters Flexible NetFlow flow exporter configuration mode. • This command also allows you to modify an existing flow exporter.
Step 4	**description** *description* **Example:** `Device(config-flow-exporter)# description Exports to the datacenter`	(Optional) Configures a description to the exporter that will appear in the configuration and the display of the **show flow exporter** command.
Step 5	**destination** {*ip-address*	*hostname*} [**vrf** *vrf-name*] **Example:** `Device(config-flow-exporter)# destination 172.16.10.2`	Specifies the IP address or hostname of the system to which the exporter sends data. **Note** You can export to a destination using either an IPv4 or IPv6 address.

	Command or Action	Purpose
Step 6	**export-protocol** {**netflow-v5** | **netflow-v9** | **ipfix** } **Example:** `Device(config-flow-exporter)# export-protocol netflow-v9`	Specifies the protocol used by the exporter. **Note** The export of extracted fields from NBAR is only supported over IPFIX.
Step 7	**dscp** *dscp* **Example:** `Device(config-flow-exporter)# dscp 63`	(Optional) Configures differentiated services code point (DSCP) parameters for datagrams sent by the exporter. • The range for the *dscp* argument is from 0 to 63. Default: 0.
Step 8	**source** *interface-type interface-number* **Example:** `Device(config-flow-exporter)# source ethernet 0/0`	(Optional) Specifies the local interface from which the exporter will use the IP address as the source IP address for exported datagrams.
Step 9	**option** {**application-attributes** | **application table** | **exporter-stats** | **interface-table** | **metadata-table** | **sampler-table** | **vrf-table**} [**timeout** *seconds*] **Example:** `Device(config-flow-exporter)# option exporter-stats timeout 120`	(Optional) Enables the use of option tables to decrease the amount of data exported. These tables allow the exporter to just export an ID that represents the complete value of the metadata and is mapped to the value by the option table. For example, the interface table maps the SNMP index to the interface name and the VRF table maps the VRF ID to the name. • You can enable the use of any combination of option tables concurrently. • The range for the *seconds* argument is 1 to 86,400. Default: 600.
Step 10	**output-features** **Example:** `Device(config-flow-exporter)# output-features`	(Optional) Enables sending export packets using quality of service (QoS) and encryption.
Step 11	**template data timeout** *seconds* **Example:** `Device(config-flow-exporter)# template data timeout 120`	(Optional) Configure the resending of templates based on a timeout. • The range for the *seconds* argument is 1 to 86400 (86400 seconds = 24 hours).
Step 12	**transport udp** *udp-port* **Example:** `Device(config-flow-exporter)# transport udp 650`	Configures UDP as the transport protocol and specifies the UDP port on which the destination system is listening for exported datagrams. • The range for the *udp-port* argument is from 1 to 65536.

	Command or Action	Purpose
Step 13	**ttl** *seconds* **Example:** `Device(config-flow-exporter)# ttl 15`	(Optional) Configures the time-to-live (TTL) value for datagrams sent by the exporter. • The range for the *seconds* argument is from 1 to 255.
Step 14	**end** **Example:** `Device(config-flow-exporter)# end`	Exits flow exporter configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow exporter, use the **show flow exporter** command.

Configuring a Flow Record for Cisco Performance Monitor

The basic concepts and techniques for configuring a flow record for Cisco Performance Monitor are the same as flow records for Flexible NetFlow. The flow record specifies how the data collected data is aggregated and presented. The only significant difference is that, for Cisco Performance Monitor, the command includes **type performance-monitor**.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow record type performance-monitor** *record-name*
4. **match application** {**name** [**account-on-resolution**] | **vendor** | **version**}
5. **match connection transaction-id**
6. **match flow** {**direction** | **sampler**}
7. **match interface** {**input** | **output**}
8. **match ipv4** {**destination**{**address** | **prefix** [**minimum-mask** *mask*]} | **protocol** | **source** {**address** | **prefix** [**minimum-mask** *mask*]}
9. **match ipv4 fragmentation** {**flags** |**offset**}
10. **match ipv4** {**section** {**header size** *header-size* | **payload size** *payload-size*}
11. **match ipv4 total-length**
12. **match ipv4 ttl**
13. **match ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**}
14. **match ipv6 destination** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]}
15. **match ipv6 extension map**
16. **match ipv6 fragmentation** {**flags** | **id** | **offset**}
17. **match ipv6 hop-limit**
18. **match ipv6 length** {**header** | **payload** | **total**}
19. **match ipv6** {**section** {**header size** *header-size* | **payload size** *payload-size*}
20. **match ipv6 source** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]}

21. **match metadata** {**global-session-id** | **multi-party-session-id**}
22. **match routing** {**destination** | **source**}
23. **match routing is-multicast**
24. **match routing multicast replication-factor**
25. **match transport** {**destination-port** | **igmp** | **rtp** [**ssrc**] | **source-port**}
26. **match transport icmp ipv4** {**code** | **type**}
27. **match transport icmp ipv6** {**code** | **type**}
28. **match transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**}
29. **match transport udp** {**destination-port** | **message-length** | **source-port**}
30. **collect application media** {**bytes**{**rate** | **counter**}| **packets** {**rate**|**counter**} | **events**}
31. **collect application** {**name** [**account-on-resolution**]| **description** | **http host** | **nntp group-name** | **pop3 server** | **rstp host-name** | **sip** {**destination** | **source**} | **smtp** {**sender** | **server**} | **vendor** | **version**}
32. **collect connection**
33. **collect counter** {**bytes** [**long** | **rate**] |**packets**[**dropped** [**long**] | **long**]}
34. **collect datalink mac source address** {**input** | **output**}
35. **collect flow direction**
36. **collect interface** {**input** | **output**}
37. **collect ipv4** {**destination mask** [**minimum-mask** *mask*]} | **dscp** | **source mask** [**minimum-mask** *mask*] | **ttl** [**minimum** | **maximum**]}
38. **collect ipv4 fragmentation** {**flags** | **offset**}
39. **collect ipv4** {**section** {**header size** *header-size* | **prefix**[**payload size** *payload-size*}
40. **collect ipv4 total-length** [**maximum** | **minimum**]
41. **collect ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**}
42. **collect ipv6 destination** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]}
43. **collect ipv6 extension-map**
44. **collect ipv6 fragmentation** {**flags** | **offset**}
45. **collect ipv6 hop-limit** [**maximum**] [**minimum**]
46. **collect ipv6 length**{**header** | **payload** | **total** [**maximum**] [**minimum**] }
47. **collect ipv6** {**section** {**header size** *header-size* | **prefix** [**payload size** *payload-size*}
48. **collect ipv6 source** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]}
49. **collect metadata** {**global-session-id** | **multi-party-session-id**}
50. **collect monitor event**
51. **collect routing forwarding-status** [**reason**]
52. **collect routing is-multicast**
53. **collect routing multicast replication-factor**
54. **collect timestamp internal**
55. **collect timestamp sys-uptime** {**first** | **last**}
56. **collect transport** {**destination-port** | **igmp type** | **source-port** | **event packet-loss counter** | **packets** {**expected counter** | **lost** {**counter** | **rate**} | **out-of-order**} | **round-trip-time** | **rtp jitter** {**minimum** | **mean** | **maximum**}}
57. **collect transport icmp ipv4**
58. **collect transport icmp ipv6**

59. **collect transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**}
60. **collect transport udp** {**destination-port** | **message-length** | **source-port**}
61. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow record type performance-monitor** *record-name* **Example:** `Device(config)# flow record type performance-monitor record-8`	Creates a flow record and enters flow record configuration mode. • This command also allows you to modify an existing flow record.
Step 4	**match application** {**name** [**account-on-resolution**]	**vendor**	**version**} **Example:** `Device(config-flow-record)# match application name`	Specifies that the application name, vendor, or version will be used as a key field.
Step 5	**match connection transaction-id** **Example:** `Device(config-flow-record)# match connection transaction-id`	Specifies that the application name will be used as a key field.
Step 6	**match flow** {**direction**	**sampler**} **Example:** `Device(config-flow-record)# match flow direction`	Specifies that the flow direction field will be used as a key field.
Step 7	**match interface** {**input**	**output**} **Example:** `Device(config-flow-record)# match flow direction`	Specifies that the input interface field will be used as a key field.

	Command or Action	Purpose
Step 8	**match ipv4** {**destination**{**address** | **prefix** [**minimum-mask** *mask*]} | **protocol** | **source** {**address** | **prefix** [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv4 destination address`	Specifies that one or more of the IPv4 fields will be used as a key field.
Step 9	**match ipv4 fragmentation** {**flags** |**offset**} **Example:** `Device(config-flow-record)# match ipv4 fragmentation flags`	Specifies that one or more of the IPv4 fields will be used as a key field.
Step 10	**match ipv4** {**section** {**header size** *header-size* | **payload size** *payload-size*} **Example:** `Device(config-flow-record)# match ipv4 section header size 8`	Specifies that one or more of the IPv4 fields will be used as a key field.
Step 11	**match ipv4 total-length** **Example:** `Device(config-flow-record)# match ipv4 total-length`	Specifies that the IPv4 total length field will be used as a key field.
Step 12	**match ipv4 ttl** **Example:** `Device(config-flow-record)# match ipv4 ttl`	Specifies that the IPv4 ttl field will be used as a key field.
Step 13	**match ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**} **Example:** `Device(config-flow-record)# match ipv6 dscp`	Specifies that the IPv6 DSCP field will be used as a key field.
Step 14	**match ipv6 destination** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv4 destination address`	Specifies that the IPv6 destination address field will be used as a key field.
Step 15	**match ipv6 extension map** **Example:**	Specifies that the IPv6 extension map field will be used as a key field.

	Command or Action	Purpose
	`Device(config-flow-record)# match ipv6 extension map`	
Step 16	**match ipv6 fragmentation** {**flags** | **id** | **offset**} **Example:** `Device(config-flow-record)# match ipv6 fragmentation flags`	Specifies that the IPv6 fragmentation flags field will be used as a key field.
Step 17	**match ipv6 hop-limit** **Example:** `Device(config-flow-record)# match ipv6 hop-limit`	Specifies that the IPv6 hop limit field will be used as a key field.
Step 18	**match ipv6 length** {**header** | **payload** | **total**} **Example:** `Device(config-flow-record)# match ipv6 length total`	Specifies that the IPv6 total length field will be used as a key field.
Step 19	**match ipv6** {**section** {**header size** *header-size* | **payload size** *payload-size*} **Example:** `Device(config-flow-record)# match ipv6 section header size 8`	Specifies that the IPv6 section header size field will be used as a key field.
Step 20	**match ipv6 source** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv6 source address`	Specifies that the IPv6 source address field will be used as a key field.
Step 21	**match metadata** {**global-session-id** | **multi-party-session-id**} **Example:** `Device(config-flow-record)# match metadata global-session-id`	Specifies that a metadata session ID field will be used as a key field.
Step 22	**match routing** {**destination** | **source**} **Example:** `Device(config-flow-record)# match routing source`	Specifies that the routing source flag field will be used as a key field.
Step 23	**match routing is-multicast** **Example:**	Specifies that the routing is-multicast flag field will be used as a key field.

	Command or Action	Purpose
	`Device(config-flow-record)# match routing is-multicast`	
Step 24	**match routing multicast replication-factor** **Example:** `Device(config-flow-record)# match routing multicast replication-factor`	Specifies that the routing multicast replication-factor flag field will be used as a key field.
Step 25	**match transport** {**destination-port** | **igmp** | **rtp** [**ssrc**] | **source-port**} **Example:** `Device(config-flow-record)# match transport destination-port`	Specifies that one or more of the transport layer fields will be used as a key field, including the Synchronization Source (SSRC) field in the Real-Time Transport Protocol (RTP) packet header.
Step 26	**match transport icmp ipv4** {**code** | **type**} **Example:** `Device(config-flow-record)# match transport icmp ipv4 code`	Specifies that the IPv4 ICMP transport code field will be used as a key field.
Step 27	**match transport icmp ipv6** {**code** | **type**} **Example:** `Device(config-flow-record)# match transport icmp ipv6 code`	Specifies that the IPv6 ICMP transport code field will be used as a key field.
Step 28	**match transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**} **Example:** `Device(config-flow-record)# match transport tcp destination-port`	Specifies that the IPv6 TCP transport destination port field will be used as a key field.
Step 29	**match transport udp** {**destination-port** | **message-length** | **source-port**} **Example:** `Device(config-flow-record)# match transport udp destination-port`	Specifies that the IPv6 UDP transport destination port field will be used as a key field.
Step 30	**collect application media** {**bytes**{**rate** | **counter**}| **packets** {**rate**|**counter**} | **events**} **Example:**	Specifies that the application media bytes, packets, or events will be used as a nonkey field. An application event occurs when either one of the thresholds specified by a

	Command or Action	Purpose
	`Device(config-flow-record)# collect application media events`	react statement for the flow was crossed at least once in the monitoring interval or no media packets were seen.
Step 31	**collect application** {**name** [**account-on-resolution**]| **description** | **http host** | **nntp group-name** | **pop3 server** | **rstp host-name** | **sip** {**destination** | **source**} | **smtp** {**sender** | **server**} | **vendor** | **version**} **Example:** `Device(config-flow-record)# collect application name`	Specifies that the application name will be used as a nonkey field.
Step 32	**collect connection** **Example:** `Device(config-flow-record)# collect connection initiator`	Specifies that the connection initiator will be used as a nonkey field.
Step 33	**collect counter** {**bytes** [**long** | **rate**] |**packets**[**dropped** [**long**] | **long**]} **Example:** `Device(config-flow-record)# collect counter bytes long`	Specifies the number of bytes or packets that will be used as a nonkey field.
Step 34	**collect datalink mac source address** {**input** | **output**} **Example:** `Device(config-flow-record)# collect flow direction`	Specifies that the flow direction field will be used as a nonkey field.
Step 35	**collect flow direction** **Example:** `Device(config-flow-record)# collect flow direction`	Specifies that the flow direction field will be used as a nonkey field.
Step 36	**collect interface** {**input** | **output**} **Example:** `Device(config-flow-record)# collect interface input`	Specifies that the input or output interface will be used as a nonkey field.
Step 37	**collect ipv4** {**destination mask** [**minimum-mask** *mask*]} | **dscp** | **source mask** [**minimum-mask** *mask*] | **ttl** [**minimum** | **maximum**]} **Example:** `Device(config-flow-record)# collect ipv4 dscp`	Specifies that the IPv4 DSCP field will be used as a nonkey field.

	Command or Action	Purpose
Step 38	**collect ipv4 fragmentation** {**flags** | **offset**} **Example:** `Device(config-flow-record)# collect ipv4 fragmentation flags`	Specifies that the IPv4 fragmentation flags field will be used as a nonkey field.
Step 39	**collect ipv4** {**section** {**header size** *header-size* | **prefix**[**payload size** *payload-size*} **Example:** `Device(config-flow-record)# collect ipv4 section header size 8`	Specifies that the IPv4 section header size field will be used as a nonkey field.
Step 40	**collect ipv4 total-length** [**maximum** | **minimum**] **Example:** `Device(config-flow-record)# collect ipv4 total-length`	Specifies that the IPv4 total-length field will be used as a nonkey field.
Step 41	**collect ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**} **Example:** `Device(config-flow-record)# collect ipv6 dscp`	Specifies that the IPv6 DSCP field will be used as a nonkey field.
Step 42	**collect ipv6 destination** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# collect ipv6 destination mask`	Specifies that the IPv6 destination mask field will be used as a nonkey field.
Step 43	**collect ipv6 extension-map** **Example:** `Device(config-flow-record)# collect ipv6 extension-map`	Specifies that the IPv6 extension-map field will be used as a nonkey field.
Step 44	**collect ipv6 fragmentation** {**flags** | **offset**} **Example:** `Device(config-flow-record)# collect ipv6 fragmentation flags`	Specifies that the IPv6 fragmentation flags field will be used as a nonkey field.
Step 45	**collect ipv6 hop-limit** [**maximum**] [**minimum**] **Example:** `Device(config-flow-record)# collect ipv6 hop-limit`	Specifies that the IPv6 hop-limit field will be used as a nonkey field.

	Command or Action	Purpose
Step 46	**collect ipv6 length** {**header** | **payload** | **total** [**maximum**] [**minimum**] } **Example:** `Device(config-flow-record)# collect ipv6 length total`	Specifies that the IPv6 total length field will be used as a nonkey field.
Step 47	**collect ipv6** {**section** {**header size** *header-size* | **prefix** [**payload size** *payload-size*} **Example:** `Device(config-flow-record)# collect ipv6 section header size 8`	Specifies that the IPv6 section header size field will be used as a nonkey field.
Step 48	**collect ipv6 source** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# collect ipv6 source mask`	Specifies that the IPv6 source mask field will be used as a nonkey field.
Step 49	**collect metadata** {**global-session-id** | **multi-party-session-id**} **Example:** `Device(config-flow-record)# collect meatdata global-session-id`	Specifies that a metadata session ID field will be used as a nonkey field.
Step 50	**collect monitor event** **Example:** `Device(config-flow-record)# collect monitor event`	Specifies that the monitor event field will be used as a nonkey field. A monitor event occurs when no media application packets were seen
Step 51	**collect routing forwarding-status** [**reason**] **Example:** `Device(config-flow-record)# collect routing forwarding-status`	Specifies that the one or more of the routing attributes will be used as a nonkey field.
Step 52	**collect routing is-multicast** **Example:** `Device(config-flow-record)# collect routing is-multicast`	Specifies that the routing is-multicast field will be used as a nonkey field.
Step 53	**collect routing multicast replication-factor** **Example:**	Specifies that the routing multicast replication-factor field will be used as a nonkey field.

	Command or Action	Purpose
	`Device(config-flow-record)# collect routing multicast replication-factor`	
Step 54	**collect timestamp internal** **Example:** `Device(config-flow-record)# collect timestamp internal`	Specifies that the system timestamp of the first seen or last seen packet in a flow will be used as a nonkey field.
Step 55	**collect timestamp sys-uptime** {**first** | **last**} **Example:** `Device(config-flow-record)# collect timestamp sys-uptime`	Specifies that the system timestamp of the sys-uptime will be used as a nonkey field.
Step 56	**collect transport** {**destination-port** | **igmp type** | **source-port** | **event packet-loss counter** | **packets** {**expected counter** | **lost** {**counter** | **rate**} | **out-of-order**} | **round-trip-time** | **rtp jitter** {**minimum** | **mean** | **maximum**}} **Example:** `Device(config-flow-record)# collect transport packets expected counter`	Specifies that one or more of the transport layer fields will be used as a nonkey field. These fields include metrics for: • Packet-loss counter • Expected packets counter • Jitter
Step 57	**collect transport icmp ipv4** **Example:** `Device(config-flow-record)# collect transport icmp ipv4`	Specifies that the transport ICMP IPv4 field will be used as a nonkey field.
Step 58	**collect transport icmp ipv6** **Example:** `Device(config-flow-record)# collect transport icmp ipv6`	Specifies that the transport ICMP IPv6 field will be used as a nonkey field.
Step 59	**collect transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**} **Example:** `Device(config-flow-record)# collect transport tcp destination-port`	

	Command or Action	Purpose
Step 60	**collect transport udp** {**destination-port** | **message-length** | **source-port**} **Example:** `Device(config-flow-record)# collect transport udp destination-port`	Specifies that the transport UDP destination port field will be used as a nonkey field.
Step 61	**end** **Example:** `Device(config-flow-record)# end`	Exits flow record configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow record, use the **show flow record type performance-monitor** command.

Configuring a Usage Record for AVC Phase 2

To configure an input usage record, perform the following required task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. flow record flow-record-name
4. match interface input
5. match flow direction
6. match connection client {ipv4 | ipv6} address
7. match connection client transport port
8. match connection server {ipv4 | ipv6} address
9. match connection server transport port
10. match ipv4 {initiator | responder} address
11. match ipv6 {initiator | responder} address
12. match transport {initiator | responder} port
13. match routing vrf {input | output}
14. match datalink {destination-vlan-id | source-vlan-id}
15. match datalink vlan {input | output}
16. match datalink mac {destination | source} address {input | output}
17. match flow {class | qos-class}
18. match policy performance-monitor classification hierarchy
19. match services waas segment
20. collect interface output
21. collect flow direction
22. collect timestamp sys-uptime first

23. collect timestamp sys-uptime last
24. collect counter bytes long
25. collect counter packets
26. collect connection client {ipv4 | ipv6} address
27. collect connection client counter {bytes long | packets long | packets retransmitted}
28. collect connection client transport port
29. collect connection new-connections
30. collect connection sum-duration
31. collect routing vrf {input | output}
32. collect connection delay application {sum | min | max}
33. collect connection delay network {client-to-server | to-server [histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7}] {sum | min | max}
34. collect connection delay response {client-to-server | to-client | to-server} {sum | min | max}
35. collect connection performance application-delay {sum | min | max}
36. collect connection performance initiator bytes long
37. collect connection performance initiator count re-transmitted-packets
38. collect connection performance initiator network-delay {sum | min | max}
39. collect connection performance initiator packets long
40. collect connection performance network-delay {sum | min | max}
41. collect connection performance new-transaction-time
42. collect connection performance total-transaction-time {sum | min | max}
43. collect connection performance total-transaction-time {sum | min | max}
44. collect connection performance responder bytes long
45. collect connection performance responder response-time {sum | min | max}
46. collect connection performance responder network-delay {sum | min | max}
47. collect connection performance responder count {histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7} | late-responses | responses}
48. collect connection performance responder packets long
49. collect connection performance total-delay {sum | min | max}
50. collect connection performance total-transaction-time {sum | min | max}
51. collect connection server {ipv4 | ipv6} address
52. collect connection server counter {bytes long | packets long | packets retransmitted}
53. collect connection server transport port
54. collect connection transaction {counter complete | duration {sum | min | max}}
55. collect datalink {destination-vlan-id | source-vlan-id}
56. collect datalink mac {destination | source} address {input | output}
57. collect datalink vlan {input | output}
58. collect policy performance-monitor classification hierarchy
59. collect services waas {passthrough-reason | segment}
60. collect timestamp absolute {first | last}
61. collect transport tcp {option map | window-size {sum | minimum | maximum} | maximum-segment-size}
62. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	flow record flow-record-name **Example:** `Router(config)# flow record my-input-usage-monitor`	Creates a flow record and enters flow record configuration mode.
Step 4	match interface input **Example:** `Router(config-flow-record)# match interface input`	Configures the input interface for the packet as a key field for the flow record. input—Traffic arrives on the Cisco router's input interface.
Step 5	match flow direction **Example:** `Router(config-flow-record)# match flow direction`	Configures the direction of the flow record as a key field. The direction is either input or output.
Step 6	match connection client {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# match connection client ipv6 address`	Configures the Ipv6 address of the client as a key field for a flow record.
Step 7	match connection client transport port **Example:** `Router(config-flow-record)# match connection client transport port`	Configures the connection port of the client as a key field for a flow record.
Step 8	match connection server {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# match connection server ipv6 address`	Configures the Ipv6 address of the server as a key field for a flow record.
Step 9	match connection server transport port **Example:**	Configures the connection port of the server as a key field for a flow record.

	Command or Action	Purpose
	`Router(config-flow-record)# match connection server transport port`	
Step 10	match ipv4 {initiator | responder} address **Example:** `Router(config-flow-record)# match ipv4 initiator address`	(Optional) For IPv4 networks, configures the IPv4 address of the initiator or responder as a key field. The direction is either input or output.
Step 11	match ipv6 {initiator | responder} address **Example:** `Router(config-flow-record)# match ipv6 initiator address`	(Optional) For IPv6 networks, configures the IPv6 address of the initiator or responder as a key field. The direction is either input or output.
Step 12	match transport {initiator | responder} port **Example:** `Router(config-flow-record)# match transport initiator port`	(Optional) Configures the transport port of the initiator or responder as a key field.
Step 13	match routing vrf {input | output} **Example:** `Router(config-flow-record)# match routing vrf input`	(Optional) Configures the virtual routing and forwarding (VRF) ID for incoming or outgoing packets as a key field.
Step 14	match datalink {destination-vlan-id | source-vlan-id} **Example:** `Router(config-flow-record)# match datalink destination-vlan-id`	(Optional) Configures the destination VLAN ID as a key field.
Step 15	match datalink vlan {input | output} **Example:** `Router(config-flow-record)# match datalink vlan input`	(Optional) Configures the VLAN ID for incoming or outgoing packets as a key field.
Step 16	match datalink mac {destination | source} address {input | output} **Example:** `Router(config-flow-record)# match datalink mac destination address output`	(Optional) Configures the destination MAC address as a key field.
Step 17	match flow {class | qos-class} **Example:**	Configures the use of the class ID as a key field for a flow record.

	Command or Action	Purpose
	`Router(config-flow-record)# match flow class`	
Step 18	match policy performance-monitor classification hierarchy **Example:** `Router(config-flow-record)# match policy performance-monitor classification hierarchy`	Configures the use of the Performance Monitor policy classification hierarchy as a key field for a flow record.
Step 19	match services waas segment **Example:** `Router(config-flow-record)# match services waas segment`	Configures the use of the WAAS segment as a key field for a flow record.
Step 20	collect interface output **Example:** `Router(config-flow-record)# collect interface output`	Configures the output interface as a non-key field for a flow record and enables collecting the output interface fields from the flows for the flow record.
Step 21	collect flow direction **Example:** `Router(config-flow-record)# collect flow direction`	Configures the flow direction as a non-key field for a flow record.
Step 22	collect timestamp sys-uptime first **Example:** `Router(config-flow-record)# collect timestamp sys-uptime first`	Configures the system uptime of the first seen packet in a flow as a nonkey field for a flow record. • first—Configures the system uptime for the time the first packet was seen from the flows as a nonkey field and enables collecting time stamps based on the system uptime for the time the first packet was seen from the flows.
Step 23	collect timestamp sys-uptime last **Example:** `Router(config-flow-record)# collect timestamp sys-uptime last`	Configures the system uptime of the last seen packet in a flow as a nonkey field for a flow record. • last—Configures the system uptime for the time the last packet was seen from the flows as a nonkey field and enables collecting time stamps based on the system uptime for the time the most recent packet was seen from the flows.
Step 24	collect counter bytes long **Example:** `Router(config-flow-record)# collect counter bytes long`	Configures the number of bytes in a flow as a nonkey field for a flow record. • bytes—Configures the number of bytes seen in a flow as a nonkey field and enables collecting the total number of bytes from the flow.

	Command or Action	Purpose
		• long—Enables collecting the total number of bytes or packets from the flow by using a 64-bit counter rather than a 32-bit counter.
Step 25	collect counter packets **Example:** `Router(config-flow-record)# collect counter packets`	Configures the number of packets in a flow as a nonkey field for a flow record. • packets—Configures the number of packets seen in a flow as a nonkey field and enables collecting the total number of packets from the flow.
Step 26	collect connection client {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# collect connection client ipv6 address`	Configures the Ipv6 address of the client as a nonkey field for a flow record.
Step 27	collect connection client counter {bytes long | packets long | packets retransmitted} **Example:** `Router(config-flow-record)# collect connection client counter packets retransmitted`	Configures the number of the client packets retransmitted as a nonkey field for a flow record.
Step 28	collect connection client transport port **Example:** `Router(config-flow-record)# collect connection client transport port`	Configures the client connection port as a nonkey field for a flow record.
Step 29	collect connection new-connections **Example:** `Router(config-flow-record)# collect connection new-connections`	Counts the number of TCP or UDP connections which were opened during the observation period. The observation period may be specified by the flow start and end timestamps.
Step 30	collect connection sum-duration **Example:** `Router(config-flow-record)# collect connection sum-duration`	Aggregates the total time, in seconds, for all the TCP or UDP connections, which were in use during the observation period. For example, if there are five concurrent connections each for 10 seconds, the value would be 50 seconds.
Step 31	collect routing vrf {input | output} **Example:** `Router(config-flow-record)# collect routing vrf output`	Configures the virtual routing and forwarding (VRF) ID for incoming or outgoing packets output as a nonkey field for a flow record.

	Command or Action	Purpose
Step 32	collect connection delay application {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay application sum`	Configures the total amount of application delay as a nonkey field for a flow record.
Step 33	collect connection delay network {client-to-server | to-server [histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7}] {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay network client-to-server sum`	Configures the total amount of network delay between the client and the server as a nonkey field for a flow record.
Step 34	collect connection delay response {client-to-server | to-client | to-server} {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay response client-to-server sum`	Configures the total amount of response delay between the client and the server as a nonkey field for a flow record.
Step 35	collect connection performance application-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance application-delay sum`	Configures the total application delay as a nonkey field for a flow record.
Step 36	collect connection performance initiator bytes long **Example:** `Router(config-flow-record)# collect connection performance initiator bytes long`	Configures the number of long bytes for the Mediatrace initiator as a nonkey field for a flow record.
Step 37	collect connection performance initiator count re-transmitted-packets **Example:** `Router(config-flow-record)# collect connection performance initiator count re-transmitted-packets`	Configures the number of retrransmitted packets for the Mediatrace initiator as a nonkey field for a flow record.
Step 38	collect connection performance initiator network-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance initiator network-delay sum`	Configures the total network delay for the Mediatrace initiator as a nonkey field for a flow record.

	Command or Action	Purpose
Step 39	collect connection performance initiator packets long **Example:** `Router(config-flow-record)# collect connection performance initiator packets long`	Configures the number of long packets for the Mediatrace initiator as a nonkey field for a flow record.
Step 40	collect connection performance network-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance network-delay sum`	Configures the total network delay as a nonkey field for a flow record.
Step 41	collect connection performance new-transaction-time **Example:** `Router(config-flow-record)# collect connection performance new-transaction`	Configures the new transaction field as a nonkey field for a flow record.
Step 42	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	Configures the total transaction time as a nonkey field for a flow record.
Step 43	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	Configures the total transaction time as a nonkey field for a flow record.
Step 44	collect connection performance responder bytes long **Example:** `Router(config-flow-record)# collect connection performance responder bytes long`	Configures the number of long bytes for the Mediatrace responder as a nonkey field for a flow record.
Step 45	collect connection performance responder response-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance responder response-time sum`	Configures the total response time for the Mediatrace responder as a nonkey field for a flow record.
Step 46	collect connection performance responder network-delay {sum | min | max} **Example:**	Configures the total network delay for the Mediatrace responder as a nonkey field for a flow record.

	Command or Action	Purpose
	`Router(config-flow-record)# collect connection performance responder network-delay sum`	
Step 47	collect connection performance responder count {histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7} | late-responses | responses} **Example:** `Router(config-flow-record)# collect connection performance responder count late-responses`	Configures the number of late responses for the Mediatrace responder as a nonkey field for a flow record.
Step 48	collect connection performance responder packets long **Example:** `Router(config-flow-record)# collect connection performance responder packets long`	Configures the number of long packets for the Mediatrace responder as a nonkey field for a flow record.
Step 49	collect connection performance total-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-delay sum`	Configures the total connection delay as a nonkey field for a flow record.
Step 50	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	Configures the total transaction time as a nonkey field for a flow record.
Step 51	collect connection server {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# collect connection server ipv6 address`	Configures the IPv6 address of the server as a nonkey field for a flow record.
Step 52	collect connection server counter {bytes long | packets long | packets retransmitted} **Example:** `Router(config-flow-record)# collect connection server counter packets retransmitted`	Configures the number of the server packets retransmitted as a nonkey field for a flow record.
Step 53	collect connection server transport port **Example:** `Router(config-flow-record)# collect connection server transport port`	Configures the server connection port as a nonkey field for a flow record.

	Command or Action	Purpose
Step 54	collect connection transaction {counter complete | duration {sum | min | max}} **Example:** `Router(config-flow-record)# collect connection transaction duration sum`	Configures the total duration of the transaction as a nonkey field for a flow record.
Step 55	collect datalink {destination-vlan-id | source-vlan-id} **Example:** `Router(config-flow-record)# collect datalink destination-vlan-id`	(Optional) Configures the destination VLAN ID as a nonkey field.
Step 56	collect datalink mac {destination | source} address {input | output} **Example:** `Router(config-flow-record)# collect datalink mac destination address input`	(Optional) Configures the destination MAC address as a nonkey field.
Step 57	collect datalink vlan {input | output} **Example:** `Router(config-flow-record)# collect datalink vlan input`	(Optional) Configures the VLAN ID for incoming or outgoing packets as a nonkey field.
Step 58	collect policy performance-monitor classification hierarchy **Example:** `Router(config-flow-record)# collect policy performance-monitor classification hierarchy`	Configures the use of the Performance Monitor policy classification hierarchy as a nonkey field for a flow record.
Step 59	collect services waas {passthrough-reason | segment} **Example:** `Router(config-flow-record)# collect services waas segment`	Configures the use of the WAAS segment as a nonkey field for a flow record.
Step 60	collect timestamp absolute {first | last} **Example:** `Router(config-flow-record)# collect timestamp absolute first`	Configures the use of the first timestamp as a nonkey field for a flow record.
Step 61	collect transport tcp {option map | window-size {sum | minimum | maximum} | maximum-segment-size} **Example:**	Configures the total network delay for the Mediatrace initiator as a nonkey field for a flow record.

	Command or Action	Purpose
	`Router(config-flow-record)# collect connection performance initiator network-delay sum`	
Step 62	end **Example:** `Router(config-flow-record)# end`	Exits flow record configuration mode and returns to privileged EXEC mode.

Configuring a Flow Monitor for Cisco Performance Monitor

The basic concepts for configuring a flow monitor for Cisco Performance Monitor are the same as flow monitors for Flexible NetFlow. Each flow monitor has a separate cache assigned to it and requires a record to define the contents and layout of its cache entries.

When you configure a flow monitor, you must use either:

- An existing flow record that you configured
- One of the following default predefined records:
  - The default RTP record (**default-rtp**)
  - The default TCP record (**default-tcp**)
  - Flexible NetFlow's "NetFlow IPv4 original input"

Note To modify a flow record, you must remove it from all flow monitors it is associated with.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow monitor type performance-monitor** *monitor-name*
4. **description** *description*
5. **cache** {**entries**| **timeout**| **type**}
6. **statistics** {**packet**}
7. **exporter** *exporter-name*
8. **record** {*record-name*| **default-rtp**| **default-tcp**|**netflow ipv4 original-input**}
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow monitor type performance-monitor** *monitor-name* **Example:** `Device(config)# flow monitor type performance-monitor FLOW-MONITOR-2`	Creates a flow monitor and enters flow monitor configuration mode. • This command also allows you to modify an existing flow monitor.
Step 4	**description** *description* **Example:** `Device(config-flow-monitor)# description Used for monitoring IPv4 traffic`	(Optional) Creates a description for the flow monitor.
Step 5	**cache** {**entries**| **timeout**| **type**} **Example:** `Device(config-flow-monitor)# cache timeout 20`	(Optional) Creates a cache for the flow monitor.
Step 6	**statistics** {**packet**} **Example:** `Device(config-flow-monitor)# statistics`	(Optional) specifies whether statistics are collected for the flow monitor.
Step 7	**exporter** *exporter-name* **Example:** `Device(config-flow-monitor)# exporter export-4`	Specifies the flow exporter for the flow monitor.
Step 8	**record** {*record-name*| **default-rtp**| **default-tcp**|**netflow ipv4 original-input**} **Example:** `Device(config-flow-monitor)# record default-rtp`	Specifies the flow record for the flow monitor.
Step 9	**end** **Example:** `Device(config-flow-monitor)# end`	Exits flow monitor configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow monitor, use the **show flow monitor type performance-monitor** command and the **show running-config flow monitor** command.

Configuring a Flow Class for Cisco Performance Monitor

The basic concepts and techniques for configuring a class for Cisco Performance Monitor are the same as for any other type of class. The class specifies the filter that determines which flow traffic to monitor. The filter is configured using various match commands in class-map mode.

If you do not already have a flow monitor configured, you can either:

Note Nested class maps are not supported. In other words, you cannot use the **class-map** command while in class-map configuration mode (config-cmap).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-name*
4. **description** *description*
5. **match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map** *class-map-name* | **cos** *cos-value* | **destination-address mac** *address* | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}}
6. **rename** *class-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**class-map** *class-name* **Example:** `Device(config)# class-map class-4`	Specifies a class to include in the policy. Repeat this command for each class that you want to include in the policy.
Step 4	**description** *description* **Example:** `Device(config-cmap)# description match any packets`	(Optional) Creates a description for the flow class.
Step 5	**match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map** *class-map-name* | **cos** *cos-value* | **destination-address mac** *address* | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}} **Example:** `Device(config-cmap)# match any`	Specifies the classification criteria. For more information and examples, see the *Cisco Media Monitoring Command Reference*.
Step 6	**rename** *class-name* **Example:** `Device(config-cmap)# rename class-4`	Specifies a new name for the flow class.
Step 7	**end** **Example:** `Device(config-cmap)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow class, use the **show policy-map type performance-monitor** or **show class-map** command.

Configuring a Flow Policy for Cisco Performance Monitor Using an Existing Flow Monitor

The basic concepts and techniques for configuring a class for Cisco Performance Monitor are the same as for any other type of class. The class specifies which flow monitor is included. The only significant difference is that, for Cisco Performance Monitor, the **policy-map** command includes **type performance-monitor**.

If you do not already have a flow monitor configured or do not want to use any of your existing flow monitors for a new class, you can configure it using the flow monitor inline option and specifying which flow record and flow exporter are included.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type performance-monitor** *policy-name*
4. **parameter-map type performance-monitor system-default-aor**
5. **class** {*class-name* | **class-default**}
6. **flow monitor** *monitor-name*
7. **monitor metric ip-cbr**
8. **rate layer3** {byte-*rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
9. **exit**
10. **monitor metric rtp**
11. **clock-rate** {*type-number* | *type-name* | **default**} *rate*
12. **max-dropout** *number*
13. **max-reorder** *number*
14. **min-sequential** *number*
15. **ssrc maximum** *number*
16. exit
17. **monitor parameters**
18. **flows** *number*
19. **interval duration** *number*
20. **history** *number*
21. **timeout** *number*
22. **exit**
23. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
24. **action** {**snmp** | **syslog**}
25. **alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**}
26. **alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*}
27. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*}
28. **description** *description*
29. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type performance-monitor** *policy-name* **Example:** `Device(config)# policy-map type performance-monitor FLOW-MONITOR-4`	Creates a policy and enters policy configuration mode. • This command also allows you to modify an existing policy.
Step 4	**parameter-map type performance-monitor system-default-aor** **Example:** `Device(config-pmap)# parameter-map type performance-monitor system-default-aor`	Creates a parameter map for Performance Monitor. The only map available is the system-default -aor map
Step 5	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class class-4`	Specifies a class to include in the policy. Repeat this command for each class that you want to include in the policy.
Step 6	**flow monitor** *monitor-name* **Example:** `Device(config-pmap-c)# flow monitor FLOW-MONITOR-4`	Enters flow monitor configuration mode. If you do not want to use an existing flow monitor, you can use the **inline** option to configure a new one, as described in the Applying a Cisco Performance Monitor Policy to an Interface Without Using an Existing Flow Policy, on page 439.
Step 7	**monitor metric ip-cbr** **Example:** `Device(config-pmap-c)# monitor metric ip-cbr`	(Optional) Enters IP-CBR monitor metric configuration mode.
Step 8	**rate layer3** {byte-*rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-pmap-c-mipcbr)# rate layer3 248 mbps`	(Optional) Specifies the rate for monitoring the metrics. • *byte-rate* --Data rate in Bps, kBps, mBps, or gBps. The range is 1 to 65535. • **packet** --Packet rate in packets per second.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-pmap-c-mipcbr)# exit`	Returns to policy class configuration mode.
Step 10	**monitor metric rtp** **Example:** `Device(config-pmap-c)# monitor metric rtp`	Enters RTP monitor metric configuration mode.
Step 11	**clock-rate** {*type-number* | *type-name* | **default**} *rate* **Example:** `Device(config-pmap-c-mrtp)# clock-rate 8 9600`	Specifies the clock rate used to sample RTP video-monitoring metrics. For more information about the clock-type numbers and names, see the *Cisco Media Monitoring Command Reference.* The range for *rate* is 1 kHz to 192 kHz.
Step 12	**max-dropout** *number* **Example:** `Device(config-pmap-c-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.
Step 13	**max-reorder** *number* **Example:** `Device(config-pmap-c-mrtp)# max-reorder 4`	Specifies the maximum number of reorders allowed when sampling RTP video-monitoring metrics.
Step 14	**min-sequential** *number* **Example:** `Device(config-pmap-c-mrtp)# min-sequential 2`	Specifies the minimum number of sequential packets required to identify a stream as being an RTP flow.
Step 15	**ssrc maximum** *number* **Example:** `Device(config-pmap-c-mrtp)# ssrc maximum 20`	Specifies the maximum number of SSRCs that can be monitored within the same flow. A flow is defined by the protocol, source/destination address, and source/destination port).
Step 16	exit **Example:** `Device(config-pmap-c-mrtp)# exit`	Returns to policy class configuration mode.
Step 17	**monitor parameters** **Example:** `Device(config-pmap-c)# monitor parameters`	Enters monitor parameters configuration mode.

	Command or Action	Purpose
Step 18	**flows** *number* **Example:** `Device(config-pmap-c-mparam)# flows 40`	Specifies the maximum number of flows for each monitor cache.
Step 19	**interval duration** *number* **Example:** `Device(config-pmap-c-mparam)# interval duration 40`	Specifies the interval, in seconds, between samples taken of video-monitoring metrics.
Step 20	**history** *number* **Example:** `Device(config-pmap-c-mparam)# history 4`	Specifies the number of historical buckets of collected video-monitoring metrics.
Step 21	**timeout** *number* **Example:** `Device(config-pmap-c-mparam)# timeout 20`	Specifies the number of intervals before a stopped flow is removed from the database.
Step 22	**exit** **Example:** `Device(config-pmap-c-mparam)# exit`	Returns to policy class configuration mode.
Step 23	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:** `Device(config-pmap-c)# react 41 rtp-jitter-average`	Enters a mode where you can specify what reaction occurs when a threshold is violated for the following metrics: • *ID*-- ID for react configuration. Range is 1 to 65535. • **media-stop** --No traffic is found for the flow. • **mrv** --Ratio calculated by dividing the difference between the actual rate and the expected rate, by the expected rate. • **rtp-jitter-average** --Average jitter. • **transport-packets-lost-rate** --Ratio calculated by dividing the number of lost packets by the expected packet count.
Step 24	**action** {**snmp** | **syslog**} **Example:** `Device(config-pmap-c-react)# action syslog`	Specifies how violations of the thresholds with be reported.

	Command or Action	Purpose
Step 25	**alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**} **Example:** `Device(config-pmap-c-react)# alarm severity critical`	Specifies which level of alarm will be reported. The default setting is **info**.
Step 26	**alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*} **Example:** `Device(config-pmap-c-react)# alarm type discrete`	Specifies which types of levels are considered alarms that require reporting. The default setting is **discrete**.
Step 27	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*} **Example:** `Device(config-pmap-c-react)# threshold value ge 20`	Specifies which types of threshold values are considered alarms that require reporting. If no value is set but the application name is configured as a key field, then the system uses the value for the threshold that it finds in the default map. If no value is set and the application name is not configured as a key field, then the default value is used for the threshold. If more than one react command is configured for the same policy and class but only one of the react configurations has threshold values set, then the values of the configured react take precedence and the rest of the threshold values are ignored. If more than one react command is configured for the same policy and none of them have the threshold value configured, then the default threshold value is applied for the configuration with the lowest react ID.
Step 28	**description** *description* **Example:** `Device(config-cmap-c-react)# description rtp-jitter-average above 40`	(Optional) Creates a description for the reaction.
Step 29	**end** **Example:** `Device(config-pmap-c-react)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow policy, use the **show policy-map type performance-monitor** command.

Configuring a Flow Policy for Cisco Performance Monitor Without Using an Existing Flow Monitor

The basic concepts and techniques for configuring a class for Cisco Performance Monitor are the same as for any other type of class. The class specifies which flow monitor is included. The only significant difference is that, for Cisco Performance Monitor, the **policy-map** command includes**type performance-monitor**.

If you do not already have a flow monitor configured or do not want to use any of your existing flow monitors for a new class, you can configure it under the class configuration mode, by specifying which flow record and flow exporter are included.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type performance-monitor** *policy-name* **class** class-name
4. **parameter-map type performance-monitor system-default-aor**
5. **class** {*class-name* | **class-default**}
6. **flow monitor inline**
7. **record** {*record-name* | **default-rtp** | **default-tcp**}
8. **exporter** *exporter-name*
9. **exit**
10. monitor metric ip-cbr
11. **rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
12. **exit**
13. **monitor metric rtp**
14. **clock-rate** {*type-number*| *type-name*} *rate*
15. **max-dropout** *number*
16. **max-reorder** *number*
17. **min-sequential** *number*
18. **ssrc maximum** *number*
19. exit
20. **monitor parameters**
21. **flows** *number*
22. **interval duration** *number*
23. **history** *number*
24. **timeout** *number*
25. **exit**
26. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
27. **action** {**snmp** | **syslog**}
28. **alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**}
29. **alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*}
30. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*
31. **description** *description*
32. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type performance-monitor** *policy-name* **class** class-name **Example:** `Device(config)# policy-map type performance-monitor FLOW-MONITOR-4`	Creates a policy and enters policy configuration mode. • This command also allows you to modify an existing policy.
Step 4	**parameter-map type performance-monitor system-default-aor** **Example:** `Device(config-pmap)# parameter-map type performance-monitor system-default-aor`	Creates a parameter map for Performance Monitor. The only map available is the system-default -aor map
Step 5	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class class-4`	Specifies a class to include in the policy. Repeat this command for each class that you want to include in the policy.
Step 6	**flow monitor inline** **Example:** `Device(config-pmap-c)# flow monitor inline`	Enters inline mode and enables you to configure a new flow monitor.
Step 7	**record** {*record-name* | **default-rtp** | **default-tcp**} **Example:** `Device(config-pmap-c-flowmon)# record default-tcp`	Specifies a flow record to associate with the flow monitor.
Step 8	**exporter** *exporter-name* **Example:** `Device(config-pmap-c-flowmon)# exporter exporter-4`	Specifies a flow record to associate with the flow exporter.
Step 9	**exit** **Example:**	Returns to policy class configuration mode.

	Command or Action	Purpose
	`Device(config-pmap-c-flowmon)# exit`	
Step 10	monitor metric ip-cbr **Example:** `Device(config-pmap-c)# monitor metric ip-cbr`	(Optional) Enters IP-CBR monitor metric configuration mode.
Step 11	**rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-pmap-c-mipcbr)# rate layer3 248 mbps`	(Optional) Specifies the rate for monitoring the metrics. • *byte-rate*—Data rate in Bps, kBps, mBps, or gBps. The range is 1 to 65535. • **packet**—Packet rate in packets per second.
Step 12	**exit** **Example:** `Device(config-pmap-c-mipcbr)# exit`	Returns to policy class configuration mode.
Step 13	**monitor metric rtp** **Example:** `Device(config-pmap-c)# monitor metric rtp`	Enters RTP monitor metric configuration mode.
Step 14	**clock-rate** {*type-number*| *type-name*} *rate* **Example:** `Device(config-pmap-c-mrtp)# clock-rate 8 9600`	Specifies the clock rate used to sample RTP video-monitoring metrics. For more information about the clock-type numbers and names, see the *Cisco Media Monitoring Command Reference*. The range for *rate* is 1 kHz to 192 kHz.
Step 15	**max-dropout** *number* **Example:** `Device(config-pmap-c-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.
Step 16	**max-reorder** *number* **Example:** `Device(config-pmap-c-mrtp)# max-reorder 4`	Specifies the maximum number of reorders allowed when sampling RTP video-monitoring metrics.
Step 17	**min-sequential** *number* **Example:** `Device(config-pmap-c-mrtp)# min-sequential 2`	Specifies the minimum number of sequential packets required to identify a stream as being an RTP flow.

	Command or Action	Purpose
Step 18	**ssrc maximum** *number* **Example:** `Device(config-pmap-c-mrtp)# ssrc maximum 20`	Specifies the maximum number of SSRCs that can be monitored within the same flow. A flow is defined by the protocol, source/destination address, and source/destination port).
Step 19	exit **Example:** `Device(config-pmap-c-mrtp)# exit`	Returns to policy class configuration mode.
Step 20	**monitor parameters** **Example:** `Device(config-pmap-c)# monitor parameters`	Enters monitor parameters configuration mode.
Step 21	**flows** *number* **Example:** `Device(config-pmap-c-mparam)# flows 40`	Specifies the maximum number of flows for each monitor cache.
Step 22	**interval duration** *number* **Example:** `Device(config-pmap-c-mparam)# interval duration 40`	Specifies the duration of the intervals, in seconds, for collecting monitoring metrics.
Step 23	**history** *number* **Example:** `Device(config-pmap-c-mparam)# history 4`	Specifies the number of historical intervals of collected monitoring metrics to display.
Step 24	**timeout** *number* **Example:** `Device(config-pmap-c-mparam)# timeout 20`	Specifies the number intervals before a stopped flow is removed from the database.
Step 25	**exit** **Example:** `Device(config-pmap-c-mparam)# exit`	Returns to policy class configuration mode.
Step 26	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:** `Device(config-pmap-c)# react 41 rtp-jitter-average`	Enters a mode where you can specify what reaction occurs when a threshold is violated for the following metrics: • *ID*—ID for react configuration. Range is 1 to 65535. • **media-stop**—No traffic is found for the flow.

	Command or Action	Purpose
		• **mrv**—Ratio calculated by dividing the difference between the actual rate and the expected rate, by the expected rate. • **rtp-jitter-average**—Average jitter. • **transport-packets-lost-rate**—Ratio calculated by dividing the number of lost packets by the expected packet count.
Step 27	**action** {**snmp** | **syslog**} **Example:** `Device(config-pmap-c-react)# action syslog`	Specifies how violations of the thresholds with be reported.
Step 28	**alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**} **Example:** `Device(config-pmap-c-react)# alarm severity critical`	Specifies which level of alarm will be reported.The default setting is **info**.
Step 29	**alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*} **Example:** `Device(config-pmap-c-react)# alarm severity critical`	Specifies which types of levels are considered alarms that require reporting. The default setting is **discrete**.
Step 30	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end* **Example:** `Device(config-pmap-c-react)# threshold value ge 20`	Specifies which types of threshold values are considered alarms that require reporting. If no value is set but the application name is configured as a key field, then the system uses the value for the threshold that it finds in the default map. If no value is set and the application name is not configured as a key field, then the default value is used for the threshold. If more than one react command is configured for the same policy and class but only one of the react configurations has threshold values set, then the values of the configured react take precedence and the rest of the threshold values are ignored. If more than one react command is configured for the same policy and none of them have the threshold value configured, then the default threshold value is applied for the configuration with the lowest react ID.
Step 31	**description** *description* **Example:**	(Optional) Creates a description for the reaction.

	Command or Action	Purpose
	`Device(config-cmap-c-react)# description rtp-jitter-average above 40`	
Step 32	**end** **Example:** `Device(config-pmap-c-react)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your flow policy, use the **show policy-map type performance-monitor** command.

Applying a Cisco Performance Monitor Policy to an Interface Using an Existing Flow Policy

Before it can be activated, a Cisco Performance Monitor policy must be applied to at least one interface. To activate a Cisco Performance Monitor policy, perform the following required task.

Note You can apply a Cisco Performance Monitor policy to an IPv6 interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-policy type performance-monitor** {**input** | **output**} *policy-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface and enters interface configuration mode. You can specify an IPv6 interface.
Step 4	**service-policy type performance-monitor** {**input** | **output**} *policy-name* **Example:** **Example:** `Device(config-if)# service-policy type performance-monitor input mypolicy-map-4` **Example:**	Attaches a policy map to an input interface or virtual circuit (VC), or an output interface or VC, to be used as the service policy for that interface or VC. • **input**—Attaches the specified policy map to the input interface or input VC. • **output**—Attaches the specified policy map to the output interface or output VC. • *policy-name*—name of a service policy map (created by the **policy-map** command) to be attached. The name can be a maximum of 40 alphanumeric characters.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

To check the configuration and status of your service policy, use the following commands:

- **show performance monitor history**
- **show performance monitor status**
- **show policy-map ypre performance-monitor interface**

Applying a Cisco Performance Monitor Policy to an Interface Without Using an Existing Flow Policy

Before it can be activated, a Cisco Performance Monitor policy must be applied to at least one interface. To activate a Cisco Performance Monitor policy, perform the following required task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-policy type performance-monitor inline** {**input** | **output**}
5. **match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map***class-map-name* | **cos** *cos-value* | **destination-address mac** address | **discard-class** *class-number* | **dscp** *dscp-value* |

flow {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}}

6. **flow monitor** {*monitor-name*| **inline**}
7. **record** {*record-name*| **default-rtp**| **default-tcp**}
8. **exporter** *exporter-name*
9. **exit**
10. **monitor metric ip-cbr**
11. **rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
12. **exit**
13. **monitor metric rtp**
14. **clock-rate** {*type-number*| *type-name*} *rate*
15. **max-dropout** *number*
16. **max-reorder** *number*
17. **min-sequential** *number*
18. **ssrc maximum** *number*
19. exit
20. **monitor parameters**
21. **flows** *number*
22. **interval duration** *number*
23. **history** *number*
24. **timeout** *number*
25. **exit**
26. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
27. **action** {**snmp** | **syslog**}
28. **alarm severity** {**alert** | **critical** | **emergency**| **error** | **info**}
29. **alarm type** {**discrete**| **grouped**{**count** *number* | **percent** *number*}}
30. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*}
31. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface and enters interface configuration mode. You can specify an IPv6 interface.
Step 4	**service-policy type performance-monitor inline** {**input** | **output**} **Example:** **Example:** `Device(config-if)# service-policy type performance-monitor inline input`	Attaches a policy map to an input interface or virtual circuit (VC), or an output interface or VC, to be used as the service policy for that interface or VC. • **input**—Attaches the specified policy map to the input interface or input VC. • **output**—Attaches the specified policy map to the output interface or output VC.
Step 5	**match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map***class-map-name* | **cos** *cos-value* | **destination-address mac** address | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}} **Example:** `Device(config-if-spolicy-inline)# match any`	Specifies the classification criteria. For more information and examples, see the *Cisco Media Monitoring Command Reference* .
Step 6	**flow monitor** {*monitor-name*| **inline**} **Example:** `Device(config-if-spolicy-inline)# flow monitor inline`	Specifies an existing flow monitor to associate with a flow policy. If you do not want to use an existing flow monitor, you can use the **inline** option to configure a new one. If needed, you can also use the **inline** option to specify a flow record and flow exporter.
Step 7	**record** {*record-name*| **default-rtp**| **default-tcp**} **Example:** `Device(config-spolicy-inline-flowmon)# record default-tcp`	(Optional) If you do not want to use an existing flow monitor, and instead used the **inline** option, use this command to configure a flow record.

	Command or Action	Purpose
Step 8	**exporter** *exporter-name* **Example:** `Device(config-spolicy-inline-flowmon)# exporter exporter-4`	(Optional) If you do not want to use an existing flow monitor, and instead used the **inline** option, use this command to configure a flow exporter.
Step 9	**exit** **Example:** `Device(config-spolicy-inline-flowmon)# exit`	Returns to service-policy inline configuration mode.
Step 10	**monitor metric ip-cbr** **Example:** `Device(config-if-spolicy-inline)# monitor metric ip-cbr`	Enters IP-CBR monitor metric configuration mode.
Step 11	**rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-spolicy-inline-mipcbr)# rate layer3 248 mbps`	Specifies the rate for monitoring the metrics. • *byte-rate*—Data rate in Bps, kBps, mBps, or gBps. The range is 1 to 65535. • **packet**—Packet rate in packets per second.
Step 12	**exit** **Example:** `Device(config-spolicy-inline-mipcbr)# exit`	Returns to service-policy inline configuration mode.
Step 13	**monitor metric rtp** **Example:** `Device(config-if-spolicy-inline)# monitor metric rtp`	Enters RTP monitor metric configuration mode.
Step 14	**clock-rate** {*type-number*| *type-name*} *rate* **Example:** `Device(config-spolicy-inline-mrtp)# clock-rate 8 9600`	Specifies the clock rate used to sample RTP video-monitoring metrics. For more information about the clock-type numbers and names, see the *Cisco Media Monitoring Command Reference.* The range for *rate* is 1 kHz to 192 kHz.
Step 15	**max-dropout** *number* **Example:** `Device(config-spolicy-inline-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.

	Command or Action	Purpose
Step 16	**max-reorder** *number* **Example:** `Device(config-spolicy-inline-mrtp)# max-reorder 4`	Specifies the maximum number of reorders allowed when sampling RTP video-monitoring metrics.
Step 17	**min-sequential** *number* **Example:** `Device(config-spolicy-inline-mrtp)# min-sequential 2`	Specifies the minimum number of sequential packets required to identify a stream as being an RTP flow.
Step 18	**ssrc maximum** *number* **Example:** `Device(config-spolicy-inline-mrtp)# ssrc maximum 20`	Specifies the maximum number of SSRCs that can be monitored within the same flow. A flow is defined by the protocol, source/destination address, and source/destination port).
Step 19	exit **Example:** `Device(config-spolicy-inline-mrtp)# exit`	Returns to service-policy inline configuration mode.
Step 20	**monitor parameters** **Example:** `Device(config-if-spolicy-inline)# monitor parameters`	Enters monitor parameters configuration mode.
Step 21	**flows** *number* **Example:** `Device(config-spolicy-inline-mparam)# flows 40`	Specifies the maximum number of flows for each monitor cache.
Step 22	**interval duration** *number* **Example:** `Device(config-spolicy-inline-mparam)# interval duration 40`	Specifies the duration of the intervals, in seconds, for collecting monitoring metrics.
Step 23	**history** *number* **Example:** `Device(config-spolicy-inline-mparam)# history 4`	Specifies the number of historical intervals of collected monitoring metrics to display.
Step 24	**timeout** *number* **Example:**	Specifies the number of intervals before a stopped flow is removed from the database.

	Command or Action	Purpose
	`Device(config-spolicy-inline-mparam)# timeout 20`	
Step 25	**exit** **Example:** `Device(config-spolicy-inline-mparam)# exit`	Returns to service-policy inline configuration mode.
Step 26	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:** `Device(config-if-spolicy-inline)# react 6 rtp-jitter-average`	Enters a mode where you can specify what reaction occurs when a threshold is violated for the following metrics: • *ID*— ID for react configuration. Range is 1 to 65535. • **media-stop** —No traffic is found for the flow. • **mrv** —Ratio calculated by dividing the difference between the actual rate and the expected rate, by the expected rate. • **rtp-jitter-average** —Average jitter. • **transport-packets-lost-rate** —Ratio calculated by dividing the number of lost packets by the expected packet count.
Step 27	**action** {**snmp** | **syslog**} **Example:** `Device(config-spolicy-inline-react)# action syslog`	Specifies how violations of the thresholds with be reported.
Step 28	**alarm severity** {**alert** | **critical** | **emergency**| **error** | **info**} **Example:** `Device(config-spolicy-inline-react)# alarm severity critical`	Specifies which level of alarm will be reported.
Step 29	**alarm type** {**discrete**| **grouped**{**count** *number* | **percent** *number*}} **Example:** `Device(config-pspolicy-inline-react)# alarm severity critical`	Specifies which types of levels are considered alarms that require reporting.
Step 30	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*} **Example:** `Device(config-spolicy-inline-react)# threshold value ge 20`	Specifies which types of threshold values are considered alarms that require reporting. If no value is set but the application name is configured as a key field, then the system uses the value for the threshold that it finds in the default map. If no value is set and the application name is not configured as a key field, then the default value is used for the threshold.

	Command or Action	Purpose
		If more than one react command is configured for the same policy and class but only one of the react configurations has threshold values set, then the values of the configured react take precedence and the rest of the threshold values are ignored. If more than one react command is configured for the same policy and none of them have the threshold value configured, then the default threshold value is applied for the configuration with the lowest react ID.
Step 31	**end** **Example:** `Device(config-spolicy-inline-react)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

What to do next

To check the configuration and status of your service policy, use the **show performance monitor status** command and **show performance monitor history** command.

Verifying That Cisco Performance Monitor Is Collecting Data

To verify that Cisco Performance Monitor is collecting data, perform the following optional task.

Note Flows are correlated so that if the same policy is applied on the same input and output interface, the **show** command will display a single flow for the input and output interfaces and the interface name and direction for the flow are not displayed.

If no data is being collected, complete the remaining tasks in this section.

Before you begin

The interface to which you applied the input flow monitor must be receiving traffic that meets the criteria defined by the original flow record before you can display the flows in the flow monitor cache.

where *filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

SUMMARY STEPS

1. **enable**
2. **show policy-map type performance-monitor** [**interface** *interface-name*][**class** *class-name*][**input** | **output**]
3. **show performance monitor status** [**interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

4. **show performance monitor history** [**interval** {**all**| *number*[**start** *number*]} | **interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

DETAILED STEPS

Step 1 **enable**

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 **show policy-map type performance-monitor** [**interface** *interface-name*][**class** *class-name*][**input** | **output**]

For a description of the fields displayed by this command, see *Cisco Media Monitoring Command Reference*.

The following example shows the output for one flow policy:

Example:

```
Policy Map type performance-monitor PM-POLICY-4
  Class PM-CLASS-4
    flow monitor PM-MONITOR-4
      record PM-RECORD-4
      exporter PM-EXPORTER-4
    monitor parameters
      interval duration 30
      timeout 10
      history 10
      flows 8000
    monitor metric rtp
      min-sequential 5
      max-dropout 5
      max-reorder 5
      clock-rate default 90000
      ssrc maximum 5
```

Table 43: show policy-map type performance-monitor Field Descriptions

Field	Description
Policy Map type performance-monitor	Name of the Cisco Performance Monitor flow policy.
flow monitor	Name of the Cisco Performance Monitor flow monitor.
record	Name of the Cisco Performance Monitor flow record.
exporter	Name of the Cisco Performance Monitor flow exporter.
monitor parameter	Parameters for the flow policy.
interval duration	The configured duration of the collection interval for the policy.
timeout	The configured amount of time wait for a response when collecting data for the policy.

Field	Description
history	The configured number of historical collections to keep for the policy.
flows	The configured number of flows to collect for the policy.
monitor metric rtp	RTP metrics for the flow policy.
min-sequential	The configured minimum number of packets in a sequence used to classify an RTP flow.
max-dropout	The configured maximum number of packets to ignore ahead of the current packet in terms of sequence number.
max-reorder	The configured maximum number of packets to ignore behind the current packet in terms of sequence number.
clock-rate default	The configured clock rate for the RTP packet timestamp clock that is used to calculate the packet arrival latency.
ssrc maximum	The configured maximum number of SSRCs that can be monitored within the same flow. A flow is defined by the protocol, source/destination address, and source/destination port. The range is from 1 to 50.

Step 3 **show performance monitor status** [**interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

where *filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

This command displays the cumulative statistics for the specified number of most recent intervals. The number of intervals is configured using the **history** command. The default settings for this commands is 10 of the most recent collection intervals. The duration of collection intervals is specified by the **interval duration** command.

To view statistics for other intervals, use the **show performance monitor history** command as described in the next step. For more information about these commands, see the *Cisco Media Monitoring Command Reference*

Step 4 **show performance monitor history** [**interval**{**all**| *number*[**start** *number*]} | **interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

where *filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

This command displays the statistics collected by Cisco Performance Monitor during any or all intervals, including the current one. The duration of collection intervals is specified by the **interval duration** command.

For more information about this command, see the *Cisco Media Monitoring Command Reference.*

The following example shows the output for the **show performance monitor history** command**:**

Note If the same policy is applied on the same input and output interface, the display shows a single flow for the input and output interfaces and the interface name and direction for the flow are not displayed.

Example:

```
Codes: *   - field is not configurable under flow record
       NA  - field is not applicable for configured parameters
Match: ipv4 source address = 21.21.21.1, ipv4 destination address = 1.1.1.1,
transport source-port = 10240, transport destination-port = 80, ip protocol = 6,
 Policy: RTP_POL, Class: RTP_CLASS

 start time                                  14:57:34
                                             ============
 *history bucket number                     : 1
 routing forwarding-status                  : Unknown
 transport packets expected counter         : NA
 transport packets lost counter             : NA
 transport round-trip-time           (msec) : 4
 transport round-trip-time sum       (msec) : 8
 transport round-trip-time samples          : 2
 transport event packet-loss counter        : 0
 interface input                            : Null
 interface output                           : Null
 counter bytes                              : 8490
 counter packets                            : 180
 counter bytes rate                         : 94
 counter client bytes                      : 80
 counter server bytes                      : 200
 counter client packets                    : 6
 counter server packets                    : 6
 transport tcp window-size minimum         : 1000
 transport tcp window-size maximum         : 2000
 transport tcp window-size average         : 1500
 transport tcp maximum-segment-size        : 0
 application media bytes counter            : 1270
 application media bytes rate               : 14
 application media packets counter          : 180
 application media event                    : Stop
 monitor event                              : false

 [data set,id=257] Global session ID|Multi-party session ID|
 [data] 11                         |22
```

Table 44: show performance monitor status and show performance-monitor history Field Descriptions

Field	Description
history bucket number	Number of the bucket of historical data collected.

Field	Description
routing forwarding-status reason	

Field	Description
	Forwarding status is encoded using eight bits with the two most significant bits giving the status and the six remaining bits giving the reason code. Status is either unknown (00), Forwarded (10), Dropped (10) or Consumed (11). The following list shows the forwarding status values for each status category. **Unknown** • 0 **Forwarded** • Unknown 64 • Forwarded Fragmented 65 • Forwarded not Fragmented 66 **Dropped** • Unknown 128, • Drop ACL Deny 129, • Drop ACL drop 130, • Drop Unroutable 131, • Drop Adjacency 132, • Drop Fragmentation & DF set 133, • Drop Bad header checksum 134, • Drop Bad total Length 135, • Drop Bad Header Length 136, • Drop bad TTL 137, • Drop Policer 138, • Drop WRED 139, • Drop RPF 140, • Drop For us 141, • Drop Bad output interface 142, • Drop Hardware 143, **Consumed** • Unknown 192, • Terminate Punt Adjacency 193, • Terminate Incomplete Adjacency 194,

Field	Description
	• Terminate For us 195
transport packets expected counter	Number of packets expected.
transport packets lost counter	Number of packets lost.
transport round-trip-time (msec)	Number of milliseconds required to complete a round trip.
transport round-trip-time sum (msec)	Total number of milliseconds required to complete a round trip for all samples.
transport round-trip-time samples	Total number of samples used to calculate a round trip times
transport event packet-loss counter	Number of loss events (number of contiguous sets of lost packets).
interface input	Incoming interface index.
interface output	Outgoing interface index.
counter bytes	Total number of bytes collected for all flows.
counter packets	Total number of IP packets sent for all flows.
counter bytes rate	Average number of packets or bits (as configured) processed by the monitoring system per second during the monitoring interval for all flows.
counter client bytes	Number of bytes sent by the client.
counter server bytes	Number of bytes sent by the server.
counter client packets	Number of packets sent by the client.
counter servers packets	Number of packets sent by the server.
transport tcp window-size-maximum	Maximum size of the TCP window.
transport tcp window-size-minimum	Minimum size of the TCP window.
transport tcp window-size-average	Average size of the TCP window.
transport tcp maximum-segment-size	Maximum TCP segment size.
application media bytes counter	Number of IP bytes from by media applications received for a specific media stream.
application media bytes rate	Average media bit rate (bps) for all flows during the monitoring interval.
application media packets counter	Number of IP packets produced from media applications received for a specific media stream.
application media event	Bit 1 is not used. Bit 2 indicates that no media application packets were seen, in other words, a Media Stop Event occurred.

Field	Description
monitor event	Bit 1 indicates that one of the thresholds specified by a react statement for the flow was crossed at least once in the monitoring interval. Bit 2 indicates that there was a loss-of-confidence in measurement.

Displaying Option Tables.

You can view the mapping contained in the various option table by using the following **show** command .

SUMMARY STEPS

1. **enable**
2. **show metadata** {**application attributes** | **application table** | **exporter stats** | **interface table** | **metadata version table** | **sampler table** | **vrf table**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show metadata** {**application attributes**	**application table**	**exporter stats**	**interface table**	**metadata version table**	**sampler table**	**vrf table**} **Example:**	The following example shows how to display the mapping of the application ID to the application name by using the **show metadata application table** command : `ID Name Vendor Version` `100673296 webex-audio - -` `100673297 webex-video - -`

Displaying Information Specific to the Catalyst 6500 Platform

To display or clear information for the Feature Manager and other functionality specific to the Catalyst 6500 platform, perform the following optional task.

SUMMARY STEPS

1. **enable**
2. **clear fm performance-monitor counters**
3. **debug fm performance-monitor** {**all** | **dynamic** | **event** | **unusual** | **verbose** | **vmr**}
4. **platform performance-monitor rate-limit pps** *number*
5. **show platform software feature-manager performance-monitor** {**all** | **counters** | **interface** *interface-type interface-number* | **rdt-indices** }

6. **show platform software feature-manager tcam dynamic performance-monitor** {**handle ip** *ip-address* | **interface** *interface-type interface-number* }
7. **show platform hardware acl entry interface** *interface-type interface-number* **security** {**in** | **out** } {**ip** | **ipv6** } [**detail**]
8. **show platform software ccm interface** *interface-type interface-number* **security** {**interface** *interface-type interface-number* | **class-group** *class-group-ID* }

DETAILED STEPS

Step 1 **enable**

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 **clear fm performance-monitor counters**

The **clearfm performance-monitor counters** command clears counters for the Performance Monitor component of Feature Monitor.

Example:

```
Device# clear fm performance-monitor counters
Device#
```

Step 3 **debug fm performance-monitor** {**all** | **dynamic** | **event** | **unusual** | **verbose** | **vmr**}

This command enables all levels of debugging for the Performance Monitor component of Feature Manager.

Example:

```
Device# debug fm performance-monitor all
Device#
```

Step 4 **platform performance-monitor rate-limit pps** *number*

This command sets the rate limit for the Performance Monitor component of Feature Monitor.

Example:

```
Device# platform performance-monitor rate-limit pps 2000
Device#
```

Step 5 **show platform software feature-manager performance-monitor** {**all** | **counters** | **interface** *interface-type interface-number* | **rdt-indices** }

This command displays information about the Performance Monitor component of Feature Manager.

Example:

```
Device# show platform software feature-manager performance-monitor all
Device#

Interface: FastEthernet2/3
```

```
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Ingress L3
==============================================================================
DPort  - Destination Port  SPort  - Source Port       Pro    - Protocol
RFTCM  - R-Recirc. Flag    MRLCS  - M-Multicast Flag  Res    - VMR Result
       - F-Fragment flag           - R-Reflexive flag  Prec   - Drop Precedence
       - T-Trailing Fragments      - L-Layer 3 only    GrpId  - Qos Group Id
       - C-From CPU                - C-Capture Flag    Adj.   - Adj. Index
       - M-L2 Lookup Miss          - S-RPF suppress    Pid    - NF Profile Index
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V       224.0.0.0         0.0.0.0                  0          0      0         -----       0
       -----
       M       240.0.0.0         0.0.0.0                   0          0      0         00000    0
       0
      PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
      L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
     ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT

     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
     ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT

     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
      PERMIT_RESULT
```

```
Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Egress L3
==============================================================================
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
      L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
     -----
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
     -----
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
       L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
     PERMIT_RESULT Adjacency: 0x5512D8F4


Adjacency: 0x5512D8F4
        FeatureId: 0x84  AdjId: 0xFFFFFFFF Flags: RecirculationAdj|
```

```
        Cause: 0x0 Priority: 0xC Device#

Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Ingress L3
==============================================================================
DPort  - Destination Port  SPort  - Source Port       Pro    - Protocol
RFTCM  - R-Recirc. Flag    MRLCS  - M-Multicast Flag  Res    - VMR Result
       - F-Fragment flag          - R-Reflexive flag  Prec   - Drop Precedence
       - T-Trailing Fragments     - L-Layer 3 only    GrpId  - Qos Group Id
       - C-From CPU               - C-Capture Flag    Adj.   - Adj. Index
       - M-L2 Lookup Miss         - S-RPF suppress    Pid    - NF Profile Index
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V       224.0.0.0         0.0.0.0                  0          0      0         -----        0
       -----
       M       240.0.0.0         0.0.0.0                   0          0      0         00000    0
       0
      PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
      L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
     ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT

     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
     ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
 0
      PERMIT_RESULT

     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000     0
        0
L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          0.0.0.0                  0          0      0         -----
 0      -----
```

```
       M           0.0.0.0          0.0.0.0                    0           0       0          00000      0
        0
      PERMIT_RESULT


Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Egress L3
==============================================================================
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V          0.0.0.0          0.0.0.0                   0           0       0          -----
 0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000      0
        0
PERMIT_RESULT

     2 V          0.0.0.0          0.0.0.0                   0           0       0          -----
 0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000      0
        0
      L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V          0.0.0.0          10.10.10.0             0           0       17         -----         0
     -----
       M           0.0.0.0           255.255.255 0       0           0       255       00000      0
 0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     2 V          0.0.0.0          10.10.20.0             0           0       17         -----         0
     -----
       M           0.0.0.0           255.255.255 0       0           0       255       00000      0
 0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     3 V          0.0.0.0          0.0.0.0                   0           0       0          -----
 0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000      0
        0
       L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     3 V          0.0.0.0          0.0.0.0                   0           0       0          -----
 0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000      0
        0
     PERMIT_RESULT Adjacency: 0x5512D8F4
```

```
Adjacency: 0x5512D8F4
        FeatureId: 0x84  AdjId: 0xFFFFFFFF Flags: RecirculationAdj|

        Cause: 0x0 Priority: 0xC
```

Step 6 **show platform software feature-manager tcam dynamic performance-monitor** {**handle ip** *ip-address* | **interface** *interface-type interface-number* }

This command displays information about dynamic and static policies for a specific host.

Example:

```
Device# show platform software feature-manager tcam dynamic performance-monitor handle ip 10.1.1.0
------------------------------------------------------------------------------
HANDLE                      Feature ID   No of entries     MD5
------------------------------------------------------------------------------
10.1.1.0            VM Ingress L3                        2
```

Step 7 **show platform hardware acl entry interface** *interface-type interface-number* **security** {**in** | **out** } {**ip** | **ipv6** } [**detail**]

This command displays inbound access control list (ACL) entries for IP on an interface.

Example:

```
Device# show platform hardware acl entry interface fastEthernet 1/1 security in ip detail

mls_if_index:2000400A dir:0 feature:0 proto:0


pass#0 features
UAPRSF: U-urg, A-ack, P-psh, R-rst, S-syn, F-fin
MLGFI: M-mpls_plus_ip_pkt, L-L4_hdr_vld, G-gpid_present,F-global_fmt_match, I-ife/ofe
's' means set; 'u' means unset; '-' means don't care
-------------------------------------------------------------------------------
INDEX  LABEL FS ACOS    AS     IP_SA      SRC_PORT       IP_DA      DST_PORT        F FF        L4PROT


TCP-F:UAPRSF MLGFI OtherL4OPs                                    RSLT                  CNT
-------------------------------------------------------------------------------
fno:0

tcam:B, bank:0, prot:0    Aces

I  V  16375   2049  0    0     0         0.0.0.0              -         0.0.0.0              - 0
 0      0      -       -----          -
0x0000000800000038            10331192<-
I  M  16375 0x1FFF  0 0x00 0x000         0.0.0.0              -         0.0.0.0              - 0
 0    0x0
```

Step 8 **show platform software ccm interface** *interface-type interface-number* **security** {**interface** *interface-type interface-number* | **class-group** *class-group-ID* }

This command displays information about ternary content addressable memory (TCAM) Cisco CallManager (CCM) entries on an interface.

Example:

```
Device# show platform software ccm interface fastEthernet 2/3 in
```

```
 Target-Class : id 0xA0000000, dir CCM_INPUT, if_type 1, if_info 0x14823998

 Class-Group List: 0xA0000001
b1-cs217#

b1-cs217#sh platform software ccm interface fastEthernet 2/3 out

 Target-Class : id 0xA0000002, dir CCM_OUTPUT, if_type 1, if_info 0x14823998

 Class-Group List: 0xA0000001
```

This command displays information about ternary content addressable memory (TCAM) Cisco CallManager (CCM) entries for a class group

Example:

```
Device# show platform software ccm class-group A0000001
 Class-group    : video-flow-test, id 0xA0000001
 Target input   : 0xA0000000
 Target Output  : 0xA0000002
        Class   : video-flow, id 0xA98681, type 1
                Filter          : type MATCH_NUMBERED_ACCESS_GROUP, id 0xF0000002
                Filter params   : ACL Index: 101 Linktype: 7

                Feature         : PERFORMANCE_MONITOR
                Params          :
                  Feature Object : 0x54224218
                    Name           :
                    Meter context  : 0x54264440
                    Sibling        : 0x0
                    Dynamic        : FALSE
                  Feature Object : 0x54221170
                    Name           :
                    Meter context  : 0x54263858
                    Sibling        : 0x0
                    Dynamic        : FALSE
                Intf List       : 0xA0000000  0xA0000002
        Class   : class-default, id 0xADA3F1, type 39
                Filter          : type MATCH_ANY, id 0xF0000003
                Filter params   : any

                Feature         : FEATURE_EMPTY
                Params          :
                  Feature Object : 0x1741629C
                    Name           :
                    Meter context  : 0x0
                    Sibling        : 0x0
                    Dynamic        : FALSE
                Intf List       : 0xA0000000  0xA0000002
```

Displaying the Performance Monitor Cache and Clients

To display the cache and the clients for Cisco Performance Monitor, perform the following optional task.

SUMMARY STEPS

1. **enable**

2. **show performance monitor cache** [**policy** *policy-map-name* **class** *class-map-name*][**interface** *interface name*]
3. **show performance monitor clients detail all**

DETAILED STEPS

Step 1 enable

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 show performance monitor cache [policy *policy-map-name* class *class-map-name*][interface *interface name*]

Example:

```
MMON Metering Layer Stats:
  static pkt cnt: 3049
  static cce sb cnt: 57
  dynamic pkt cnt: 0
  Cache type:                               Permanent
  Cache size:                                    2000
  Current entries:                                  8
  High Watermark:                                   9
  Flows added:                                      9
  Updates sent             (  1800 secs)            0
IPV4 SRC ADDR    IPV4 DST ADDR    IP PROT  TRNS SRC PORT  TRNS DST PORT
ipv4 ttl ipv4 ttl min ipv4 ttl max  ipv4 dscp bytes long perm pktslong perm  user space vm
==========================================================================================
10.1.1.1         10.1.2.3              17           4000           1967
0             0             0  0x00                          80
1 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           6000           1967
0             0             0  0x00                          80
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           4000           2000
0             0             0  0x00                          44
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3               6           6000           3000
```

```
0             0             0  0x00                          84
2  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           1967           6001
0             0             0  0x00                          36
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           1967           4001
0             0             0  0x00                          36
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3               6           3001           6001
0             0             0  0x00                         124
3  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           2001           4001
0             0             0  0x00                          44
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
```

Step 3 **show performance monitor clients detail all**

Example:

```
Client name for ID 1 : Mediatrace-131419052
   Type: Mediatrace
   Age: 443 seconds
   Monitor Object: _MMON_DYN_-class-map-69
        Flow spec: (dvmc-acl#47) 10.10.130.2 1000 10.10.132.2 2000 17
        monitor parameters
                interval duration 60
                timeout 2
                history 1
                flows 100
        monitor metric rtp
                min-sequential 10
                max-dropout 5
                max-reorder 5
                clock-rate 112 90000
```

```
        clock-rate default 90000
        ssrc maximum 20
monitor metric ip-cbr
        rate layer3 packet 20
Flow record: dvmc_fnf_fdef_47
        Key fields:
                ipv4 source address
                ipv4 destination address
                transport source-port
                transport destination-port
                ip protocol
        Non-key fields:
                monitor event
                application media event
                routing forwarding-status
                ip dscp
                ip ttl
                counter bytes rate
                application media bytes rate
                transport rtp jitter mean
                transport packets lost counter
                transport packets expected counter
                transport event packet-loss counter
                transport packets lost rate
                timestamp interval
                counter packets dropped
                counter bytes
                counter packets
                application media bytes counter
                application media packets counter
Monitor point: _MMON_DYN_-policy-map-70 GigabitEthernet0/3 output
Classification Statistic:
        matched packet: 545790
        matched byte: 64403220
```

Displaying the Clock Rate for Cisco Performance Monitor Classes

To display the clock rate for one or more classes, perform the following optional task.

SUMMARY STEPS

1. **enable**
2. **show performance monitor clock rate** [**policy** *policy-map-name* **class** *class-map-name*]

DETAILED STEPS

Step 1 enable

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 show performance monitor clock rate [policy *policy-map-name* class *class-map-name*]

If no class name is specified, information for all classes are displayed.

Example:

```
Device# show performance monitor clock rate policy all-apps class telepresence-CS4
Load for five secs: 6%/2%; one minute: 5%; five minutes: 5% Time source is NTP, 17:41:35.508 EST Wed
 Feb 16 2011
RTP clock rate for Policy: all-apps, Class: telepresence-CS4
     Payload type     Clock rate(Hz)
     pcmu    (0  )     8000
     gsm     (3  )     8000
     g723    (4  )     8000
     dvi4    (5  )     8000
     dvi4-2  (6  )     16000
     lpc     (7  )     8000
     pcma    (8  )     8000
     g722    (9  )     8000
     l16-2   (10 )     44100
     l16     (11 )     44100
     qcelp   (12 )     8000
     cn      (13 )     8000
     mpa     (14 )     90000
     g728    (15 )     8000
     dvi4-3  (16 )     11025
     dvi4-4  (17 )     22050
     g729    (18 )     8000
     celb    (25 )     90000
     jpeg    (26 )     90000
     nv      (28 )     90000
     h261    (31 )     90000
     mpv     (32 )     90000
     mp2t    (33 )     90000
     h263    (34 )     90000
             (96 )     48000
             (112)     90000
     default           90000
```

Displaying the Current Status of a Flow Monitor

To display the current status of a flow monitor, perform the following optional task.

Before you begin

The interface to which you applied the input flow monitor must be receiving traffic that meets the criteria defined by the original flow record before you can display the flows in the flow monitor cache.

SUMMARY STEPS

1. **enable**
2. **show flow monitor type performance-monitor**

DETAILED STEPS

Step 1 **enable**

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 **show flow monitor type performance-monitor**

The **show flow monitor type performance-monitor** command shows the current status of the flow monitor that you specify.

Example:

```
Device# show flow monitor type performance-monitor
Flow Monitor type performance-monitor monitor-4:
  Description:           User defined
  Flow Record:           record-4
  Flow Exporter:         exporter-4
  No. of Inactive Users: 0
  No. of Active Users:   0
```

Verifying the Flow Monitor Configuration

To verify the configuration commands that you entered, perform the following optional task.

Before you begin

The interface to which you applied the input flow monitor must be receiving traffic that meets the criteria defined by the original flow record before you can display the flows in the flow monitor cache.

SUMMARY STEPS

1. **enable**
2. **show running-config flow monitor**

DETAILED STEPS

Step 1 **enable**

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 **show running-config flow monitor**

The **show running-config flow monitor** command shows the configuration commands of the flow monitor that you specify.

Example:

```
Device# show running-config flow monitor
Current configuration:
!
flow monitor FLOW-MONITOR-1
 description Used for basic IPv4 traffic analysis
 record netflow ipv4 original-input
!
!
flow monitor FLOW-MONITOR-2
 description Used for basic IPv6 traffic analysis
 record netflow ipv6 original-input
!
```

Verifying That Cisco IOS Flexible NetFlow and Cisco Performance Monitor Is Enabled on an Interface

To verify that Flexible NetFlow and Cisco Performance Monitor is enabled on an interface, perform the following optional task.

SUMMARY STEPS

1. **enable**
2. **show flow interface** *type number*

DETAILED STEPS

Step 1 enable

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Router> enable
Router#
```

Step 2 show flow interface *type number*

The **show flow interface** command verifies that Flexible NetFlow and Cisco Performance Monitor is enabled on an interface.

Example:

```
Router# show flow interface ethernet 0/0
Interface Ethernet0/0
  FNF:  monitor:          FLOW-MONITOR-1
        direction:        Input
        traffic(ip):      on
  FNF:  monitor:          FLOW-MONITOR-2
        direction:        Input
        traffic(ipv6):    on
```

Displaying the Flow Monitor Cache

To display the data in the flow monitor cache, perform the following optional task.

Before you begin

The interface to which you applied the input flow monitor must be receiving traffic that meets the criteria defined by the original flow record before you can display the flow data in the flow monitor cache.

SUMMARY STEPS

1. **enable**
2. **show flow monitor name** *monitor-name* **cache format record**

DETAILED STEPS

Step 1 enable

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 show flow monitor name *monitor-name* cache format record

The **show flow monitor name** *monitor-name* **cache format record** command string displays the status, statistics, and the flow data in the cache for a flow monitor.

Example:

```
Device# show flow monitor name FLOW-MONITOR-1 cache format record
Cache type:                               Normal
  Cache size:                               4096
  Current entries:                             8
  High Watermark:                              8
  Flows added:                                24
  Flows aged:                                 16
    - Active timeout    (  1800 secs)          0
    - Inactive timeout (    15 secs)          16
    - Event aged                               0
    - Watermark aged                           0
    - Emergency aged                           0
IPV4 SOURCE ADDRESS:       10.251.10.1
IPV4 DESTINATION ADDRESS:  172.16.10.2
TRNS SOURCE PORT:          0
TRNS DESTINATION PORT:     2048
INTERFACE INPUT:           Et0/0
FLOW SAMPLER ID:           0
IP TOS:                    0x00
IP PROTOCOL:               1
ip source as:              0
ip destination as:         0
ipv4 next hop address:     172.16.7.2
ipv4 source mask:          /0
ipv4 destination mask:     /24
tcp flags:                 0x00
```

```
interface output:          Et1/0
counter bytes:             733500
counter packets:           489
timestamp first:           720892
timestamp last:            975032
.
.
.
IPV4 SOURCE ADDRESS:       172.16.6.1
IPV4 DESTINATION ADDRESS:  224.0.0.9
TRNS SOURCE PORT:          520
TRNS DESTINATION PORT:     520
INTERFACE INPUT:           Et0/0
FLOW SAMPLER ID:           0
IP TOS:                    0xC0
IP PROTOCOL:               17
ip source as:              0
ip destination as:         0
ipv4 next hop address:     0.0.0.0
ipv4 source mask:          /24
ipv4 destination mask:     /0
tcp flags:                 0x00
interface output:          Null
counter bytes:             52
counter packets:           1
timestamp first:           973804
timestamp last:            973804
Device# show flow monitor name FLOW-MONITOR-2 cache format record
Cache type:                              Normal
  Cache size:                              4096
  Current entries:                            6
  High Watermark:                             8
  Flows added:                             1048
  Flows aged:                              1042
    - Active timeout   (  1800 secs)         11
    - Inactive timeout (    15 secs)       1031
    - Event aged                             0
    - Watermark aged                         0
    - Emergency aged                         0
IPV6 FLOW LABEL:           0
IPV6 EXTENSION MAP:        0x00000040
IPV6 SOURCE ADDRESS:       2001:DB8:1:ABCD::1
IPV6 DESTINATION ADDRESS:  2001:DB8:4:ABCD::2
TRNS SOURCE PORT:          3000
TRNS DESTINATION PORT:     55
INTERFACE INPUT:           Et0/0
FLOW DIRECTION:            Input
FLOW SAMPLER ID:           0
IP PROTOCOL:               17
IP TOS:                    0x00
ip source as:              0
ip destination as:         0
ipv6 next hop address:     ::
ipv6 source mask:          /48
ipv6 destination mask:     /0
tcp flags:                 0x00
interface output:          Null
counter bytes:             521192
counter packets:           9307
timestamp first:           9899684
timestamp last:            11660744
.
.
.
```

```
IPV6 FLOW LABEL:           0
IPV6 EXTENSION MAP:        0x00000000
IPV6 SOURCE ADDRESS:       FE80::A8AA:BBFF:FEBB:CC03
IPV6 DESTINATION ADDRESS:  FF02::9
TRNS SOURCE PORT:          521
TRNS DESTINATION PORT:     521
INTERFACE INPUT:           Et0/0
FLOW DIRECTION:            Input
FLOW SAMPLER ID:           0
IP PROTOCOL:               17
IP TOS:                    0xE0
ip source as:              0
ip destination as:         0
ipv6 next hop address:     ::
ipv6 source mask:          /10
ipv6 destination mask:     /0
tcp flags:                 0x00
interface output:          Null
counter bytes:             92
counter packets:           1
timestamp first:           11653832
timestamp last:            11653832
```

Displaying the Current Status of a Flow Exporter

To display the current status of a flow exporter, perform the following optional task.

SUMMARY STEPS

1. **enable**
2. **show flow exporter** [*exporter-name*]

DETAILED STEPS

Step 1 enable

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 show flow exporter [*exporter-name*]

The **show flow exporter** command shows the current status of the flow exporter that you specify.

Example:

```
Device# show flow exporter EXPORTER-1
Flow Exporter EXPORTER-1:
  Description:              Exports to Chicago datacenter
  Transport Configuration:
    Destination IP address: 172.16.10.2
    Source IP address:      172.16.7.1
    Transport Protocol:     UDP
```

```
Destination Port:       65
Source Port:            56041
DSCP:                   0x0
TTL:                    255
```

Verifying the Flow Exporter Configuration

To verify the configuration commands that you entered to configure the flow exporter, perform the following optional task.

SUMMARY STEPS

1. **enable**
2. **show running-config flow exporter** *exporter-name*

DETAILED STEPS

Step 1 **enable**

The **enable** command enters privileged EXEC mode (enter the password if prompted).

Example:

```
Device> enable
Device#
```

Step 2 **show running-config flow exporter** *exporter-name*

The **show running-config flow exporter** command shows the configuration commands of the flow exporter that you specify.

Example:

```
Device# show running-config flow exporter EXPORTER-1
Building configuration...
!
flow exporter EXPORTER-1
 description Exports to datacenter
 destination 172.16.10.2
 transport udp 65
!
```

Enabling Debugging

To enable debugging for Cisco Performance Monitor, perform the following optional task in privileged EXEC mode.

SUMMARY STEPS

1. **debug performance monitor** {**database** | **dynamic** | **event** | **export** | **flow-monitor** | **metering** | **provision** | **sibling** | **snmp** | **tca** | **timer**}

DETAILED STEPS

debug performance monitor {**database** | **dynamic** | **event** | **export** | **flow-monitor** | **metering** | **provision** | **sibling** | **snmp** | **tca** | **timer**}

The **debug performance monitor** command enables debugging for the following performance monitor components:

- Flow database
- Dynamic monitoring
- Performance events
- Exporting
- Flow monitors
- Metering layer
- Provisioning
- Sibling management
- SNMP
- TCA
- Timers

The following example shows how to enable debugging for dynamic monitoring:

Example:

```
Device# debug performance monitor dynamic
```

Configuration Example for Cisco Performance Monitor

Example Monitor for Lost RTP Packets and RTP Jitter

This example show a configuration that monitors the number of lost RTP packets, the amount of RTP jitter, and other basic statistics for the **gig1** interface. In this example, Cisco Performance Monitor is also configured to make an entry in the syslog when the any of the following events occur on the interface:

- The percentage of lost RTP packets is between 5 percent and 9 percent.
- The percentage of lost RTP packets is greater than 10 percent.

- A media stop event has occurred.

```
! Set the filter spec for the flows to monitor.
access-list 101 ip permit host 10.10.2.20 any
! Use the flow record to define the flow keys and metric to collect.
flow record type performance-monitor video-monitor-record
 match ipv4 source
 match ipv4 destination
 match transport source-port
 match transport destination-port
 match rtp ssrc
 collect timestamp
 collect counter byte
 collect counter packet
 collect mse
 collect media-error
 collect counter rtp interval-jitter
 collect counter rtp packet lost
 collect counter rtp lost event
! Set the exporting server. The export message format is based on FNFv.9.
flow export video-nms-server
 export-protocol netflow-v9
 destination cisco-video-management
 transport udp 32001
! Set the flow filter in the class-map.
class-map match-all video-class
 access-group  ipv4 101
! Set the policy map with the type performance-monitor for video monitor.
policy-map type performance-monitor video-monitor
 ! Set the video monitor actions.
 class video-class
  ! Specify where the metric data is being exported to.
  export  flow video-nms-server
  flow monitor inline
   record video-monitor-record
! Set the monitoring modeling parameters.
monitor parameters
 ! Set the measurement timeout to 10 secs.
 interval duration 10
 ! Set the timeout to 10 minutes.
 timeout 10
 ! Specify that 30 flow intervals can be kept in performance database.
 history 30
 priority 7
 ! Set rtp flow verification criteria.
 monitor metric rtp
 ! Configure a RTP flow criteria: at least 10 packets in sequence.
 min-sequential   10
 ! Ignore packets that are more than 5 packet ahead in terms of seq  number.  max-dropout
 5
 ! Ignore packets that are more than 5 packets behind in terms of seq  number.
 max-reorder 5
 ! Set the clock rate frequency for rtp packet timestamp clock.
 clock-rate 89000
 ! Set the maximum number of ssrc allowed within this class.
 ssrc maximum  100
 ! Set TCA for alarm.
 react 100 transport-packets-lost-rate
  description critical  TCA
  ! Set the threshold to greater than 10%.
  threshold gt 10
  ! Set the threshold to the average number based on the last five intervals.
  threshold type average 5
```

```
 action  syslog
 alarm severity critical
react 110 transport-packets-lost-rate
 description medium TCA
 ! Set the threshold to between 5% and 9% of packet lost.
 threshold range gt 5 le 9
 threshold type average 10
 action  syslog
 alarm type grouped percent 30
react 3000 media-stop
 action syslog
 alarm severity critical
 alarm type grouped percent 30

interface gig1
 service-policy type performance-monitor video-mon in
```

Where to Go Next

For more information about configuring the products in the Medianet product family, see the other chapter in this guide or see the *Cisco Media Monitoring Configuration Guide*.

Additional References

Related Documents

Related Topic	Document Title
Design, configuration, and troubleshooting resources for Performance Monitor and other Cisco Medianet products, including a Quick Start Guide and Deployment Guide.	See the Cisco Medianet Knowledge Base Portal, located at http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html
IP addressing commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco Media Monitoring Command Reference*
Configuration commands for Flexible NetFlow	*Cisco IOS Flexible NetFlow Command Reference*
Overview of Flexible NetFlow	“Cisco IOS Flexible NetFlow Overview”
Flexible NetFlow Feature Roadmap	“Cisco IOS Flexible NetFlow Features Roadmap”
Configuring flow exporters to export Flexible NetFlow data.	“Configuring Data Export for Cisco IOS Flexible NetFlow with Flow Exporters”

Related Topic	Document Title
Customizing Flexible NetFlow	"Customizing Cisco IOS Flexible NetFlow Flow Records and Flow Monitors"
Configuring flow sampling to reduce the overhead of monitoring traffic with Flexible NetFlow	"Using Cisco IOS Flexible NetFlow Flow Sampling to Reduce the CPU Overhead of Analyzing Traffic"
Configuring Flexible NetFlow using predefined records	"Configuring Cisco IOS Flexible NetFlow with Predefined Records"
Using Flexible NetFlow Top N Talkers to analyze network traffic	"Using Cisco IOS Flexible NetFlow Top N Talkers to Analyze Network Traffic"
Configuring IPv4 multicast statistics support for Flexible NetFlow	"Configuring IPv4 Multicast Statistics Support for Cisco IOS Flexible NetFlow"

Standards

Standard	Title
None	—

MIBs

MIB	MIBs Link
• CISCO-FLOW-MONITOR-TC-MIB • CISCO-FLOW-MONITOR-MIB • CISCO-RTP-METRICS-MIB • CISCO-IP-CBR-METRICS-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 3954	*Cisco Systems NetFlow Services Export Version 9* http://www.ietf.org/rfc/rfc3954.txt
RFC 3550	*RTP: A Transport Protocol for Real-Time Applications* http://www.ietf.org/rfc/rfc3550.txt

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Cisco Performance Monitor

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

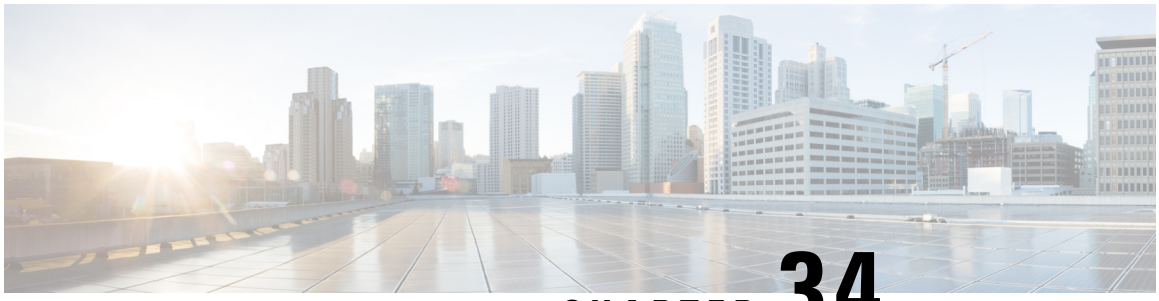
Table 45: Feature Information for Cisco Performance Monitor

Feature Name	Releases	Feature Information
Cisco Performance Monitor 1.0	15.1(3)T 12.2(58)SE 15.1(4)M1 15.0(1)SY Cisco IOS XE Release 3.5S 15.1(1)SG Cisco IOS XE Release 3.3 SG	This feature enables you to monitor the flow of packets in your network and bec of any issues that might impact the flow before it starts to significantly impact y applications' performance. Support for this feature was added for Cisco ASR 1000 Series Aggregation Servi in Cisco IOS XE Release 3.5S. There are some limitations to the monitoring of ingress or egress data on certain interfaces for the Cisco IOS XE Release 3.3 SG and Cisco IOS release 15.1(1)SG information, see the "Limitations" section. For all other releases, the following commands were introduced or modified by t **action**(policy react and policy inline react), **alarm severity** (policy react and pol react), **alarm type**(policy react and policy inline react), **class-map**, **clock-rate**(p **collect application media**, **clear fm performance-monitor counters**, **collect c collect flow direction**, **collect interface**, **collect ipv4**, **collect ipv4 destinatio ipv4 source**, **collect ipv4 ttl**, **collect monitor event**, **collect routing**, **collect t interval**, **collect transport event packet-loss counter**, **collect transport packe transport rtp jitter**, **debug fm performance-monitor counters**, **debug performance-monitor counters**, **description** (Performance Monitor), **destinat** (Flexible NetFlow), **export-protocol**, **exporter**, **flow monitor type performanc flow record type performance-monitor**, **flows**, **history** (monitor parameters), **duration**, **match access-group**, **match any**, **match class-map**, **match cos**, **r destination-address mac**, **match discard-class**, **match dscp**, **match flow**, **m match fr-dlci**, **match input-interface**, **match ip dscp**, **match ip precedence**, **rtp**, **match ipv4**, **match ipv4 destination**, **match ipv4 source**, **match mpls exp topmost**, **match not**, **match packet length** (class-map), **match precedence**, **r protocol**, **match qos-group**, **match source-address mac**, **match transport destination-port**, **match transport rtp ssrc**, **match transport source-port**, **m max-dropout** (policy RTP), **max-reorder** (policy RTP), **min-sequential** (policy **monitor metric ip-cbr**, **monitor metric rtp**, **monitor parameters**, **option** (F NetFlow), **output-features**, **platform performance-monitor rate-limit**, **policy performance-monitor**, **rate layer3**, **react** (policy), **record** (Performance Monito (policy), **service-policy type performance-monitor**, **show performance monit show performance monitor status**, **show platform hardware acl entry interf platform software ccm**, **show platform software feature-manager performanc show platform software feature-manager tcam**, **show policy-map type performance-monitor**, **snmp-server host**, **snmp-server enable traps flowmon**, **flowmon alarm history** , **source**(Flexible NetFlow), **ssrc maximum**, **templat timeout**, **threshold value** (policy react and policy inline react), **timeout** (monitor p **transport** (Flexible NetFlow), and **ttl** (Flexible NetFlow).

Feature Name	Releases	Feature Information
Cisco Performance Monitor (phase 2)	15.2(2)T Cisco IOS XE Release 3.5S	This feature enables you monitor IPv6 fields and also use all other Flexible Netflow and match commands not supported in the previous release. Flows are now correlated so that if the same policy is applied on the same input and interface, the show command will display a single flow for the input and output inte Support for this feature was added for Cisco ASR 1000 Series Aggregation Services in Cisco IOS XE Release 3.5S. The following commands were introduced or modified by this feature: **collect datalin collect ipv4 fragmentation**, **collect ipv4 section**, **collect ipv4 total-length**, **colle collect ipv6 destination**, **collect ipv6 extensionmap**, **collect ipv6 fragmentation**, **ipv6 hop-count**, **collect ipv6 length**, **collect ipv6 section**, **collect ipv6 source**, **c routing is-multicast**, **collect routing multicast replication-factor**, **collect timest sys-uptime**, **collect transport**, **collect transport icmp ipv4**, **collect transport icm collect transport tcp**, **collect transport udp**, **match application name**, **match con transaction-id**, **match datalink dot1q vlan**, **match datalink mac**, **match datalin match interface**, **match ipv4 fragmentation**, **match ipv4 section**, **match ipv4 total- match ipv4 ttl**, **match ipv6**, **match ipv6 destination**, **match ipv6 extension map**, **ipv6 fragmentation**, **match ipv6 hop-limit**, **match ipv6 length**, **match ipv6 sect match ipv6 source**, **match routing**, **match routing is-multicast**, **match routing mu replication-factor**, **match transport**, **match transport icmp ipv4**, **match transpo ipv6**, **match transport tcp**, **match transport udp**
Cisco Performance Monitor (phase 3)	15.2(3)T Cisco IOS XE Release 3.7S	This feature enables you to configure multiple exporters and monitor metadata field new TCP metrics. Support for this feature was added for Cisco ASR 1000 Series Aggregation Services in Cisco IOS XE Release 3.7S. The following commands were introduced or modified by this feature: **collect appli collect transport tcp bytes out-of-order**, **collect transport packets out-of-order**, **transport tcp maximum-segment-size**, **collect transport tcp window-size maxim collect transport tcp window-size minimum**, **collect transport tcp window-size a match application**, **match transport tcp bytes out-of-order**, **match transport pa out-of-order**, **match transport tcp maximum-segment-size**, **match transport tcp window-size maximum**, **match transport tcp window-size minimum**, **match tra tcp window-size average**
Performance Monitoring - IPv6 support	Cisco IOS XE Release 3.6S	This feature enables you to attach a monitor to IPv6 interfaces. Support for this feature was added for Cisco ASR 1000 Series Aggregation Services in Cisco IOS XE Release 3.6S.
Performance Monitoring - transport packet out of order	Cisco IOS XE Release 3.6S	This feature enables you to monitor the total number of out-of-order TCP packets. Support for this feature was added for Cisco ASR 1000 Series Aggregation Services in Cisco IOS XE Release 3.6S. The following commands were introduced or modified by this feature: **collect transp bytes out-of-order** and **collect transport packets out-of-order**.

Feature Name	Releases	Feature Information
Flexible NetFlow: IPFIX Export Format	15.2(4)M Cisco IOS XE Release 3.7S	Enables sending export packets using the IPFIX export protocol. The export of e fields from NBAR is only supported over IPFIX. Support for this feature was added for Cisco ASR 1000 Series Aggregation Servi in Cisco IOS XE Release 3.7S. The following command was introduced: **export-protocol**.
Flexible NetFlow: Export to an IPv6 Address	Cisco IOS XE Release 3.7S	This feature enables Flexible NetFlow to export data to a destination using an IP Support for this feature was added for Cisco ASR 1000 Series Aggregation Servi in Cisco IOS XE Release 3.7S. The following command was introduced: **destination**.
Flexible NetFlow: Extracted Fields Support	Cisco IOS XE Release 3.7S	Enables the collection of extracted fields using NBAR. The export of extracted fi supported over IPFIX. Support for this feature was added for Cisco ASR 1000 Series Aggregation Servi in Cisco IOS XE Release 3.7S. The following commands were introduced or modified by this feature: **collect h collect nntp group-name**, **collect pop3 server** , **collect rtsp host-name**, **collec destination**, **collect sip source**, **collect smtp server**, ,and **collect smtp sender**.

Feature Name	Releases	Feature Information
Application Visibility and Control (AVC) 2.0, which includes the following features: • Enable visualization of application usage under performance-monitoring policy • Enable performance of application usage • Enable Prime integration with router packet capture • Enable visualization of service path • FNF: Account On Resolution (AOR) for WAAS Segment • FNF: Account On Resolution (AOR) for performance monitoring policy-map	Cisco IOS XE Release 3.8S	AVC 2.0 provides extensive new functionality, including the integration of AVC wit Media Monitoring technology. This book only describes how to configure a flow record for AVC 2.0. For a comple explanation of AVC 2.0, see the *AVC Configuration Guide* at http://www.cisco.com/en/US/docs/ios-xml/ios/avc/configuration/xe-3s/avc-xe-3s-boo



CHAPTER 34

Metrics for Assurance Monitoring

Metrics for Assurance monitoring refers to Assurance-related metrics collected per network application, for flows forwarded through specific interfaces, to support Assurance monitoring by Cisco DNA Center. FNF provides a pair of record types (for IPv4 and IPv6) to collect this data. Monitoring for Assurance is optimized to provide better than typical performance for FNF monitors.

Feature Information for Metrics for Assurance Monitoring

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 46: Feature Information for Metrics for Assurance Monitoring

Feature Name	Releases	Feature Information
Metrics for Assurance Monitoring	Cisco IOS XE Gibraltar 16.10.1	FNF provides a pair of record types to collect data for Assurance, optimized to provide better than typical performance for FNF monitors.

Information About Metrics for Assurance Monitoring

Overview

DNA Center Assurance

Cisco DNA Center Assurance collects and analyzes network data to help provide better and more consistent network performance. DNA Center uses Flexible NetFlow (FNF) to collect specific network metrics for Assurance, providing quantitative and qualitative information about devices in the network. The FNF records designed for Assurance-related metrics are specially optimized for improved performance.

FNF provides a pair of record types (for IPv4 and IPv6) to collect data for Assurance. Monitoring Assurance metrics using these dedicated record types is optimized to provide better performance, as compared with typical FNF monitors configured to collect the same metrics. (Modifying the records cancels the dedicated performance enhancements for Assurance, and may prevent attaching a monitor to an interface.)

Manual Configuration

In typical use, DNA Center configures the monitors to collect data for Assurance, without requiring user input. However, it is also possible to use these record types manually.

Metrics Collected for Assurance

Most of the metrics collected for Assurance are metrics that have been available through FNF and other monitor types, but when they are collected specifically for Assurance records, some metrics may behave slightly differently.

Table 47: Metrics

Metric	Information
match ipv4/ipv6 version	IPv4/IPv6 version from IPv4/IPv6 header. [1]
match ipv4/ipv6 protocol	Layer4 protocol from the IPv4/IPv6 header.
match application name	Application ID
match connection client ipv4/ipv6 address	Field name: clientIPv4/IPv6Address IPv4/IPv6 client address in the IP packet header. The client is the device that triggered the session creation, and remains the same for the life of the session. [2]

Metric	Information
match connection server ipv4/ipv6 address	Field name: serverIPv4/IPv6Address IPv4/IPv6 server address in the IP packer header. The server is the device that replies to the client, and remains the same for the life of the session. [2]
match connection server transport port	Field name: serverTransportPort Server transport port identifier. This may be the source or destination transport port. The server is the device that replies to the client, and remains the same for the life of the session. [2]
match flow observation point	Field name: observationPointId Identifier of an observation point unique for each observation domain. [2]
collect connection initiator	Field name: biflowDirection Description of the direction assignment method used to assign the Biflow Source and Destination. [2]
collect flow direction	Direction (ingress/egress) of the flow observed at the observation point.
collect routing vrf input	Field name: ingressVRFID (Applies only to routers, not wireless controllers) VRF ID from incoming packets on a router. If a packet arrives on an interface that does not belong to a VRF, a VRF ID of 0 is recorded.
collect wireless client mac address	(Applies only to wireless controllers) Field name: staMacAddress The IEEE 802 MAC address of a wireless station (STA).
collect timestamp absolute first	Field name: flowStartMilliseconds The absolute timestamp of the first packet of the flow.
collect timestamp absolute last	Field name: flowEndMilliseconds The absolute timestamp of the last packet of the flow.
collect connection new-connections	Field name: connectionCountNew This information element counts the number of TCP or UDP connections which were opened during the observation period. The observation period may be specified by the flow start and end timestamps. [2]

Metric	Information
collect connection server counter packets long	Field name: serverPackets Number of layer 4 packets in a flow from the server. The server is the device that replies to the client, and remains the same for the life of the session. [2]
collect connection server counter bytes network long	Field name: serverOctets Overall IP packet bytes in a flow from the server. The server is the device that replies to the client, and remains the same for the life of the session. [2]
collect connection client counter packets long	Field name: clientPackets Number of layer 4 packets in a flow from the client. The client is the device that triggered the session creation, and remains the same for the life of the session. [2]
collect connection client counter bytes network long	Overall IP packet bytes from client to server. [2]
collect connection delay network client-to-server sum	Field name: sumNwkTime Network delay is the round-trip time between the client and the server, as measured by the observation point, calculated once per session. The value of this information element is the sum of all network delays observed for the sessions of this flow. [2] [3]
collect connection delay network to-server sum	Field name: sumServerNwkTime Server network delay is the round-trip time between the observation point and the server, calculated once per session. The value of this information element is the sum of all server network delays observed for the sessions of this flow. [2] [3]
collect connection client counter packets retransmitted	Field name: retransClientPackets Number of packets retransmitted by the client. [2] [3]
collect connection server counter packets retransmitted	Field name: retransServerPackets Number of packets retransmitted by the server. [3]

Metric	Information
collect connection delay application sum	Field name: sumServerRespTime The sum of all application delays observed for all responses of the flow. [2] [3]
collect connection server counter responses	Field name: numRespsCountDelta Total number of responses sent by the server. [2] [3]

Notes

[1] See Cisco IOS Flexible NetFlow Command Reference.
[2] See Cisco AVC Field Definition Guide.
[3] This metric can be used in Cisco Performance Monitor record types. It can be used with FNF only as part of the specially optimized Assurance-related records. Attempting to use this metric in a different FNF record type will cause the record to be rejected when attaching it to an interface.

How to Configure Metrics for Assurance Monitoring

Configuring Assurance Monitors Outside of DNA Center

In typical use, DNA Center configures the monitors without requiring additional user input, but it is possible to configure monitors for Assurance-related metrics manually.

Manual methods for monitoring Assurance-related metrics:

Method	Applicable to...	See section...
ezPM profile	Platforms that support ezPM Not wireless controllers	Configuring Assurance Monitors Using ezPM, on page 483
Pre-defined FNF records for Assurance	Routers Wireless controllers	Configuring Assurance Monitors Using Pre-defined FNF Records, on page 484

Configuring Assurance Monitors Using ezPM

Applicable to: routers, not wireless controllers

The application-assurance ezPM profile makes use of the application performance monitoring (APM) FNF records designed for Assurance-related metrics. Configuring APM with ezPM greatly simplifies the configuration, as compared with working with the FNF records directly.

1. Configure the ezPM context.

   performance monitor context *context-name* **profile application-assurance**

   traffic-monitor assurance-monitor ipv4

traffic-monitor assurance-monitor ipv6

2. Attach the context to an interface. The following attaches the performance monitor to an interface, monitoring both input and output.

   interface *interface*

   performance monitor context *context-name*

Result

This attaches monitors to the interface to collect Assurance-related metrics.

Example

In the following example, a monitor called apm is attached to the Gigabit Ethernet 1 interface.

```
performance monitor context apm profile application-assurance
traffic-monitor assurance-monitor ipv4
traffic-monitor assurance-monitor ipv6

interface GigabitEthernet1
performance monitor context apm
```

Configuring Assurance Monitors Using Pre-defined FNF Records

Applicable to: routers, wireless controllers

ezPM is the preferred method for configuring monitors for Assurance-related metrics, but it is also possible to use the FNF records pre-defined for these metrics. For platforms that do not support ezPM, this is the preferred method.

The FNF records designed for Assurance-related metrics are specially optimized for improved performance.

How to configure on a routing platform

Note Does not apply to wireless platforms.

1. Define two flow monitors for assurance-related metrics, one for IPv4 and one for IPv6.

   flow monitor *monitor-name-for-ipv4*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record netflow ipv4 assurance

   flow monitor *monitor-name-for-ipv6*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record netflow ipv6 assurance

2. Attach the context to an interface. The following attaches the performance monitor to an interface, monitoring both input and output.

   interface *interface*

ipv4 flow monitor *monitor-name-for-ipv4* **input**

ipv4 flow monitor *monitor-name-for-ipv4* **output**

ipv6 flow monitor *monitor-name-for-ipv6* **input**

ipv6 flow monitor *monitor-name-for-ipv6* **output**

Result

This attaches two IPv4 and two IPv6 monitors to the interface for collecting the metrics that are needed for Assurance.

Example

This example defines monitors called assurance-ipv4 and assurance-ipv6, and attaches the monitors to the GigabitEthernet1 interface.

```
flow monitor assurance-ipv4
cache entries 100000
record netflow ipv4 assurance

flow monitor assurance-ipv6
cache entries 100000
record netflow ipv6 assurance

interface GigabitEthernet1
ipv4 flow monitor assurance-ipv4 input
ipv4 flow monitor assurance-ipv4 output
ipv6 flow monitor assurance-ipv6 input
ipv6 flow monitor assurance-ipv6 output
```

How to configure on a wireless platform

Note Does not apply to routing platforms.

1. Enter the configuration mode for the relevant wireless profile.

   interface *policy-name*

2. Define two monitors for the wireless controller, one for IPv4 and one for IPv6.

   flow monitor *monitor-name-wlc-for-ipv4*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record wireless avc ipv4 assurance

   flow monitor monitor-name-wlc-ipv6

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record wireless avc ipv6 assurance

3. Attach the two flow monitors to the wireless profile, including input and output traffic.

   wireless profile policy *policy-name*

ipv4 flow monitor *monitor-name-for-wireless-ipv4* **input**

ipv4 flow monitor *monitor-name-for-wireless-ipv4* **output**

ipv6 flow monitor monitor-name-for-wireless-ipv6 **input**

ipv6 flow monitor *monitor-name-for-wireless-ipv6* **output**

Example

This example defines monitors called assurance-wlc-ipv4 and assurance-wlc-ipv6, and attaches the monitors to a wireless profile.

```
flow monitor assurance-wlc-ipv4
cache entries 100000
record wireless avc ipv4 assurance

flow monitor assurance-wlc-ipv6
cache entries 100000
record wireless avc ipv6 assurance

wireless profile policy AVC_POL
central association
central switching
ipv4 flow monitor assurance-wlc-ipv4 input
ipv4 flow monitor assurance-wlc-ipv4 output
ipv6 flow monitor assurance-wlc-ipv6 input
ipv6 flow monitor assurance-wlc-ipv6 output
no shutdown
```

About Attaching the Assurance Monitors to Interfaces

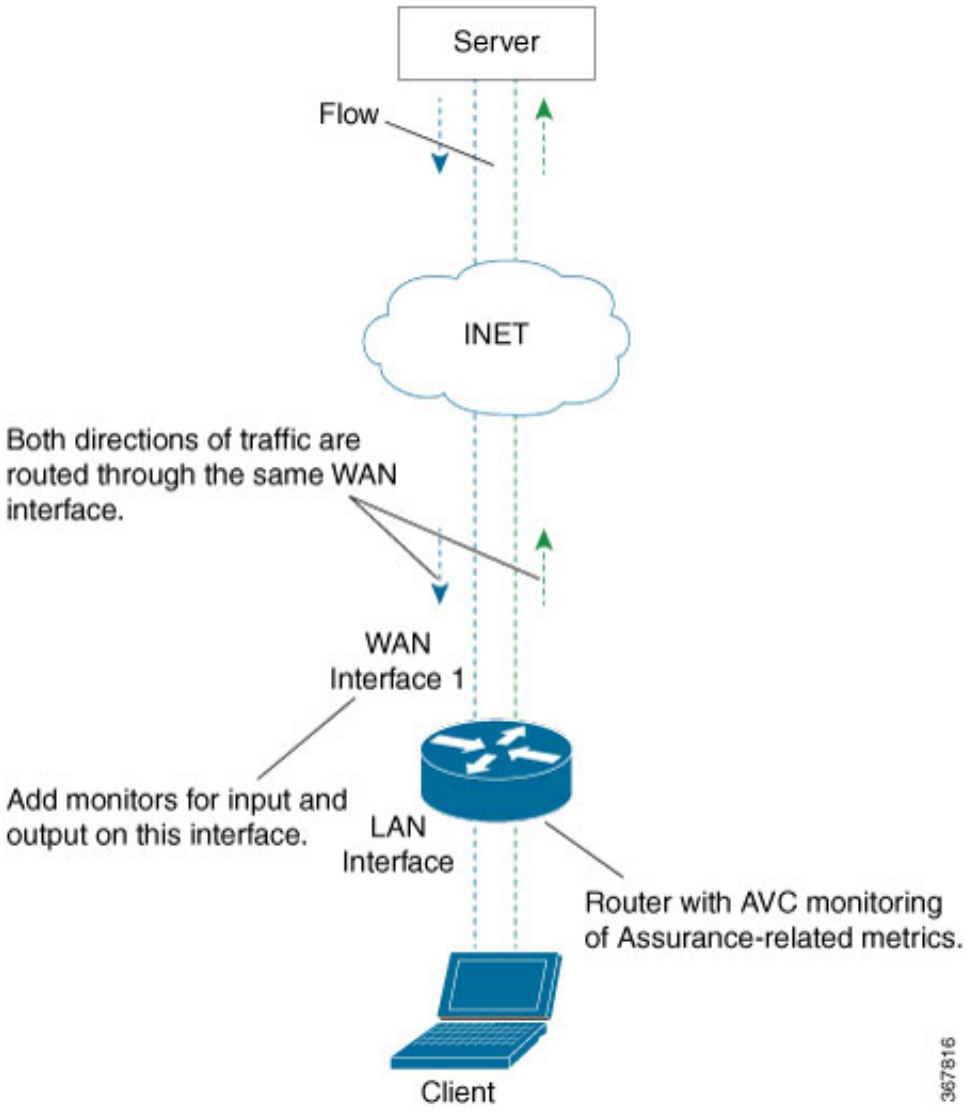
Monitor a Flow on Only One Interface

Monitors for Assurance-related metrics should only see a single flow one time. In the typical symmetric routing scenario, they should monitor the flow on only one interface.

Do not attach monitors for Assurance-related metrics to two separate interfaces that handle both directions of the same flow. Doing so will cause incorrect traffic metrics to be reported. For example, if traffic enters a device on interface A and leaves on interface B, do not attach monitors for Assurance-related metrics to both interfaces A and B.

Typical symmetric routing, with monitors for input and output on the same interface:

Figure 12: Symmetric Routing

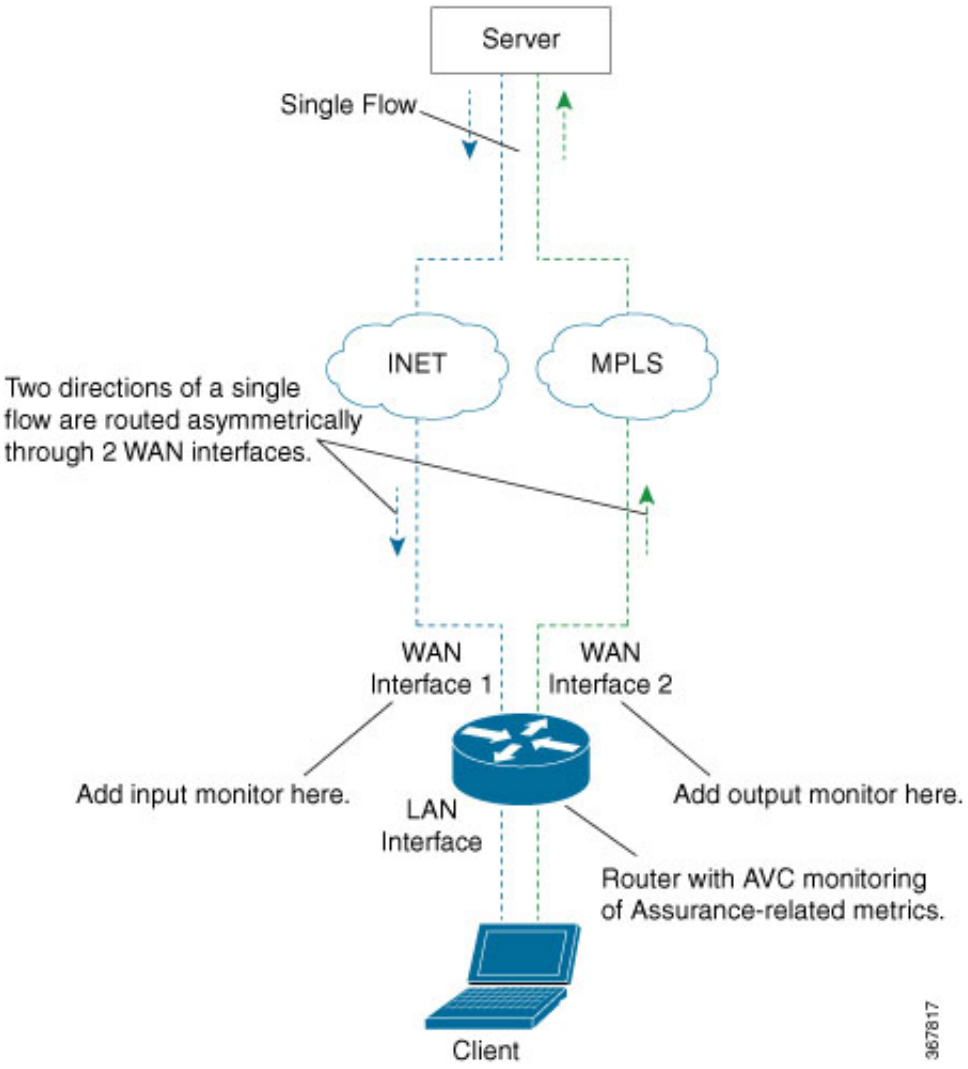


Asymmetric Routing

In some cases, such as for asymmetric routing, it might be necessary to attach a monitor for input on one interface, and a monitor for output on another interface.

In some scenarios, a single flow may be routed asymmetrically, with upstream and downstream traffic for the flow occurring on two different interfaces. In this case, place monitors for input and output on two separate interfaces to monitor the complete flow.

Figure 13: Asymmetric Routing



Viewing Details of Assurance Records and Contexts

Overview

After you attach a context to an interface, two **show** commands can be used to display information about Assurance records or about contexts.

Displaying Structure of the Assurance Record

The following command displays the structure of the pre-defined Assurance records (IPv4 and IPv6):

show fnf record netflow {**ipv4** | **ipv6**} **assurance**

Displaying Configuration of a Context

The following command displays the full configuration of a specified context.

show performance monitor context *context-name* **configuration**

The following output shows the Assurance-related monitoring through an ezPM context called ApmContext, attached to a router interface.

```
Device#show performance monitor context ApmContext configuration
!================================================================================
!                Equivalent Configuration of Context ApmContext                !
!================================================================================
!Exporters
!==========
!
flow exporter ApmContext-1
description performance monitor context ApmContext exporter
destination 64.103.113.128 vrf FNF
source GigabitEthernet2/2/0
transport udp 2055
export-protocol ipfix
template data timeout 300
option interface-table timeout 300
option vrf-table timeout 300
option sampler-table timeout 300
option application-table timeout 300
option application-attributes timeout 300
!
!Access Lists
!=============
!Class-maps
!===========
!Samplers
!=========
!Records and Monitors
!=====================
!
flow record ApmContext-app_assurance_ipv4
description ezPM record
match ipv4 version
match ipv4 protocol
match application name
match connection client ipv4 address
match connection server ipv4 address
match connection server transport port
match flow observation point
collect routing vrf input
collect flow direction
collect timestamp absolute first
collect timestamp absolute last
collect connection initiator
collect connection new-connections
collect connection server counter responses
collect connection delay network to-server sum
collect connection client counter packets retransmitted
collect connection delay network client-to-server sum
collect connection delay application sum
collect connection server counter packets long
collect connection client counter packets long
collect connection server counter packets retransmitted
collect connection server counter bytes network long
collect connection client counter bytes network long
!
```

```
!
flow monitor ApmContext-app_assurance_ipv4
description ezPM monitor
exporter ApmContext-1
cache timeout active 60
cache entries 100000
record ApmContext-app_assurance_ipv4
!
!
flow record ApmContext-app_assurance_ipv6
description ezPM record
match ipv6 version
match ipv6 protocol
match application name
match connection client ipv6 address
match connection server transport port
match connection server ipv6 address
match flow observation point
collect routing vrf input
collect flow direction
collect timestamp absolute first
collect timestamp absolute last
collect connection initiator
collect connection new-connections
collect connection server counter responses
collect connection delay network to-server sum
collect connection client counter packets retransmitted
collect connection delay network client-to-server sum
collect connection delay application sum
collect connection server counter packets long
collect connection client counter packets long
collect connection server counter packets retransmitted
collect connection server counter bytes network long
collect connection client counter bytes network long
!
!
flow monitor ApmContext-app_assurance_ipv6
description ezPM monitor
exporter ApmContext-1
cache timeout active 60
cache entries 100000
record ApmContext-app_assurance_ipv6
!
!Interface Attachments
!=====================
interface TenGigabitEthernet2/0/0
ip flow monitor ApmContext-app_assurance_ipv4 input
ip flow monitor ApmContext-app_assurance_ipv4 output
ipv6 flow monitor ApmContext-app_assurance_ipv6 input
ipv6 flow monitor ApmContext-app_assurance_ipv6 output
```

Notes and Limitations

Assurance-related Metrics and Elephant Flows

In networking, especially long flows are termed, “elephant flows,” and can pose a challenge to networking resources.

In a case where a single high-burst flow consumes too many QFP resources, the monitor collecting Assurance metrics might stop collecting qualitative metrics for the flow, to preserve resources for other traffic. No other traffic is affected.

Quantitative metrics are collected fully:

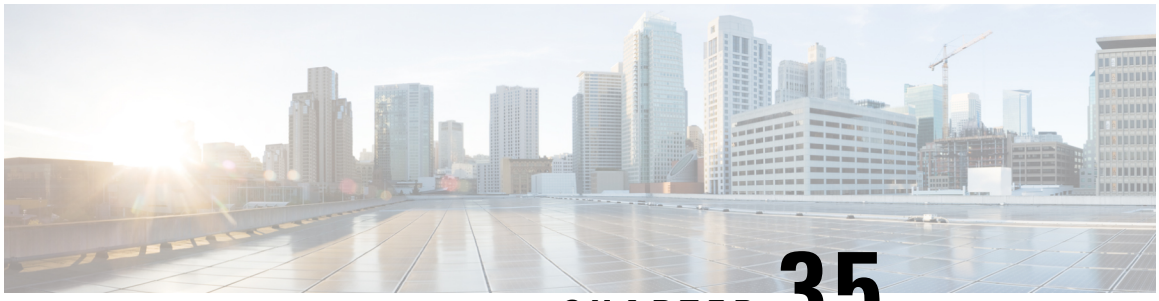
- Flow packets start time
- Flow packet end time
- Packets
- Bytes

Qualitative metrics are not collected fully:

- Total network delay sum (in the TCP handshake)
- Network to-server delay sum (in the TCP handshake)
- Client packets retransmitted
- Server packets retransmitted
- Application delay sum
- Number of server application responses

PART VIII

Embedded Event Manager



CHAPTER 35

Embedded Event Manager Overview

Embedded Event Manager (EEM) is a distributed and customized approach to event detection and recovery offered directly in a Cisco IOS device. EEM offers the ability to monitor events and take informational, corrective, or any desired EEM action when the monitored events occur or when a threshold is reached. An EEM policy is an entity that defines an event and the actions to be taken when that event occurs.

This module contains a technical overview of EEM. EEM can be used alone, or with other network management technologies to help monitor and maintain your network. Before you begin to implement EEM, it is important that you understand the information presented in this module.

Information About Embedded Event Manager

Embedded Event Manager

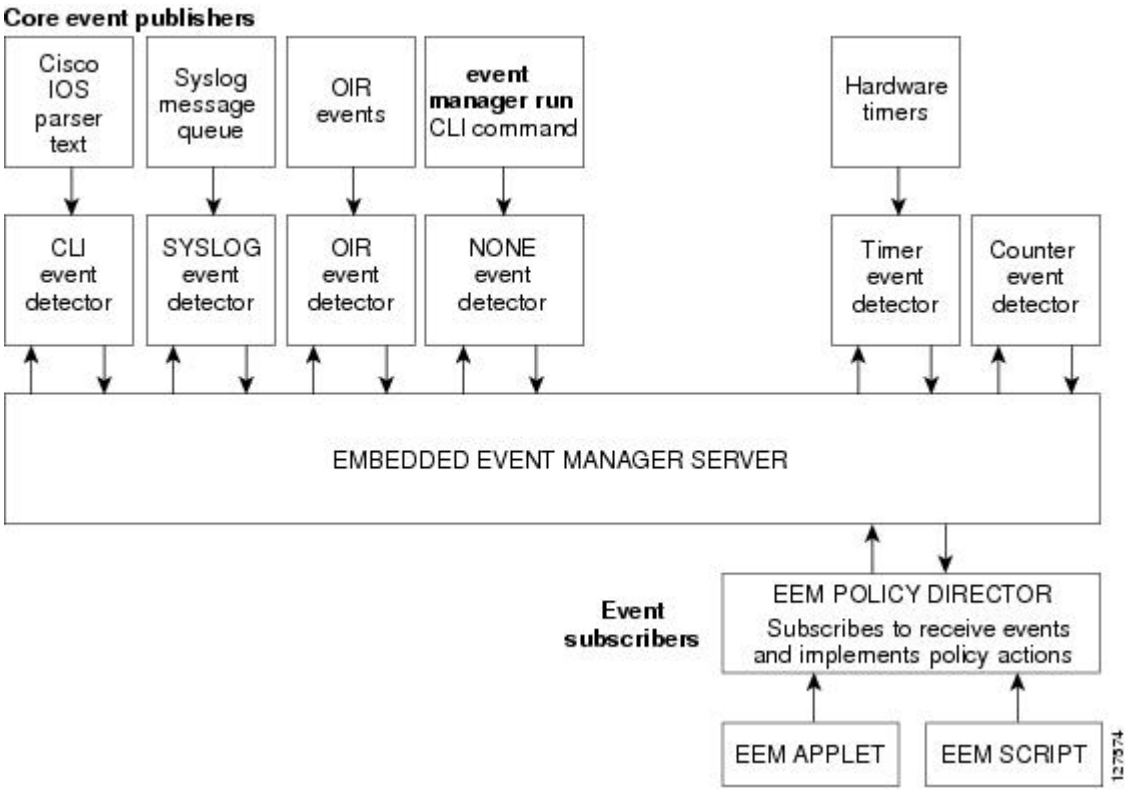
Event tracking and management has traditionally been performed by devices external to the networking device. Embedded Event Manager (EEM) has been designed to offer event management capability directly in Cisco IOS devices. The on-device, proactive event management capabilities of EEM are useful because not all event management can be done off device because some problems compromise communication between the device and the external network management device. Capturing the state of the device during such situations can be invaluable in taking immediate recovery actions and gathering information to perform root-cause analysis. Network availability is also improved if automatic recovery actions are performed without the need to fully reboot the routing device.

EEM is a flexible, policy-driven framework that supports in-box monitoring of different components of the system with the help of software agents known as event detectors. The figure below shows the relationship between the EEM server, core event publishers (event detectors), and the event subscribers (policies). Basically, event publishers screen events and publish them when there is a match on an event specification that is provided by the event subscriber. Event detectors notify the EEM server when an event of interest occurs. The EEM policies that are configured using the Cisco command-line interface (CLI) then implement recovery on the basis of the current state of the system and the actions specified in the policy for the given event.

EEM offers the ability to monitor events and take informational or corrective action when the monitored events occur or when a threshold is reached. An EEM policy is an entity that defines an event and the actions

to be taken when that event occurs. There are two types of EEM policies: an applet or a script. An applet is a simple form of policy that is defined within the CLI configuration. A script is a form of policy that is written in Tool Command Language (Tcl).

Figure 14: Embedded Event Manager Core Event Detectors



Note If your network has a higher version of EEM, that version would include the previous releases of EEM version as well.

Embedded Event Manager 1.0

EEM 1.0 introduced Embedded Event Manager. EEM 1.0 introduced the following event detectors:

- SNMP—The Simple Network Management Protocol (SNMP) event detector allows a standard SNMP MIB object to be monitored and an event to be generated when the object matches specified values or crosses specified thresholds.
- Syslog—The syslog event detector allows for screening syslog messages for a regular expression pattern match.

EEM 1.0 introduced the following actions:

- Generating prioritized syslog messages.
- Generating a Cisco Networking Services (CNS) event for upstream processing by CNS devices.

- Reloading the Cisco software.
- Switching to a secondary processor in a fully redundant hardware configuration.

Embedded Event Manager 2.0

EEM 2.0 introduced some new features. EEM 2.0 introduced the following event detectors:

- Application-Specific—The application-specific event detector allows any Embedded Event Manager policy to publish an event.
- Counter—The counter event detector publishes an event when a named counter crosses a specified threshold.
- Interface Counter—The interface counter event detector publishes an event when a generic Cisco IOS interface counter for a specified interface crosses a defined threshold.
- Timer—The timer event detector publishes events for the following four different types of timers: absolute-time-of-day, countdown, watchdog, and CRON.
- Watchdog System Monitor (IOSWDSysMon)—The Cisco IOS watchdog system monitor event detector publishes an event when CPU or memory utilization for a Cisco IOS process crosses a threshold.

EEM 2.0 introduced the following actions:

- Setting or modifying a named counter.
- Publishing an application-specific event
- Generating an SNMP trap.

The ability to run a Cisco defined sample policy written using Tool Command Language (Tcl) was introduced. A sample policy was provided that could be stored in the system policy directory.

Embedded Event Manager 2.1

EEM 2.1 and introduced some new features. EEM 2.1 introduced the following new event detectors:

- CLI—The CLI event detector screens command-line interface (CLI) commands for a regular expression match.
- None—The none event detector publishes an event when the Cisco IOS **event manager run** command executes an EEM policy.
- OIR—The online insertion and removal (OIR) event detector publishes an event when a particular hardware insertion or removal event occurs.

EEM 2.1 introduced the following actions:

- Executing a Cisco CLI command.
- Requesting system information when an event occurs.
- Sending a short e-mail.
- Manually running an EEM policy.

EEM 2.1 also permits multiple concurrent policies to be run using the new **event manager scheduler script** command. Support for SNMP event detector rate-based events is provided as is the ability to create policies using Tool Command Language (Tcl).

Embedded Event Manager 2.1 (Software Modularity)

EEM 2.1 (Software Modularity) is supported on Cisco Software Modularity images. EEM 2.1 (Software Modularity) introduced the following event detectors:

- GOLD—The Generic Online Diagnostic (GOLD) event detector publishes an event when a GOLD failure event is detected on a specified card and subcard.
- System Manager—The system manager event detector generates events for Cisco IOS Software Modularity process start, normal or abnormal stop, and restart events. The events generated by the system manager allows policies to change the default behavior of the process restart.
- Watchdog System Monitor (WDSysMon)—The Cisco Software Modularity watchdog system monitor event detector detects infinite loops, deadlocks, and memory leaks in Cisco IOS Software Modularity processes.

EEM 2.1 for Software Modularity introduced the ability to display EEM reliability metric data for processes.

Note EEM 2.1 for Software Modularity images also supports the resource and RF event detectors introduced in EEM 2.2, but it does not support the enhanced object tracking event detector or the actions to read and set tracked objects.

Embedded Event Manager 2.2

EEM 2.2 introduced some new features. EEM 2.2 introduced the following event detectors:

- Enhanced Object Tracking—The enhanced object tracking event detector publishes an event when the tracked object changes. Enhanced object tracking provides complete separation between the objects to be tracked and the action to be taken by a client when a tracked object changes.
- Resource—The resource event detector publishes an event when the Embedded Resource Manager (ERM) reports an event for the specified policy.
- RF—The redundancy framework (RF) event detector publishes an event when one or more RF events occur during synchronization in a dual Route Processor (RP) system. The RF event detector can also detect an event when a dual RP system continuously switches from one RP to another RP (referred to as a ping-pong situation).

EEM 2.2 introduced the following actions:

- Reading the state of a tracked object.
- Setting the state of a tracked object.

Embedded Event Manager 2.3

EEM 2.3 is supported on the Cisco Catalyst 6500 Series switches and introduces enhancements to the Generic Online Diagnostics (GOLD) Event Detector on that product.

- The **event gold** command was enhanced with the addition of the **action-notify**, **testing-type**, **test-name**, **test-id**, **consecutive-failure**, **platform-action**, and **maxrun** keywords for improved reaction to GOLD test failures and conditions.
- The following platform-wide GOLD Event Detector information can be accessed through new read-only EEM built-in environment variables:
  - Boot-up diagnostic level
  - Card index, name, serial number
  - Port counts
  - Test counts
- The following test-specific GOLD Event Detector information can be accessed through new read-only EEM built-in environment variables (available to EEM applets only):
  - Test name, attribute, total run count
  - Test result per test, port, or device
  - Total failure count, last fail time
  - Error code
  - Occurrence of consecutive failures

These enhancements result in reduced mean time to recovery (MTTR) and higher availability through improved automation and fault detection.

Embedded Event Manager 2.4

EEM 2.4 introduced the following event detectors:

- SNMP Notification—The SNMP notification event detector provides the ability to intercept SNMP trap and inform messages coming into the device. An SNMP notification event is generated when an incoming SNMP trap or inform message matches specified values or crosses specified thresholds.
- RPC—The remote procedure call (RPC) event detector provides the ability to invoke EEM policies from outside the device over an encrypted connection using Secure Shell (SSH). The RPC event detector uses Simple Object Access Protocol (SOAP) data encoding for exchanging XML-based messages. This event detector can be used to run EEM policies and then receive output in a SOAP XML-formatted reply.

EEM 2.4 added enhancements to the following event detectors:

- Interface counter rate-based trigger—This feature adds the ability for an interface event to be triggered based on a rate of change over a period of time. A rate can be specified both for the entry value and the exit value. This feature copies the rate-based functionality that currently exists for the SNMP event detector.
- SNMP delta value—The difference between the monitored Object Identifier (OID) value at the beginning of the monitored period and the actual OID value when the event is published will be provided in the **event reqinfo** data for both the SNMP event detector and the Interface Counter event detector.

EEM 2.4 introduced the following actions:

- Multiple event support—The ability to run multiple events was introduced. and **show event manager** commands were enhanced to show multiple events.
- Support for parameters—The *parameter* argument has been added to the **event manager run** command. A maximum of 15 parameters can be used.
- Display of Job IDs and completion status--Some of the **show event manager** commands were enhanced to display Job IDs and completion status.
- Bytecode support—Tcl 8 defines a specialized bytecode language (BCL) and includes a just-in-time compiler that translates Tcl scripts to BCL. Byte sequence is executed by a "virtual machine," Tcl_ExecuteByteCode(), or TEBC for short, as often as needed. Currently EEM accepts file extensions, such as *.tcl for user policies and *.tm for system policies. Tcl standard extension for bytecode scripts are *.tbc. Now EEM will accept *.tbc as valid EEM policies.
- Registration substitution enhancement—Supports replacing multiple parameters in the event registration statement lines with a single environment variable.
- Tcl package support

Embedded Event Manager 3.0

EEM 3.0 introduces the following new event detectors:

- Custom CLI--The custom CLI event detector publishes an event to add and enhance existing CLI command syntax.
- Routing--The Routing event detector publishes an event when route entries change in the Routing Information Base (RIB).
- NetFlow-- The NetFlow event detector publishes an event when a NetFlow event is triggered.
- IP SLA--The IP SLA event detector publishes an event when an IP SLA reaction is triggered.

EEM 3.0 introduces the following features.

- Class-based scheduling--The EEM policies will be assigned a class using the **class** keyword when they are registered. EEM policies registered without a class will be assigned to the default class.
- High performance Tcl policies--Three new Tcl commands are introduced **event_completion**, **event_wait**, and **event_completion_with_wait**.
- Interactive cli support--The synchronous applets are enhanced to support interaction with the local console (TTY). Two new IOS commands, **action gets** and **action puts,** are introduced to allow users to enter and display input directly on the console.
- Variable logic for applets--The Variable Logic for EEM Applets feature adds the ability to apply conditional logic within EEM applets. Conditional logic introduces a control structure that can change the flow of actions within applets depending on conditional expressions.
- Digital signature support--A new API performs digital signature verification for a Tcl script to check it the script is signed by Cisco before execution.
- Support authenticating e-mail servers--The **action mail**command is modified to include an optional username and password.

- SMTP IPv6 support--The keyword **sourceaddr** is added in Tcl e-mail templates to specify either an IPv6 or IPv4 address.
- SNMP library extensions--The EEM applet **action info** and Tcl **sys_reqinfo_snmp**commands are enhanced to include functionality for SNMP getid, inform, trap, and set-type operations.
- SNMP Notification IPv6 support--IPv6 address is supported for the source and destination IP addresses.
- CLI Library XML-PI support--Provides a programmable interface which encapsulates IOS command-line interface (CLI) show commands in XML format in a consistent way across different Cisco products. Customers using XML-PI will be able to parse IOS show command output from within Tcl scripts using well-known keywords instead of having to depend on the use of regular expression support.

Embedded Event Manager 3.1

EEM 3.1 introduced one new event detector:

- SNMP Object--The Simple Network Management Protocol (SNMP) object trap event detector provides an extension to replace the value when an SNMP trap with the specified SNMP object ID (OID) is encountered on a specific interface or address.

EEM 3.1 added an enhancement to the following event detector:

- SNMP Notification--The SNMP notification event detector now can wait and intercept the outgoing SNMP traps and informs.

EEM 3.1 added enhancement to the following action:

- Specify facility--The **action syslog** command has been enhanced to specify syslog facility.

EEM 3.1 introduces the following features:

- Provides the ability to create a short description for the registered policy--A new **description** command has been introduced to register policies with a brief description in Cisco IOS CLI and Tcl policies. The **show event manager policy available** command and the **show event manager policy registered** command have been enhanced to add the **description** keyword to display the description of the registered applet.
- Enables EEM policies to bypass AAA authorization--The **event manager application** command has been enhanced to provide authorization and bypass keywords to disable AAA.
- Introduces CLI Library enhancements--Provides two new commands in the CLI library: **cli_run** and **cli_run_interactive**.

Embedded Event Manager 3.2

EEM 3.2 introduced the following new event detectors:

- Neighbor Discovery--Neighbor Discovery event detector provides the ability to publish a policy to respond to automatic neighbor detection when:
  - a Cisco Discovery Protocol (CDP) cache entry is added, deleted or updated.
  - a Link Layer Discovery Protocol (LLDP) cache entry is added, deleted, or updated.
  - an interface link status changes.

- an interface line status changes.

- Identity--Identity event detector generates an event when AAA authorization and authentication is successful, when failure occurs, or after normal user traffic on the port is allowed to flow.
- Mac-Address-Table--Mac-Address-Table event detector generates an event when a MAC address is learned in the MAC address table.

Note The Mac-Address-Table event detector is supported only on switch platforms and can be used only on Layer 2 interfaces where MAC addresses are learned. Layer 3 interfaces do not learn addresses and devices do not usually support the mac-address-table infrastructure needed to notify EEM of a learned MAC address.

EEM 3.2 also introduces new CLI commands to support the applets to work with the new event detectors.

Embedded Event Manager 4.0

EEM 4.0 introduces the following new features:

- EEM Email Action Enhancements
  - TLS support for SMTP mail actions—The new optional **secure** keyword is added to the **action mail** CLI with **tls** and **none** keyword options. There are no updates to the corresponding Tcl Policy.
  - Custom port for SMTP mail actions—The new optional **port** keyword is added to the **action mail** CLI. In the Tcl policy, the port number can be specified by adding a line to the e-mail template.
- EEM Security Enhancements
  - Checksum-based script integrity—Where digital signature is not supported or unavailable, users can still enforce some basic integrity check on the TCL policy by using the Unix command **openssl sha1**. The new optional **checksum**, **md5**, and **sha-1** keywords have been added to the **event manager policy** command.
  - Third-party digital signature support—Requires Tcl secure mode and a trustpoint to associate with the TCL scripts in order to verify the signature.
  - Script owner identification—If a policy is successfully registered with a digital signature, the owner (or signer) of the policy can be identified by using the **show event manager policy registered** command and checking the **Dsig** keyword in the show output.
  - Registration of remote Tcl policies—The new optional**remote** keyword has been added to the **event manager policy** command.
- EEM Resource Management
  - Resource consumption throttling—The new optional **resource-limit** keyword has been added to the **event manager scheduler** command.
  - Rate limiting of triggered policies per event—The new optional **rate-limit** keyword has been added to the **event syslog** command.
- EEM Usability Enhancements
  - File operations in EEM applet actions—The new CLI **action file** has been added to allow file selection.
  - New fields are added in EEM to track statistics of queue size, dropped events, and run-time using the **show event manager statistics** EXEC command. A set of new clear commands—**clear event**

manager detector counters and **clear event manager server counters** —are introduced to clear the event manager queue counters.

- EEM Event Detector Enhancements
  - CLI event detector enhancement—Provides the ability to detect the session where the user enters the event cli command. Four new keywords and built-in environmental variables—**username**, **host**, **privilege**, and **tty**— are added to the **event cli** applet and event_reqinfo array names to the **event_register_cli** event detector. The **show event manager detector** EXEC command has also been modified to reflect the enhancement.
  - Syslog event detector performance enhancement—Provides the option to perform string matching on specific log message fields. The four new keywords—**facility**, **mnemonic**, **sequence**, and **timestamp** keywords— are added to the **action syslog** command, **event syslog** command, and to the **event_register_syslog** event detector. The **show event manager detector** EXEC command has also been modified to reflect the enhancement.

EEM Event Detectors Available by Cisco IOS Release

EEM uses software programs known as event detectors to determine when an EEM event occurs. Some event detectors are available on every Cisco IOS release, but most event detectors have been introduced in a specific release. Use the table below to determine which event detectors are available in your specific Cisco IOS release. A blank entry (--) indicates that the event detector is not available: the text "Yes" indicates that the event detector is available. The event detectors shown in the table are supported in later releases of the same Cisco IOS release train. For more details on each event detector, see the Event Detectors concept in the "Embedded Event Manager Overview" module.

Table 48: Availability of Event Detectors by Cisco IOS Release

Event Detector	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS Software Modularity	12.2(33)SXH	12.4(20)T 12.2(33)SXI	12.4(22)T 12.2(33)SRE	15.0(1)M 15.1(3)T	15.1(2)SY	15 E XE 3E
Application-Specific	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
CLI	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	--	Yes
Counter	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Custom CLI	--	--	--	--	--	--	Yes	Yes	--	--
Enhanced Object Tracking	--	--	Yes	--	Yes	Yes	Yes	Yes	--	--
Environmental	--	--	--	--	--	--	--	--	--	Yes
GOLD	--	--	--	Yes	Yes	Yes	Yes	Yes	--	Yes
Identity	--	--	--	--	--	--	--	Yes	Yes	Yes
Interface Counter	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	--	Yes

Event Detector	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS Software Modularity	12.2(33)SXH	12.4(20)T 12.2(33)SXI	12.4(22)T 12.2(33)SRE	15.0(1)M 15.1(3)T	15.1(2)SY	15 E XE 3E
IPSLA	--	--	--	--	--	--	Yes	Yes	--	Yes
Mac-Address-Table	--	--	--	--	--	--	--	Yes	Yes	Yes
Neighbor Discovery	--	--	--	--	--	--	--	Yes	Yes	Yes
NF	--	--	--	--	--	--	Yes	Yes	--	--
None	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
OIR	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Resource	--	--	Yes	Yes	Yes	Yes	Yes	Yes	--	--
RF	--	--	Yes	Yes	Yes	Yes	Yes	Yes	--	Yes
Routing	--	--	--	--	--	--	Yes	Yes	--	Yes
RPC	--	--	--	--	--	Yes	Yes	Yes	Yes	--
SNMP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	--	Yes
SNMP Proxy	--	--	--	--	--	--	--	--	Yes	--
SNMP Notification	--	--	--	--	--	Yes	Yes	Yes	--	Yes
SNMP Object	--	--	--	--	--	--	--	Yes	--	Yes
Syslog	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
System Manager	--	--	--	Yes	Yes	Yes	Yes	Yes	Yes	--
Timer	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
IOSWDSysMon (Cisco IOS watchdog)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	--	Yes
WDSysMon (Cisco IOS Software Modularity watchdog)	--	--	--	Yes	--	--	--		--	--

Event Detectors

Embedded Event Manager (EEM) uses software programs known as *event detectors* to determine when an EEM event occurs. Event detectors are separate systems that provide an interface between the agent being monitored, for example Simple Network Management Protocol (SNMP), and the EEM policies where an action can be implemented. Some event detectors are available on every Cisco IOS release, but most event detectors have been introduced in a specific release. For details of which event detector is supported in each Cisco IOS release, see the EEM Event Detectors Available by Cisco IOS Release concept in the "Writing Embedded Event Manager Policies Using the Cisco IOS CLI" or the "Writing Embedded Event Manager Policies Using Tcl" modules. EEM contains the following event detectors.

Application-Specific Event Detector

The application-specific event detector allows any Embedded Event Manager policy to publish an event. When an EEM policy publishes an event it must use an EEM subsystem number of 798 with any event type. If an existing policy is registered for subsystem 798 and a specified event type, a second policy of the same event type will trigger the first policy to run when the specified event is published.

CLI Event Detector

The CLI event detector screens command-line interface (CLI) commands for a regular expression match. When a match is found, an event is published. The match logic is performed on the fully expanded CLI command after the command is successfully parsed and before it is executed. The CLI event detector supports three publish modes:

- Synchronous publishing of CLI events--The CLI command is not executed until the EEM policy exits, and the EEM policy can control whether the command is executed. The read/write variable, _exit_status, allows you to set the exit status at policy exit for policies triggered from synchronous events. If _exit_status is 0, the command is skipped, if _exit_status is 1, the command is run.
- Asynchronous publishing of CLI events--The CLI event is published, and then the CLI command is executed.
- Asynchronous publishing of CLI events with command skipping--The CLI event is published, but the CLI command is not executed.

Counter Event Detector

The counter event detector publishes an event when a named counter crosses a specified threshold. There are two or more participants that affect counter processing. The counter event detector can modify the counter, and one or more subscribers define the criteria that cause the event to be published. After a counter event has been published, the counter monitoring logic can be reset to start monitoring the counter immediately or it can be reset when a second threshold--called an exit value--is crossed.

Custom CLI Event Detector

The custom CLI event detector publishes an event to add and enhance existing CLI command syntax. When the special parser characters Tab, ? (question mark), and Enter are entered, the parser sends the input to the custom CLI event detector for processing. The custom CLI event detector then compares this input against registered strings to determine if this is a new or enhanced CLI command. Upon a match the custom CLI event detector takes appropriate actions, such as displaying help for the command if ? is entered, displaying the entire command if Tab is entered, or executing the command if Enter was entered. If a match does not occur, the parser regains control and processes the information as usual.

Enhanced Object Tracking Event Detector

The enhanced object tracking (EOT) event detector publishes an event when the status of a tracked object changes. Object tracking was first introduced into the Hot Standby Router Protocol (HSRP) as a simple tracking mechanism that allowed you to track the interface line-protocol state only. If the line-protocol state of the interface went down, the HSRP priority of the device was reduced, allowing another HSRP device with a higher priority to become active.

Object tracking was enhanced to provide complete separation between the objects to be tracked and the action to be taken by a client when a tracked object changes. Thus, several clients such as HSRP, VRRP, or GLBP can register their interest with the tracking process, track the same object, and each take different action when the object changes. Each tracked object is identified by a unique number that is specified on the tracking command-line interface (CLI). Client processes use this number to track a specific object. The tracking process periodically polls the tracked objects and notes any change of value. The changes in the tracked object are communicated to interested client processes, either immediately or after a specified delay. The object values are reported as either up or down.

Enhanced object tracking is now integrated with EEM to allow EEM to report on a status change of a tracked object and to allow enhanced object tracking to track EEM objects. A new type of tracking object--a stub object--is created. The stub object can be manipulated using the existing CLI commands that already allow tracked objects to be manipulated.

GOLD Event Detector

The GOLD event detector publishes an event when a GOLD failure event is detected on a specified card and subcard.

Interface Counter Event Detector

The interface counter event detector publishes an event when a generic Cisco IOS interface counter for a specified interface crosses a defined threshold. A threshold can be specified as an absolute value or an incremental value. If the incremental value is set to 50, for example, an event would be published when the interface counter increases by 50.

After an interface counter event has been published, the interface counter monitoring logic is reset using two methods. The interface counter is reset either when a second threshold--called an exit value--is crossed or when an elapsed period of time occurs.

IP SLA Event Detector

The IP SLA event detector publishes an event when an IP SLA reaction is triggered.

NetFlow Event Detector

The NetFlow event detector publishes an event when a NetFlow event is triggered.

None Event Detector

The none event detector publishes an event when the Cisco IOS **event manager run** CLI command executes an EEM policy. EEM schedules and runs policies on the basis of an event specification that is contained within the policy itself. An EEM policy must be identified and registered to be permitted to run manually before the **event manager run** command will execute.

OIR Event Detector

The online insertion and removal (OIR) event detector publishes an event when one of the following hardware insertion or removal events occurs:

- A card is removed.
- A card is inserted.

Route Processors (RPs), line cards, or feature cards can be monitored for OIR events.

Resource Event Detector

The resource event detector publishes an event when the Embedded Resource Manager (ERM) reports an event for the specified policy. The ERM infrastructure tracks resource depletion and resource dependencies across processes and within a system to handle various error conditions. The error conditions are handled by providing an equitable sharing of resources between various applications. The ERM framework provides a communication mechanism for resource entities and allows communication between these resource entities from numerous locations. The ERM framework also helps in debugging CPU and memory-related issues. The ERM monitors system resource usage to better understand scalability needs by allowing you to configure threshold values for resources such as the CPU, buffers, and memory. The ERM event detector is the preferred method for monitoring resources in Cisco software but the ERM event detector is not supported in Software Modularity images. For more details about ERM, go to “Embedded Resource Manager” module.

RF Event Detector

The redundancy framework (RF) event detector publishes an event when one or more RF events occur during synchronization in a dual Route Processor (RP) system. The RF event detector can also detect an event when a dual RP system continuously switches from one RP to another RP (referred to as a ping-pong situation).

RPC Event Detector

The remote procedure call (RPC) event detector provides the ability to invoke EEM policies from outside the device over an encrypted connection using Secure Shell (SSH). The RPC event detector uses Simple Object Access Protocol (SOAP) data encoding for exchanging XML-based messages. This event detector can be used to run EEM policies and then receive output in a SOAP XML-formatted reply.

Routing Event Detector

The routing event detector publishes an event when a route entry changes in the Routing Information Base (RIB).

SNMP Event Detector

The SNMP event detector allows a standard SNMP MIB object to be monitored and an event to be generated when the object matches specified values or crosses specified thresholds.

SNMP Notification Event Detector

The SNMP notification event detector provides the ability to intercept SNMP trap and inform messages coming into or going out of the device. An SNMP notification event is generated when an incoming or outgoing SNMP trap or inform message matches specified values or crosses specified thresholds. The SNMP event detector can wait and intercept the outgoing SNMP traps and informs.

SNMP Object Event Detector

The Simple Network Management Protocol (SNMP) object trap event detector provides an extension to replace the value when an SNMP trap with the specified SNMP object ID (OID) is encountered on a specific interface or address.

Syslog Event Detector

The syslog event detector allows for screening syslog messages for a regular expression pattern match. The selected messages can be further qualified, requiring that a specific number of occurrences be logged within a specified time. A match on a specified event criteria triggers a configured policy action.

System Manager Event Detector

The system manager event detector generates events for Cisco IOS Software Modularity process start, normal or abnormal stop, and restart events. The events generated by the system manager allows policies to change the default behavior of the process restart.

Timer Event Detector

The timer event detector publishes events for the following four different types of timers:

- An absolute-time-of-day timer publishes an event when a specified absolute date and time occurs.
- A countdown timer publishes an event when a timer counts down to zero.
- A watchdog timer publishes an event when a timer counts down to zero and then the timer automatically resets itself to its initial value and starts to count down again.
- A CRON timer publishes an event using a UNIX standard CRON specification to indicate when the event is to be published. A CRON timer never publishes events more than once per minute.

Watchdog System Monitor (IOSWDSysMon) Event Detector for Cisco IOS

The Cisco IOS watchdog system monitor event detector publishes an event when one of the following occurs:

- CPU utilization for a Cisco IOS task crosses a threshold.
- Memory utilization for a Cisco IOS task crosses a threshold.

Note Cisco IOS processes are now referred to as tasks to distinguish them from Cisco IOS Software Modularity processes.

Two events may be monitored at the same time, and the event publishing criteria can be specified to require one event or both events to cross their specified thresholds.

Watchdog System Monitor (WDSysMon) Event Detector for Cisco IOS Software Modularity

The Cisco IOS Software Modularity watchdog system monitor event detector detects infinite loops, deadlocks, and memory leaks in Cisco IOS Software Modularity processes.

EEM Actions Available by Cisco IOS Release

The CLI-based corrective actions that are taken when event detectors report events enable a powerful on-device event management mechanism. Some actions are available in every Cisco IOS release, but most actions have been introduced in a specific release. Use the table below to determine which actions are available in your specific Cisco IOS release. A blank entry (--) indicates that the action is not available; the text "Yes" indicates that the action is available. The actions shown in the table are supported in later releases of the same Cisco IOS release train. For more details on each action, see the Embedded Event Manager Actions concept in the "Embedded Event Manager Overview" module.

Table 49: Availability of Actions by Cisco IOS Release

Action	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS Software Modularity	12.2(33)SXH	12.4(20)T	12.4(22)T	15.0(1)M	15E XE 3E
Execute a CLI command	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Generate a CNS event	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Generate a prioritized syslog message	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Generate an SNMP trap	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Manually run an EEM policy	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Publish an application-specific event	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Read the state of a tracked object	--	--	Yes	--		Yes	Yes	Yes	Yes
Reload the Cisco software	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Request system information	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Send a short e-mail	--	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Set or modify a named counter	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Set the state of a tracked object	--	--	Yes	--		Yes	Yes	Yes	Yes
Switch to a secondary RP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Embedded Event Manager Actions

The CLI-based corrective actions that are taken when event detectors report events enable a powerful on-device event management mechanism. Some EEM actions are available on every Cisco IOS release, but most EEM actions have been introduced in a specific release. For details of which EEM action is supported in each Cisco IOS release, see the EEM Actions Available by Cisco IOS Release concept in the "Writing Embedded Event Manager Policies Using the Cisco IOS CLI" or the "Writing Embedded Event Manager Policies Using Tcl" modules. EEM supports the following actions:

- Executing a Cisco IOS command-line interface (CLI) command.
- Generating a CNS event for upstream processing by Cisco CNS devices.
- Setting or modifying a named counter.
- Switching to a secondary processor in a fully redundant hardware configuration.
- Requesting system information when an event occurs.
- Sending a short e-mail.
- Manually running an EEM policy.
- Publishing an application-specific event.
- Reloading the Cisco software.
- Generating an SNMP trap.
- Generating prioritized syslog messages.
- Reading the state of a tracked object.
- Setting the state of a tracked object.

EEM action CLI commands contain an EEM action label that is a unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric (lexicographical) key sequence using the label as the sort key. If you are using numbers as labels be aware that alphanumerical sorting will sort 10.0 after 1.0, but before 2.0, and in this situation we recommend that you use numbers such as 01.0, 02.0, and so on, or use an initial letter followed by numbers.

Embedded Event Manager Environment Variables

EEM allows environment variables to be used in EEM policies. Tool Command Language (Tcl) allows global variables to be defined that are known to all procedures within a Tcl script. EEM allows environment variables to be defined using a CLI command, the **event manager environment** command, for use within an EEM policy. All EEM environment variables are automatically assigned to Tcl global variables before a Tcl script is run. There are three different types of environment variables associated with Embedded Event Manager:

- User-defined--Defined by you if you create an environment variable in a policy that you have written.
- Cisco-defined--Defined by Cisco for a specific sample policy.
- Cisco built-in (available in EEM applets)--Defined by Cisco and can be read only or read/write. The read only variables are set by the system before an applet starts to execute. The single read/write variable, _exit_status, allows you to set the exit status at policy exit for policies triggered from synchronous events.

Cisco-defined environment variables (see the table below) and Cisco system-defined environment variables may apply to one specific event detector or to all event detectors. Environment variables that are user-defined or defined by Cisco in a sample policy are set using the **event manager environment** command. Variables that are used in the EEM policy must be defined before you register the policy. A Tcl policy contains a section called "Environment Must Define" that can be defined to check that any required environment variables are defined before the policy runs.

Cisco built-in environment variables are a subset of the Cisco-defined environment variables and the built-in variables are available to EEM applets only. The built-in variables can be read-only or can be read and write, and these variables may apply to one specific event detector or to all event detectors. For more details and a table listing the Cisco system-defined variables, see the ""Writing Embedded Event Manager Policies Using the Cisco IOS CLI" module.

Note Cisco-defined environment variables begin with an underscore character (_). We strongly recommend that customers avoid the same naming convention to prevent naming conflicts.

The table below describes the Cisco-defined variables used in the sample EEM policies. Some of the environment variables do not have to be specified for the corresponding sample policy to run and these are marked as optional.

Table 50: Cisco-Defined Environmental Variables and Examples

Environment Variable	Description	Example
_config_cmd1	The first configuration command that is executed.	**interface Ethernet1/0**
_config_cmd2	(Optional) The second configuration command that is executed.	**no shutdown**
_crash_reporter_debug	(Optional) A value that identifies whether debug information for tm_crash_reporter.tcl will be enabled.	1
_crash_reporter_url	The URL location to which the crash report is sent.	http://www.yourdomain.com/fm/interface_tm.cgi
_cron_entry	A CRON specification that determines when the policy will run. See the "Writing Embedded Event Manager Policies Using Tcl" module for more information about how to specify a cron entry.	0-59/1 0-23/1 * * 0-7
_email_server	A Simple Mail Transfer Protocol (SMTP) mail server used to send e-mail.	mailserver.yourdomain.com
_email_to	The address to which e-mail is sent.	engineer@yourdomain.com
_email_from	The address from which e-mail is sent.	devtest@yourdomain.com
_email_cc	The address to which the e-mail is be copied.	manager@yourdomain.com

Environment Variable	Description	Example
_email_ipaddr	The source IP address of the recipient.	209.165.201.1 or (IPv6 address) 2001:0DB8::1
_info_snmp_oid	The SNMP object ID.	1.3.6.1.2.1.2 or iso.internet.mgmt.mib-2.interfaces
_info_snmp_value	The value string of the associated SNMP data element.	
_show_cmd	The CLI **show** command to be executed when the policy is run.	**show version**
_syslog_pattern	A regular expression pattern match string that is used to compare syslog messages to determine when the policy runs.	.*UPDOWN.*FastEthernet 0/0.*
_tm_fsys_usage_cron	(Optional) A CRON specification that is used in the **event_register** keyword extension. If unspecified, the _tm_fsys_usage.tcl policy is triggered once per minute.	0-59/1 0-23/1 * * 0-7
_tm_fsys_usage_debug	(Optional) When this variable is set to a value of 1, disk usage information is displayed for all entries in the system.	1
_tm_fsys_usage_ freebytes	(Optional) Free byte threshold for systems or specific prefixes. If free space falls below a given value, a warning is displayed.	disk2:98000000
_tm_fsys_usage_percent	(Optional) Disk usage percentage thresholds for systems or specific prefixes. If disk usage percentage exceeds a given percentage, a warning is displayed. If unspecified, the default disk usage percentage is 80 percent for all systems.	nvram:25 disk2:5

Embedded Event Manager Policy Creation

EEM is a policy driven process in which the EEM policy engine receives notifications when faults and other events occur in the Cisco software system. Embedded Event Manager policies implement recovery based on the current state of the system and the actions specified in the policy for a given event. Recovery actions are triggered when the policy is run.

Although there are some EEM CLI configuration and **show** commands, EEM is implemented through the creation of policies. An EEM policy is an entity that defines an event and the actions to be taken when that event occurs. There are two types of EEM policies: an applet or a script. An applet is a simple form of policy that is defined within the CLI configuration. A script is a form of policy that is written in Tcl.

The creation of an EEM policy involves:

- Selecting the event for which the policy is run.

- Defining the event detector options associated with logging and responding to the event.
- Defining the environment variables, if required.
- Choosing the actions to be performed when the event occurs.

There are two ways to create an EEM policy. The first method is to write applets using CLI commands, and the second method is to write Tcl scripts. Cisco provides enhancements to Tcl in the form of Tcl command extensions that facilitate the development of EEM policies. Scripts are defined off the networking device using an ASCII editor. The script is then copied to the networking device and registered with EEM. When a policy is registered with the Embedded Event Manager, the software examines the policy and registers it to be run when the specified event occurs. Policies can be unregistered or suspended. Both types of policies can be used to implement EEM in your network.

For details on writing EEM policies using the Cisco IOS CLI, go to "Writing Embedded Event Manager Policies Using the Cisco IOS CLI" module.

For details on writing EEM policies using Tcl, go to "Writing Embedded Event Manager Policies Using Tcl" module.

Where to Go Next

- If you want to write EEM policies using the Cisco IOS CLI, see the "Writing Embedded Event Manager Policies Using the Cisco IOS CLI" module.
- If you want to write EEM policies using Tcl, see the "Writing Embedded Event Manager Policies Using Tcl" module.

Feature Information for Embedded Event Manager 4.0 Overview

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 51: Feature Information for Embedded Event Manager 4.0 Overview

Feature Name	Releases	Feature Information
Embedded Event Manager 1.0		EEM 1.0 introduced Embedded Event Manager applet creation with the SNMP and syslog event detectors. EEM 1.0 also introduced the following actions: generating prioritized syslog messages, generating a CNS event for upstream processing by Cisco CNS devices, reloading the Cisco IOS software, and switching to a secondary processor in a fully redundant hardware configuration. The following commands were introduced by this feature: **action cns-event**, **action force-switchover**, **action reload**, **action syslog**, **debug event manager**, **event manager applet**, **event snmp**, **event syslog**, **show event manager policy registered**.
Embedded Event Manager 2.0		EEM 2.0 introduced the application-specific event detector, the counter event detector, the interface counter event detector, the timer event detector, and the watchdog event detector. New actions included modifying a named counter, publishing an application-specific event, and generating an SNMP trap. The ability to define environment variables and to run EEM policies written using Tcl was introduced, and two sample policies were included with the software. The following commands were introduced by this feature: **action counter**, **action publish-event**, **action snmp-trap**, **event application**, **event counter**, **event interface**, **event ioswdsysmon**, **event manager environment**, **event manager history size**, **event manager policy**, **event manager scheduler suspend**, **event timer**, **show event manager environment**, **show event manager history events**, **show event manager history traps**, **show event manager policy available**, **show event manager policy pending**.
Embedded Event Manager 2.1		EEM 2.1 introduced some new event detectors and actions with new functionality to allow EEM policies to be run manually and the ability to run multiple concurrent policies. Support for Simple Network Management Protocol (SNMP) event detector rate-based events was provided as was the ability to create policies using Tool Command Language (Tcl). The following commands were introduced or modified by this feature: **action cli**, **action counter**, **action info**, **action mail**, **action policy**, **debug event manager**, **event cli**, **event manager directory user**, **event manager policy**, **event manager run**, **event manager scheduler script**, **event manager session cli username**, **event none**, **event oir**, **event snmp**, **event syslog**, **set**(EEM), **show event manager directory user**, **show event manager policy registered**, **show event manager session cli username**.

Feature Name	Releases	Feature Information
Embedded Event Manager 2.1 (Software Modularity)	12.2(18)SXF4 Cisco IOS Software Modularity images	EEM 2.1 for Software Modularity images introduced the GOLD, system manager, and WDSysMon (Cisco IOS Software Modularity watchdog) event detectors, and the ability to display Cisco IOS Software Modularity processes and process metrics. The following commands were introduced by this feature: **event gold**, **event process**, **show event manager metric process**. **Note** EEM 2.1 for Software Modularity images also supports the resource and RF event detectors introduced in EEM 2.2, but it does not support the enhanced object tracking event detector or the actions to read and set tracked objects.
Embedded Event Manager 2.2	12.2(31)SB3 12.2(33)SRB	EEM 2.2 introduced the enhanced object tracking, resource, and RF event detectors. The actions of reading and setting the state of a tracked object were also introduced. The following commands were introduced or modified by this feature: **action track read**, **action track set**, **default-state**, **event resource**, **event rf**, **event track**, **show track**, **track stub-object**.
SNMP event detector delta environment variable		A new SNMP event detector environment variable, _snmp_oid_delta_val, was introduced. This is a minor enhancement. Minor enhancements are not typically listed in Feature Navigator.
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB	EEM 2.3 introduced some new features relative to the Generic Online Diagnostics (GOLD) Event Detector on the Cisco Catalyst 6500 Series switches. The **event gold** command was enhanced in addition to the Tcl keywords–**action-notify**, **testing-type**, **test-name**, **test-id**, **consecutive-failure**, **platform-action**, and **maxrun**--for improved reaction to GOLD test failures and conditions Read-only variables were added under the **GOLD Event Detector** category to provide access to platform-wide and test-specific GOLD event detector information for a detected event.
Embedded Event Manager 2.4	12.2(33)SXI 12.2(33)SRE	EEM 2.4 introduced several new features. The following commands were introduced by this feature: **attribute (EEM)** , **correlate**, **event manager detector rpc**, **event manager directory user repository**, **event manager update user policy**, **event manager scheduler clear**, **event manager update user policy**, **event owner**, **event rpc**, **event snmp-notification**, **show event manager detector**, **show event manager version**, **trigger (EEM)**.

Feature Name	Releases	Feature Information
Embedded Event Manger 3.0	12.2(33)SRE	EEM 3.0 introduced several new features. The following commands were introduced or modified by this feature: **action add** , **action append**, **action break**, **action comment**, **action context retrieve**, **action context save**, **action continue**, **action decrement**, **action divide**, **action else**, **action elseif**, **action end**, **action exit**, **action foreach**, **action gets**, **action if**, **action if goto**, **action increment**, **action info type interface-names**, **action info type snmp getid**, **action info type snmp inform**, **action info type snmp oid**, **action info type snmp trap**, **action info type snmp var**, **action multiply**, **action puts**, **action regexp**, **action set (EEM)**, **action string compare**, **action string equal**, **action string first**, **action string index**, **action string last**, **action string length**, **action string match**, **action string range**, **action string replace**, **action string tolower**, **action string toupper**, **action string trim**, **action string trimleft**, **action string trimright**, **action subtract**, **action while**, **event cli**, **event ipsla**, **event manager detector routing**, **event manager scheduler**, **event manager scheduler clear**, **event manager scheduler hold**, **event manager scheduler modify**, **event manager scheduler release**, **event nf**, **event routing**, **show event manager policy active**, **show event manager policy pending**, and **show event manager scheduler**.
Embedded Event Manager 3.1		EEM 3.1 introduced several new features. The following commands were introduced or modified by this feature: **action syslog**, **description (EEM)**, **event manager applet**, **event manager policy**, **event snmp-notification**, **event snmp-object**, **show event manager policy registered**, and **show event manager policy available**.
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG	EEM 3.2 introduced several new features. The following commands were introduced or modified: **debug event manager**, **event identity**, **event mat**, **event neighbor-discovery**, **show event manager detector**.
Embedded Event Manager 4.0		EEM 4.0 introduced several new features. The following commands were introduced or modified: **action file**, **action mail**, **action syslog**, **clear event manager detector counters**, **clear event manager server counters**, **event cli**, **event manager policy**, **event manager scheduler**, **event syslog**, **show event manager detector**, **show event manager policy registered**, **show event manager statistics**.

Additional References

The following sections provide references related to EEM.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
EEM commands: complete command syntax, defaults, command mode, command history, usage guidelines, and examples	Cisco IOS Embedded Event Manager Command Reference
Embedded Event Manager policy writing using the CLI	Writing Embedded Event Manager Policies Using the Cisco IOS CLI module
Embedded Event Manager policy writing using Tcl	Writing Embedded Event Manager Policies Using Tcl module
Embedded Resource Manager	Embedded Resource Manager module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
CISCO-EMBEDDED-EVENT-MGR-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

CHAPTER 36

Writing Embedded Event Manager Policies Using the Cisco IOS CLI

This module describes how to write Embedded Event Manager (EEM) policies using Cisco IOS command-line interface (CLI) applets to handle Cisco software faults and events. EEM is a distributed and customized approach to event detection and recovery offered directly in a Cisco IOS device. EEM offers the ability to monitor events and take informational, corrective, or any desired action when the monitored events occur or when a threshold is reached. The EEM policy engine receives notifications when faults and other events occur. EEM policies implement recovery on the basis of the current state of the system and the actions specified in the policy for a given event. Recovery actions are triggered when the policy is run.

Prerequisites for Writing EEM Policies Using the Cisco IOS CLI

- Before writing EEM policies, you should be familiar with the concepts explained in the “Embedded Event Manager Overview” module.
- If the **action cns-event** command is used, access to a Cisco Networking Services (CNS) Event gateway must be configured.
- If the **action force-switchover** command is used, a secondary processor must be configured on the device.
- If the **action snmp-trap** command is used, the **snmp-server enable traps event-manager** command must be enabled to permit SNMP traps to be sent from the Cisco IOS device to the SNMP server. Other relevant **snmp-server** commands must also be configured; for details see the **action snmp-trap** command page.

Information About Writing EEM Policies Using the Cisco IOS CLI

Embedded Event Manager Policies

EEM offers the ability to monitor events and take informational or corrective action when the monitored events occur or a threshold is reached. An EEM policy is an entity that defines an event and the actions to be taken when that event occurs. There are two types of EEM policies: an applet or a script. An applet is a simple form of policy that is defined within the CLI configuration. A script is a form of policy that is written in Tool Command Language (Tcl).

EEM Applet

An EEM applet is a concise method for defining event screening criteria and the actions to be taken when that event occurs. In applet configuration mode, three types of configuration statements are supported. The **event** commands are used to specify the event criteria to trigger the applet to run, the **action** commands are used to specify an action to perform when the EEM applet is triggered, and the **set** command is used to set the value of an EEM applet variable. Currently only the _exit_status variable is supported for the **set** command.

Only one **event** configuration command is allowed within an applet configuration. When applet configuration mode is exited and no **event** command is present, a warning is displayed stating that no event is associated with this applet. If no event is specified, this applet is not considered registered. When no action is associated with this applet, events are still triggered but no actions are performed. Multiple **action** configuration commands are allowed within an applet configuration. Use the **show event manager policy registered** command to display a list of registered applets.

Before modifying an EEM applet, be aware that the existing applet is not replaced until you exit applet configuration mode. While you are in applet configuration mode modifying the applet, the existing applet may be executing. It is safe to modify the applet without unregistering it. When you exit applet configuration mode, the old applet is unregistered and the new version is registered.

The action configuration commands are uniquely identified using the *label* argument, which can be any string value. Actions are sorted in ascending alphanumeric key sequence using the *label* argument as the sort key, and they are run using this sequence.

The Embedded Event Manager schedules and runs policies on the basis of an event specification that is contained within the policy itself. When applet configuration mode is exited, EEM examines the **event** and **action** commands that are entered and registers the applet to be run when a specified event occurs.

EEM Script

Scripts are defined off the networking device using an ASCII editor. The script is then copied to the networking device and registered with EEM. Tcl scripts are supported by EEM.

EEM allows you to write and implement your own policies using Tcl. Writing an EEM policy involves:

- Selecting the event for which the policy is run.
- Defining the event detector options associated with logging and responding to the event.
- Choosing the actions to be followed when the event occurs.

Cisco provides enhancements to Tcl in the form of keyword extensions that facilitate the development of EEM policies. The main categories of keywords identify the detected event, the subsequent action, utility information, counter values, and system information. For more details about writing EEM policies using Tcl, see the "Writing Embedded Event Manager Policies Using Tcl" module.

Embedded Event Manager Built-In Environment Variables Used in EEM Applets

EEM built-in environment variables are a subset of the Cisco-defined environment variables and the built-in variables are available to EEM applets only. The built-in variables can be read-only or can be read and write and these variables may apply to one specific event detector or to all event detectors. The table below lists the Cisco built-in environment variables that are read-only alphabetically by event detector and subevent.

Table 52: EEM Built-In Environment Variables (Read Only)

Environment Variable	Description
All Events	
_event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
_event_type	Type of event.
_event_type_string	An ASCII string identifier of the event type that triggered the event.
_event_pub_sec _event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.
_event_severity	The severity of the event.
Application-Specific Event Detector	
_application_component_id	The event application component identifier.
_application_data1	The value of an environment variable, character text, or a combination of the two to be passed to an application-specific event when the event is published.
_application_data2	The value of an environment variable, character text, or a combination of the two to be passed to an application-specific event when the event is published.
_application_data3	The value of an environment variable, character text, or a combination of the two to be passed to an application-specific event when the event is published.
_application_data4	The value of an environment variable, character text, or a combination of the two to be passed to an application-specific event when the event is published.
_application_sub_system	The event application subsystem number.

Environment Variable	Description
_application_type	The type of application.
CLI Event Detector	
_cli_msg	The fully expanded message that triggered the CLI event.
_cli_msg_count	The number of times that a message match occurred before the event was published.
Counter Event Detector	
_counter_name	The name of the counter.
_counter_value	The value of the counter.
Enhanced Object Tracking Event Detector	
_track_number	The number of the tracked object.
_track_state	The state of the tracked object; down or up.
GOLD Event Detector	
_action_notify	The action notify information in a GOLD event flag; either false or true.
_event_severity	The event severity which can be one of the following; normal, minor, or major.
_gold_bl	The boot diagnostic level, which can be one of the following values: • 0: complete diagnostic • 1: minimal diagnostic • 2: bypass diagnostic
_gold_card	The card on which a GOLD failure event was detected.
_gold_cf *testnum*	Consecutive failure, where *testnum* is the test number. For example, **_gold_cf3** is the EEM built-in environment variable for consecutive failure of test 3.
_gold_ci	Card index.
_gold_cn	Card name.
_gold_ec *testnum*	Test error code, where *testnum* is the test number. For example, **_gold_ec3** is the EEM built-in environment variable for the error code of test 3.

Environment Variable	Description
_gold_lf *testnum*	Last fail time, where *testnum* is the test number. For example, **_gold_lf3** is the EEM built-in variable for the last fail time of test 3. The time-stamp format is *mmm dd yyyy hh*:*mm*:*ss*. For example, Mar 11 2005 08:47:00.
_gold_new_failure	The new test failure information in a GOLD event flag; either true or false.
_gold_overall_result	The overall diagnostic result, which can be one of the following values: • 0: OK • 3: minor error • 4: major error • 14: unknown result
_gold_pc	Port counts.
_gold_rc *testnum*	Test total run count, where *testnum* is the test number. For example, **_gold_rc3** is the EEM built-in variable for the total run count of test 3.
_gold_sn	Card serial number.
_gold_sub_card	The subcard on which a GOLD failure event was detected.
_gold_ta *testnum*	Test attribute, where *testnum* is the test number. For example, **_gold_ta3** is the EEM built-in variable for the test attribute of test 3.
_gold_tc	Test counts.
_gold_tf *testnum*	Total failure count, where *testnum* is the test number. For example, **_gold_tf3** is the EEM built-in variable for the total failure count of test 3.
_gold_tn *testnum*	Test name, where *testnum* is the test number. For example, **_gold_tn3** is the EEM built-in variable for the name of test 3.

Environment Variable	Description
_gold_tr *testnum*	Test result, where *testnum* is the test number. For example, **_gold_tr6** is the EEM built-in variable for test 6, where test 6 is not a per-port test and not a per-device test. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
_gold_tr *testnum* **d** *devnum*	Per-device test result, where *testnum* is the test number and *devnum* is the device number. For example, **_gold_tr3d20** is the EEM built-in variable for the test result for test 3, device 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
_gold_tr *testnum* **p** *portnum*	Per-port test result, where *testnum* is the test number and *portnum* is the port number. For example, **_gold_tr5p20** is the EEM built-in variable for the test result for test 5, port 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
_gold_tt	The testing type, which can be one of the following: • 1: a boot diagnostic • 2: an on-demand diagnostic • 3: a schedule diagnostic • 4: a monitoring diagnostic
Interface Counter Event Detector	
_interface_is_increment	A value to indicate whether the current interface counter value is an absolute value (0) or an increment value (1).
_interface_name	The name of the interface to be monitored.
_interface_parameter	The name of the interface counter to be monitored.

Environment Variable	Description
_interface_value	A value with which the current interface counter value is compared.
None Event Detector	
_event_id	A value of 1 indicates an insertion event; a value of 2 indicates a removal event.
_none_argc **_none_arg1** **_none_arg2** **_none_arg3** **_none_arg4** **_none_arg5** **_none_arg6** **_none_arg7** **_none_arg8** **_none_arg9** **_none_arg10** **_none_arg11** **_none_arg12** **_none_arg13** **_none_arg14** **_none_arg15**	The parameters that are passed from the XML SOAP command to the script.
OIR Event Detector	
_oir_event	A value of 1 indicates an insertion event; a value of 2 indicates a removal event.
_oir_slot	The slot number for the OIR event.
Resource Event Detector	
_resource_configured_threshold	The configured ERM threshold.
_resource_current_value	The current value reported by ERM.
_resource_dampen_time	The ERM dampen time, in nanoseconds.
_resource_direction	The ERM event direction. The event direction can be one of the following: up, down, or no change.

Environment Variable	Description
_resource_level	The ERM event level. The four event levels are normal, minor, major, and critical.
_resource_notify_data_flag	The ERM notify data flag.
_resource_owner_id	The ERM resource owner ID.
_resource_policy_id	The ERM policy ID.
_resource_policy_violation_flag	The ERM policy violation flag; either false or true.
_resource_time_sent	The ERM event time, in nanoseconds.
_resource_user_id	The ERM resource user ID.
RF Event Detector	
_rf_event	A value of 0 indicates that this is not an RF event; a value of 1 indicates an RF event.
RPC Event Detector	
_rpc_event	A value of 0 indicates that there is no error; a value of 1 to 83 indicates error.
_rpc_argc **_rpc_arg0** **_rpc_arg1** **_rpc_arg2** **_rpc_arg3** **_rpc_arg4** **_rpc_arg5** **_rpc_arg6** **_rpc_arg7** **_rpc_arg8** **_rpc_arg9** **_rpc_arg10** **_rpc_arg11** **_rpc_arg12** **_rpc_arg13** **_rpc_arg14**	The parameters that are passed from the XML SOAP command to the applet.
SNMP Event Detector	

Environment Variable	Description
_snmp_exit_event	A value of 0 indicates that this is not an exit event; a value of 1 indicates an exit event.
_snmp_oid	The SNMP object ID that caused the event to be published.
_snmp_oid_delta_val	The actual incremental difference between the value of the current SNMP object ID and the value when the event was last triggered.
_snmp_oid_val	The SNMP object ID value when the event was published.
SNMP Notification Event Detector	
_snmp_notif_oid	A user specified object ID.
_snmp_notif_oid_val	A user specified object ID value.
_snmp_notif_src_ip_addr	The source IP address of the SNMP Protocol Data Unit (PDU).
_snmp_notif_dest_ip_addr	The destination IP address of the SNMP PDU.
_x_x_x_x_x_x_x(varbinds)	The SNMP PDU varbind information.
_snmp_notif_trunc_vb_buf	Indicates whether the varbind information has been truncated due to the lack of space in the buffer.
Syslog Event Detector	
_syslog_msg	The syslog message that caused the event to be published.
System Manager (Process) Event Detector	
_process_dump_count	The number of times that a Posix process was dumped.
_process_exit_status	The status of the Posix process at exit.
_process_fail_count	The number of times that a Posix process failed.
_process_instance	The instance number of the Posix process.
_process_last_respawn	The Posix process that was last respawned.
_process_node_name	The node name of the Posix process.
_process_path	The path of the Posix process.
_process_process_name	The name of the Posix process.
_process_respawn_count	The number of times that a Posix process was respawned.
Timer Event Detector	

Environment Variable	Description
_timer_remain	The time available before the timer expires. **Note** This environment variable is not available for the CRON timer.
_timer_time	The time at which the last event was triggered.
_timer_type	The type of timer.
Watchdog System Monitor (IOSWDSysMon) Event Detector	
_ioswd_node	The slot number for the Route Processor (RP) reporting node.
_ioswd_num_subs	The number of subevents present.
All Watchdog System Monitor (IOSWDSysMon) Subevents	
_ioswd_sub1_present **_ioswd_sub2_present**	A value to indicate whether subevent 1 or subevent 2 is present. A value of 1 means that the subevent is present; a value of 0 means that the subevent is not present.
_ioswd_sub1_type _ioswd_sub2_type	The event type, either cpu_proc or mem_proc.
Watchdog System Monitor (IOSWDSysMon) cpu_proc Subevents	
_ioswd_sub1_path _ioswd_sub2_path	A process name of subevents.
_ioswd_sub1_period _ioswd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.
_ioswd_sub1_pid _ioswd_sub2_pid	The process identifier of subevents.
_ioswd_sub1_taskname **_ioswd_sub2_taskname**	The task name of subevents.
_ioswd_sub1_value _ioswd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (IOSWDSysMon) mem_proc Subevents	
_ioswd_sub1_diff _ioswd_sub2_diff	A percentage value of the difference that triggered the event. **Note** This variable is set only when the **_ioswd_sub1_is_percent** or **_ioswd_sub2_is_percent** variable contains a value of 1.
_ioswd_sub1_is_percent **_ioswd_sub2_is_percent**	A number that identifies whether the value is a percentage. A value of 0 means that the value is not a percentage; a value of 1 means that the value is a percentage.

Environment Variable	Description
_ioswd_sub1_path _ioswd_sub2_path	The process name of subevents.
_ioswd_sub1_pid _ioswd_sub2_pid	The process identifier of subevents.
_ioswd_sub1_taskname **_ioswd_sub2_taskname**	The task name of subevents.
_ioswd_sub1_value _ioswd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (WDSysMon) Event Detector	
_wd_sub1_present _wd_sub2_present	A value to indicate whether subevent 1 or subevent 2 is present. A value of 1 means that the subevent is present; a value of 0 means that the subevent is not present.
_wd_num_subs	The number of subevents present.
_wd_sub1_type _wd_sub2_type	The event type: cpu_proc, cpu_tot, deadlock, dispatch_mgr, mem_proc, mem_tot_avail, or mem_tot_used.
Watchdog System Monitor (WDSysMon) cpu_proc Subevents	
_wd_sub1_node _wd_sub2_node	The slot number for the subevent RP reporting node.
_wd_sub1_period _wd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.
_wd_sub1_procname _wd_sub2_procname	The process name of subevents.
_wd_sub1_value _wd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (WDSysMon) cpu_tot Subevents	
_wd_sub1_node _wd_sub2_node	The slot number for the subevent RP reporting node.
_wd_sub1_period _wd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.
_wd_sub1_value _wd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (WDSysMon) deadlock Subevents	
_wd_sub1_entry_ [1-N]**_b_node** **_wd_sub2_entry_**[1-N]**_b_node**	The slot number for the subevent RP reporting node.
_wd_sub1_entry_ [1-N]**_b_pid** **_wd_sub2_entry_**[1-N]**_b_pid**	The process identifier of subevents.

Environment Variable	Description
_wd_sub1_entry_ [1-N]**_b_procname** **_wd_sub2_entry_**[1-N]**_b_procname**	The process name of subevents.
_wd_sub1_entry_ [1-N]**_b_tid** **_wd_sub2_entry_**[1-N]**_b_tid**	The time identifier of subevents.
_wd_sub1_entry_ [1-N]**_node** **_wd_sub2_entry_**[1-N]**_node**	The slot number for the subevent RP reporting node.
_wd_sub1_entry_ [1-N]**_pid** **_wd_sub2_entry_**[1-N]**_pid**	The process identifier of subevents.
_wd_sub1_entry_ [1-N]**_procname** **_wd_sub2_entry_**[1-N]**_procname**	The process name of subevents.
_wd_sub1_entry_ [1-N]**_state** **_wd_sub2_entry_**[1-N]**_state**	The time identifier of subevents.
_wd_sub1_entry_ [1-N]**_tid** **_wd_sub2_entry_**[1-N]**_tid**	The time identifier of subevents.
_wd_sub1_num_entries **_wd_sub2_num_entries**	The number of subevents.
Watchdog System Monitor (WDSysMon) dispatch manager Subevents	
_wd_sub1_node _wd_sub2_node	The slot number for the subevent RP reporting node.
_wd_sub1_period _wd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.
_wd_sub1_procname _wd_sub2_procname	The process name of subevents.
_wd_sub1_value _wd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (WDSysMon) mem_proc Subevents	
_wd_sub1_diff _wd_sub2_diff	A percentage value of the difference that triggered the event. **Note** This variable is set only when the **_wd_sub1_is_percent** or **_wd_sub2_is_percent** variable contains a value of 1.
_wd_sub1_is_percent **_wd_sub2_is_percent**	A number that identifies whether the value is a percentage. A value of 0 means that the value is not a percentage; a value of 1 means that the value is a percentage.
_wd_sub1_node _wd_sub2_node	The slot number for the subevent RP reporting node.
_wd_sub1_period _wd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.

Environment Variable	Description
_wd_sub1_pid _wd_sub2_pid	The process identifier of subevents.
_wd_sub1_procname _wd_sub2_procname	The process name of subevents.
_wd_sub1_value _wd_sub2_value	The CPU utilization of subevents measured as a percentage.
Watchdog System Monitor (WDSysMon) mem_tot_avail and mem_tot_used Subevents	
_wd_sub1_avail _wd_sub2_avail	The memory available for subevents.
_wd_sub1_diff _wd_sub2_diff	A percentage value of the difference that triggered the event. **Note** This variable is set only when the **_wd_sub1_is_percent** or **_wd_sub2_is_percent** variable contains a value of 1.
_wd_sub1_is_percent **_wd_sub2_is_percent**	A number that identifies whether the value is a percentage. A value of 0 means that the value is not a percentage; a value of 1 means that the value is a percentage.
_wd_sub1_node _wd_sub2_node	The slot number for the subevent RP reporting node.
_wd_sub1_period _wd_sub2_period	The time period, in seconds and optional milliseconds, used for measurement in subevents.
_wd_sub1_value _wd_sub2_value	The CPU utilization of subevents measured as a percentage.
_wd_sub1_used _wd_sub2_used	The memory used by subevents.

How to Write EEM Policies Using the Cisco IOS CLI

Registering and Defining an Embedded Event Manager Applet

Perform this task to register an applet with Embedded Event Manager and to define the EEM applet using the Cisco IOS CLI **event** and **action** commands. Only one **event** command is allowed in an EEM applet. Multiple **action** commands are permitted. If no **event** and no **action** commands are specified, the applet is removed when you exit configuration mode.

The SNMP event detector and the syslog **action** commands used in this task are just representing any event detector and **action** commands. For examples using other event detectors and **action** commands, see the Embedded Event Manager Applet Configuration Examples, on page 574.

EEM Environment Variables

EEM environment variables for EEM policies are defined using the EEM **event manager environment** configuration command. By convention, all Cisco EEM environment variables begin with " _ ". In order to avoid future conflict, customers are urged not to define new variables that start with " _ ".

You can display the EEM environment variables set on your system by using the **show event manager environment** privileged EXEC command.

For example, you can create EEM policies that can send e-mails when an event occurs. The table below describes the e-mail-specific environment variables that can be used in EEM policies.

Table 53: EEM E-mail-Specific Environmental Variables

Environment Variable	Description
_email_server	A Simple Mail Transfer Protocol (SMTP) mail server use send e-mail.
_email_to	The address to which e-mail is sent.
_email_from	The address from which e-mail is sent.
_email_cc	The address to which the e-mail is copied.

Alphabetical Order of EEM Action Labels

An EEM action label is a unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric (lexicographical) key sequence using the label as the sort key. If you are using numbers as labels be aware that alphanumerical sorting will sort 10.0 after 1.0, but before 2.0, and in this situation we recommend that you use numbers such as 01.0, 02.0, and so on, or use an initial letter followed by numbers.

SUMMARY STEPS

1. **enable**
2. **show event manager environment** [**all**| *variable-name*]
3. **configure terminal**
4. **event manager environment** *variable-name string*
5. Repeat Step 4 for all the required environment variables.
6. **event manager applet** *applet-name*
7. Do one of the following:
   - **event snmp oid** *oid-value* **get-type** {**exact**| **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb**| **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
8. **action** *label* **cli command** *cli-string* [**pattern** *pattern-string*]
9. **action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string*
10. **action** *label* **mail server** *server-address* **to** *to-address* **from** *from-address* [**cc** *cc-address*] **subject** *subject* **body** *body-text*
11. Add more action commands as required.
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager environment** [**all**| *variable-name*] **Example:** `Device# show event manager environment all`	(Optional) Displays the name and value of EEM environment variables. • The optional **all** keyword displays all the EEM environment variables. • The optional *variable-name*argument displays information about the specified environment variable.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**event manager environment** *variable-name string* **Example:** `Device(config)# event manager environment _email_to engineering@example.com`	Configures the value of the specified EEM environment variable. • In this example, the environment variable that holds the e-mail address to which e-mail is sent is set to engineering@example.com.
Step 5	Repeat Step 4 for all the required environment variables.	Repeat Step 4 to configure all the environment variables required by the policy to be registered in Step 6.
Step 6	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet memory-fail`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 7	Do one of the following: • **event snmp oid** *oid-value* **get-type** {**exact**| **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb**| **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value* **Example:** `Device(config-applet)# event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val 5120000 poll-interval 90`	Specifies the event criteria that cause the EEM applet to run. • In this example, an EEM event is triggered when free memory falls below the value of 5120000. • Exit criteria are optional, and if not specified, event monitoring is reenabled immediately.

	Command or Action	Purpose
Step 8	**action** *label* **cli command** *cli-string* [**pattern** *pattern-string*] **Example:** `Device(config-applet)# action 1.0 cli command "enable"` **Example:** `Device(config-applet)# action 2.0 cli command "clear counters Ethernet0/1" pattern "confirm"` **Example:** `Device(config-applet)# action 3.0 cli command "y"`	Specifies the action of executing a Cisco IOS CLI command when an EEM applet is triggered. The **pattern** keyword is optional and is used only when the command string solicits input. The **action cli**command ends when the solicited prompt as specified in the optional **pattern** keyword is received. You are required to specify a regular expression pattern that will match the next solicited prompt. Specification of an incorrect pattern will cause the **action cli** command to wait forever until the applet execution times out due to the maxrun timer expiration. • The action taken is to specify an EEM applet to run when the **pattern** keyword specifies the *confirm* argument for the **clear counters Ethernet0/1** command. In this case the command string solicits input, such as "confirm," which has to be completed with a "yes" or a "no" input.
Step 9	**action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string* **Example:** `Device(config-applet)# action 1.0 syslog priority critical msg "Memory exhausted; current available memory is $_snmp_oid_val bytes"` **Example:** `Device(config-applet)# action 1.0 syslog priority errors facility EEM-FAC message "TEST MSG"`	Specifies the action to be taken when an EEM applet is triggered. In this example, the action taken is to write a message to syslog. • The optional **priority** keyword specifies the priority level of the syslog messages. If selected, the *priority-level* argument must be defined. • The *msg-text* argument can be character text, an environment variable, or a combination of the two. • The **facility** keyword specifies the location of generated message • The *string* argument can be character text, an environment variable, or a combination of the two.
Step 10	**action** *label* **mail server** *server-address* **to** *to-address* **from** *from-address* [**cc** *cc-address*] **subject** *subject* **body** *body-text* **Example:** `Device(config-applet)# action 2.0 mail server 192.168.1.10 to engineering@example.com from devtest@example.com subject "Memory failure" body "Memory exhausted; current available memory is $_snmp_oid_val bytes"`	Specifies the action of sending a short e-mail when an EEM applet is triggered. • The *server-address* argument specifies the fully qualified domain name of the e-mail server to be used to forward the e-mail. • The *to-address* argument specifies the e-mail address where the e-mail is to be sent. • The *from-address* argument specifies the e-mail address from which the e-mail is sent. • The *subject* argument specifies the subject line content of the e-mail as an alphanumeric string.

	Command or Action	Purpose
		• The *body-text* argument specifies the text content of the e-mail as an alphanumeric string.
Step 11	Add more action commands as required.	--
Step 12	**end** **Example:** `Device(config-applet)# end`	Exits applet configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the **debug event manager** command in privileged EXEC mode to troubleshoot EEM command operations. Use any debugging command with caution as the volume of generated output can slow or stop the device operations. We recommend that this command be used only under the supervision of a Cisco engineer.

Registering and Defining an Embedded Event Manager Policy to Run Manually

There are two ways to manually run an EEM policy. EEM usually schedules and runs policies on the basis of an event specification that is contained within the policy itself. The **event none** command allows EEM to identify an EEM policy that can be manually triggered. To run the policy, use either the **action policy** command in applet configuration mode or the **event manager run** command in privileged EXEC mode.

Perform this task to register an EEM policy to be run manually using the **event manager run** command. For an example of how to manually run a policy using the **action policy** command, see the Embedded Event Manager Manual Policy Execution Examples, on page 579.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string*
6. **end**
7. **event manager run** *applet-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet manual-policy`	Registers the applet with the Embedded Event Manager and enters applet configuration mode.
Step 4	**event none** **Example:** `Device(config-applet)# event none`	Specifies that an EEM policy is to be registered with the EEM and can be run manually.
Step 5	**action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string* **Example:** `Device(config-applet)# action 1.0 syslog msg "Manual-policy triggered"`	Specifies the action to be taken when an EEM applet is triggered. In this example, the action to be taken is to write a message to syslog. • The optional **priority** keyword specifies the priority level of the syslog messages. If selected, the *priority-level* argument must be defined. • The *msg-text* argument can be character text, an environment variable, or a combination of the two. • The **facility** keyword specifies the location of generated message. • The *string* argument can be character text, an environment variable, or a combination of the two.
Step 6	**end** **Example:** `Device(config-applet)# end`	Exits applet configuration mode and returns to privileged EXEC mode.
Step 7	**event manager run** *applet-name* **Example:** `Device# event manager run manual-policy`	Manually runs a registered EEM policy.

Unregistering Embedded Event Manager Policies

Perform this task to remove an EEM policy from the running configuration file. Execution of the policy is canceled.

SUMMARY STEPS

1. **enable**

2. **show event manager policy registered** [**description** [*policy-name*] | **detailed** *policy-filename* [**system** | **user**] | [**event-type** *event-name*] [**system** | **user**] [**time-ordered** | **name-ordered**]]
3. **configure terminal**
4. **no event manager policy** *policy-filename*
5. **exit**
6. Repeat Step 2 to ensure that the policy has been removed.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy registered** [**description** [*policy-name*]	**detailed** *policy-filename* [**system**	**user**]	[**event-type** *event-name*] [**system**	**user**] [**time-ordered**	**name-ordered**]] **Example:** `Device# show event manager policy registered`	(Optional) Displays the EEM policies that are currently registered. • The optional **system**and **user** keywords display the registered system and user policies. • If no keywords are specified, EEM registered policies for all event types are displayed in time order.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**no event manager policy** *policy-filename* **Example:** `Device(config)# no event manager policy IPSLAping1`	Removes the EEM policy from the configuration, causing the policy to be unregistered.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	Repeat Step 2 to ensure that the policy has been removed. **Example:** `Device# show event manager policy registered`	--

Examples

In the following example, the **show event manager policy registered** privileged EXEC command is used to display the two EEM applets that are currently registered:

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  snmp                Off   Fri Aug 12 17:42:52 2005  IPSLAping1
 oid {1.3.6.1.4.1.9.9.42.1.2.9.1.6.4} get-type exact entry-op eq entry-val {1}
 exit-op eq exit-val {2} poll-interval 90.000
 action 1.0 syslog priority critical msg "Server IPecho Failed: OID=$_snmp_oid_val"
 action 1.1 snmp-trap strdata "EEM detected server reachability failure to 10.1.88.9"
 action 1.2 publish-event sub-system 88000101 type 1 arg1 "10.1.88.9" arg2 "IPSLAEcho"
arg3 "fail"
 action 1.3 counter name _IPSLA1F op inc value 1
2    applet  system  snmp                Off   Thu Sep 15 05:57:16 2005  memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg Memory exhausted; current available memory is
$_snmp_oid_val bytes
 action 2.0 force-switchover
```

In the following example, the **show event manager policy registered** privileged EXEC command is used to show that applet IPSLAping1 has been removed after entering the **no event manager policy** command:

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  snmp                Off   Thu Sep 15 05:57:16 2005  memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg Memory exhausted; current available memory is
$_snmp_oid_val bytes
 action 2.0 force-switchover
```

Suspending All Embedded Event Manager Policy Execution

Perform this task to immediately suspend the execution of all EEM policies. Suspending policies, instead of unregistering them might be necessary for reasons of temporary performance or security.

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**description** [*policy-name*] | **detailed** *policy-filename* [**system** | **user**] | [**event-type** *event-name*] [**system** | **user**] [**time-ordered** | **name-ordered**]]
3. **configure terminal**
4. **event manager scheduler suspend**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show event manager policy registered** [**description** [*policy-name*] | **detailed** *policy-filename* [**system** | **user**] | [**event-type** *event-name*] [**system** | **user**] [**time-ordered** | **name-ordered**]] **Example:** `Device# show event manager policy registered`	(Optional) Displays the EEM policies that are currently registered. • The optional **system**and **user** keywords display the registered system and user policies. • If no keywords are specified, EEM registered policies for all event types are displayed in time order.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**event manager scheduler suspend** **Example:** `Device(config)# event manager scheduler suspend`	Immediately suspends the execution of all EEM policies.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Displaying Embedded Event Manager History Data

Perform this optional task to change the size of the history tables and to display EEM history data.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager history size** {**events** | **traps**} [*size*]
4. **exit**
5. **show event manager history events** [**detailed**] [**maximum** *number*]
6. **show event manager history traps** {**server** | **policy**}

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **configure terminal**

Enters global configuration mode.

Example:

```
Device# configure terminal
```

Step 3 **event manager history size** {**events** | **traps**} [*size*]

Use this command to change the size of the EEM event history table or the size of the EEM SNMP trap history table. In the following example, the size of the EEM event history table is changed to 30 entries:

Example:

```
Device(config)# event manager history size events 30
```

Step 4 **exit**

Exits global configuration mode and returns to privileged EXEC mode.

Example:

```
Device(config)# exit
```

Step 5 **show event manager history events** [**detailed**] [**maximum** *number*]

Use this command to display detailed information about each EEM event, for example:

Example:

```
Device# show event manager history events
No.  Time of Event             Event Type          Name
1    Fri Aug13  21:42:57 2004  snmp                applet: SAAping1
2    Fri Aug13  22:20:29 2004  snmp                applet: SAAping1
3    Wed Aug18  21:54:48 2004  snmp                applet: SAAping1
4    Wed Aug18  22:06:38 2004  snmp                applet: SAAping1
5    Wed Aug18  22:30:58 2004  snmp                applet: SAAping1
6    Wed Aug18  22:34:58 2004  snmp                applet: SAAping1
7    Wed Aug18  22:51:18 2004  snmp                applet: SAAping1
8    Wed Aug18  22:51:18 2004  application         applet: CustApp1
```

Step 6 **show event manager history traps** {**server** | **policy**}

Use this command to display the EEM SNMP traps that have been sent either from the EEM server or from an EEM policy. In the following example, the EEM SNMP traps that were triggered from within an EEM policy are displayed.

Example:

```
Device# show event manager history traps policy
No.  Time                       Trap Type           Name
1    Wed Aug18  22:30:58 2004  policy              EEM Policy Director
2    Wed Aug18  22:34:58 2004  policy              EEM Policy Director
3    Wed Aug18  22:51:18 2004  policy              EEM Policy Director
```

Displaying Embedded Event Manager Registered Policies

Perform this optional task to display registered EEM policies.

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**]

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 show event manager policy registered [event-type event-name] [time-ordered| name-ordered]

Use this command with the **time-ordered** keyword to display information about currently registered policies sorted by time, for example:

Example:

```
Device# show event manager policy registered time-ordered
No.  Type    Event Type          Time                    Registered Name
1    applet  snmp                Thu May30 05:57:16 2004 memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val
{5120000} poll-interval 90
action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
2    applet  syslog              Wed Jul16 00:05:17 2004 intf-down
 pattern {.*UPDOWN.*Ethernet1/0.*}
 action 1.0 cns-event msg "Interface state change: $_syslog_msg"
```

Use this command with the **name-ordered** keyword to display information about currently registered policies sorted by name, for example:

Example:

```
Device# show event manager policy registered name-ordered
No.  Type    Event Type          Time Registered          Name
1    applet  syslog              Wed Jul16  00:05:17 2004 intf-down
 pattern {.*UPDOWN.*Ethernet1/0.*}
 action 1.0 cns-event msg "Interface state change: $_syslog_msg"
2    applet  snmp                Thu May30 05:57:16 2004   memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val
{5120000} poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
```

Use this command with the **event-type**keyword to display information about currently registered policies for the event type specified in the *event-name* argument, for example:

Example:

```
Device# show event manager policy registered event-type syslog
No.  Type    Event Type          Time Registered           Name
1    applet  syslog              Wed Jul16  00:05:17 2004 intf-down
```

```
pattern {.*UPDOWN.*Ethernet1/0.*}
action 1.0 cns-event msg "Interface state change: $_syslog_msg"
```

Configuring Event SNMP Notification

Perform this task to configure SNMP notifications.

Before you begin

- SNMP event manager must be configured using the **snmp-server manager** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event** [**tag** *event-tag*] **snmp-notification oid** *oid-string* **oid-val** *comparison-value* **op** *operator* [**maxrun** *maxruntime-number*] [**src-ip-address** *ip-address*] [**dest-ip-address** *ip-address*] [**default** *seconds*] [**direction** {**incoming** | **outgoing**}] [**msg-op** {**drop** | **send**}]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Registers the applet with the event manager server and enters applet configuration mode.
Step 4	**event** [**tag** *event-tag*] **snmp-notification oid** *oid-string* **oid-val** *comparison-value* **op** *operator* [**maxrun** *maxruntime-number*] [**src-ip-address** *ip-address*] [**dest-ip-address** *ip-address*] [**default** *seconds*] [**direction** {**incoming** | **outgoing**}] [**msg-op** {**drop** | **send**}] **Example:**	Specifies the event criteria for an Embedded Event Manager (EEM) applet that is run by sampling Simple Network Management Protocol (SNMP) notification.

	Command or Action	Purpose
	`Device(config-applet)# event snmp-notification dest-ip-address 192.168.1.1 oid 1 op eq oid-val 10`	
Step 5	**end** **Example:** `Device(config-applet)# end`	Exits applet configuration mode and returns to privileged EXEC mode.

Configuring Multiple Event Support

The multiple event support feature adds the ability to register multiple events in the EEM server. The multiple event support involves one or more event occurrences, one or more tracked object states, and a time period for the event to occur. The event parameters are specified in the CLI commands. The data structure to handle multiple events contains multiple event identifiers and correlation logic. This data is used to register multiple events in the EEM Server.

Setting the Event Configuration Parameters

The **trigger** command enters the trigger applet configuration mode and specifies the multiple event configuration statements for EEM applets. The trigger statement is used to relate multiple event statement using the *tag* argument specified in each event statement. The events are raised based on the specified parameters.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event** [**tag** *event-tag*] **cli pattern** *regular-expression* **sync** {**yes** | **no skip** {**yes** | **no**}} [**occurs** *num-occurrences*] [**period** *period-value*] [**maxrun** *maxruntime-number*]
5. **trigger** [**occurs** *occurs-value*] [**period** *period-value*] [**period-start** *period-start-value*] [**delay** *delay-value*]
6. **correlate** {**event** *event-tag* | **track** *object-number*} [*boolean-operator* **event** *event-tag*]
7. **attribute tag** *event-tag* [**occurs** *occurs-value*]
8. **action** *label* **cli command** *cli-string*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet EventInterface`	Registers an applet with EEM and enters applet configuration mode.
Step 4	**event** [**tag** *event-tag*] **cli pattern** *regular-expression* **sync** {**yes** | **no skip** {**yes** | **no**}} [**occurs** *num-occurrences*] [**period** *period-value*] [**maxrun** *maxruntime-number*] **Example:** `Device(config-applet)# event tag 1.0 cli pattern "show bgp all" sync yes occurs 32 period 60 maxrun 60`	Specifies the event criteria for an EEM applet that is run by matching a Cisco IOS command-line interface (CLI) command.
Step 5	**trigger** [**occurs** *occurs-value*] [**period** *period-value*] [**period-start** *period-start-value*] [**delay** *delay-value*] **Example:** `Device(config-applet)# trigger occurs 1 period-start "0 8 * * 1-5" period 60`	Specifies the complex event configuration parameters for an EEM applet.
Step 6	**correlate** {**event** *event-tag* | **track** *object-number*} [*boolean-operator* **event** *event-tag*] **Example:** `Device(config-applet)# correlate event 1.0 or event 2.0`	Specifies a complex event correlation in the trigger mode for an EEM applet. **Note** When "and" is used to group events such as traps or syslog messages, then the default trigger occurrence window is three minutes.
Step 7	**attribute tag** *event-tag* [**occurs** *occurs-value*] **Example:** `Device(config-applet)# attribute tag 1.0 occurs 1`	Specifies up to eight attribute statements to build a complex event for an EEM applet.
Step 8	**action** *label* **cli command** *cli-string* **Example:** `Device(config-applet)# action 1.0 cli command "show pattern"`	Specifies the action of executing a CLI command when an EEM applet is triggered.

Examples

In the following example, applet is run if the **show bgp all** CLI command and any syslog message that contains the string "COUNT" occurred within a period 60 seconds.

```
event manager applet delay_50
 event tag 1.0 cli pattern "show bgp all" sync yes occurs 32 period 60 maxrun 60
 event tag 2.0 syslog pattern "COUNT"
 trigger occurs 1 delay 50
```

```
 correlate event 1.0 or event 2.0
 attribute tag 1.0 occurs 1
 attribute tag 2.0 occurs 1
action 1.0 cli command "show pattern"
action 2.0 cli command "enable"
action 3.0 cli command "config terminal"
action 4.0 cli command " ip route 192.0.2.0 255.255.255.224 192.0.2.12"
action 91.0 cli command "exit"
action 99.0 cli command "show ip route | incl 192.0.2.5"
```

Configuring EEM Class-Based Scheduling

To schedule Embedded Event Manager (EEM) policies and set policy scheduling options, perform this task. In this task, two EEM execution threads are created to run applets assigned to the default class.

The EEM policies will be assigned a class using the **class** keyword when they are registered. EEM policies registered without a class will be assigned to the default class. Threads that have default class, will service the default class when the thread is available for work. Threads that are assigned specific class letters will service any policy with a matching class letter when the thread is available for work.

If there is no EEM execution thread available to run the policy in the specified class and a scheduler rule for the class is configured, the policy will wait until a thread of that class is available for execution. Synchronous policies that are triggered from the same input event should be scheduled in the same execution thread.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager scheduler** {**applet** | **axp** | **call-home**} **thread class** *class-options* **number** *thread-number*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager scheduler** {**applet** | **axp** | **call-home**} **thread class** *class-options* **number** *thread-number* **Example:** `Device(config)# event manager scheduler applet thread class default number 2`	Schedules EEM policies and sets policy scheduling options. • In this example, two EEM execution threads are created to run applets assigned to the default class.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Holding a Scheduled EEM Policy Event or Event Queue

To hold a scheduled EEM policy event or event queue in the EEM scheduler, perform this task. In this task, all pending EEM policies are displayed. A policy identified using a job ID of 2 is held in the EEM scheduler, and the final step shows that the policy with a job ID of 2 has changed status from pending to held.

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending** [**queue-type**{**applet** | **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]
3. **event manager scheduler hold** {**all**| **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary**| **rp_standby**}]
4. **show event manager policy pending** [**queue-type**{**applet** | **call-home** | **axp** | **script**} **class** *class-options* | **detailed**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy pending** [**queue-type**{**applet**	**call-home**	**axp**	**script**} **class** *class-options*	**detailed**] **Example:** `Device# show event manager policy pending`	Displays the pending EEM policies.
Step 3	**event manager scheduler hold** {**all**| **policy** *job-id*	**queue-type** {**applet**	**call-home**	**axp**	**script**} **class** *class-options*} [**processor** {**rp_primary**| **rp_standby**}] **Example:** `Device# event manager scheduler hold policy 2`	Holds a scheduled EEM policy event or event queue in the EEM scheduler. • In this example, a policy with a job ID of 2 is put on hold.
Step 4	**show event manager policy pending** [**queue-type**{**applet**	**call-home**	**axp**	**script**} **class** *class-options*	**detailed** **Example:** `Device# show event manager policy pending`	Displays the status of EEM policy put on hold in Step 3 as held, along with other pending policies.

Examples

The following example shows how to view all pending EEM policies and to hold the EEM policy with a job ID of 2.

```
Device# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
Device# event manager scheduler hold policy 2
Device# show event manager policy pending

no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2  2      held   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

Resuming Execution of EEM Policy Events or Event Queues

To resume the execution of specified EEM policies, perform this task. In this task, the policy that was put on hold in the Holding a Scheduled EEM Policy Event or Event Queue task is now allowed to resume execution.

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler release** {**all** | **policy** *policy-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**}} **class** *class-options* [**processor** {*rp_primary* | *rp_standby*}]
4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	Displays the pending and held EEM policies. **Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.
Step 3	**event manager scheduler release** {**all** | **policy** *policy-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**}} **class** *class-options* [**processor** {*rp_primary* | *rp_standby*}] **Example:** `Device# event manager scheduler release policy 2`	Resumes execution of specified EEM policies. • The example shows how to resume the execution of the policy with job ID of 2.

	Command or Action	Purpose
Step 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	Displays the status of the EEM policy resumed in Step 3 as pending, along with other pending policies. **Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.

Examples

The following example shows how to view all pending EEM policies, to specify the policy that will resume execution, and to see that the policy is now back in a pending status.

```
Device# show event manager policy pending

no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2  2      held   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
Rotuer# event manager scheduler release policy 2
Rotuer# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

Clearing Pending EEM Policy Events or Event Queues

Perform this task to clear EEM policies that are executing or pending execution. In this task, the EEM policy with a job ID of 2 is cleared from the pending queue. The **show event manager policy pending** command is used to display the policies that are pending before and after the policy is cleared.

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler clear** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary** | **rp_standby**}]
4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy pending**	Displays the pending EEM policies.

	Command or Action	Purpose
	Example: `Device# show event manager policy pending`	**Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.
Step 3	**event manager scheduler clear** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary** | **rp_standby**}] **Example:** `Device# event manager scheduler clear policy 2`	Clears EEM policies that are executing or pending execution. • In this example, the EEM policy with a job ID of 2 is cleared from the pending queue.
Step 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	Displays all the pending EEM policies except the policy cleared in Step 3. **Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.

Examples

The following example shows how to clear the EEM policy with a job ID of 2 that was pending execution. The **show** commands are used to display the policies that are pending before and after the policy is cleared.

```
Device# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three

Device# event manager scheduler clear policy 2
Device# show event manager policy pending

no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

Modifying the Scheduling Parameters of EEM Policy Events or Event Queues

To modify the scheduling parameters of the EEM policies, perform this task. The **show event manager policy pending** command displays policies that are assigned to the B or default class. All the currently pending policies are then changed to class A. After the configuration modification, the **show event manager policy pending** command shows all policies assigned as class A.

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler modify** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} | **class** *class-options*} [**queue-priority** {**high** | **last** | **low** | **normal**}][**processor** {**rp_primary** | **rp_standby**}]

4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	Displays the pending EEM policies. **Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.
Step 3	**event manager scheduler modify** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} | **class** *class-options*} [**queue-priority** {**high** | **last** | **low** | **normal**}][**processor** {**rp_primary** | **rp_standby**}] **Example:** `Device# event manager scheduler modify all class A`	Modifies the scheduling parameters of the EEM policies. • In this example, all currently pending EEM policies are assigned to class A.
Step 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	Displays the EEM policies modified in Step 3 along with other pending policies. **Note** Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS Network Management Command Reference.

Examples

The following example shows how to modify the scheduling parameters of the EEM policies. In this example, the **show event manager policy pending** command displays policies that are assigned to the B or default class. All the currently pending policies are then changed to class A. After the configuration modification, the **show event manager policy pending** command verifies that all policies are now assigned as class A.

```
Device# show event manager policy pending
no. class   status time of event             event type     name
1   default pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   default pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   B       pend   Thu Sep 7  02:54:04 2006  syslog         applet: three

Device# event manager scheduler modify all class A
Device# show event manager policy pending

no. class status time of event             event type     name
1   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
```

```
2   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

Verifying Class-Based Scheduled Activities of EEM Policies

To verify the scheduled activities of the EEM policies, use the **show event manager scheduler**command.

SUMMARY STEPS

1. **show event manager scheduler thread** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager scheduler thread [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

This command displays all the EEM execution threads from the scheduler perspective and the details of the running policies. This command includes **detailed** and **queue-type** optional keywords. The following is sample output from this command:

Example:

```
Device# show event manager scheduler thread
1 Script threads service class default
  total: 1 running: 1 idle: 0
2 Script threads service class range A-D
  total: 3 running: 0 idle: 3
3 Applet threads service class default
  total: 32 running: 0 idle: 32
4 Applet threads service class W X
  total: 5 running: 0 idle: 5
```

To display the details of the running policies using the scheduler threads use the **detailed** keyword. The following is sample output for this keyword:

Example:

```
Device# show event manager scheduler thread detailed

1 Script threads service class default
total: 5 running: 5 idle: 0
1 job id: 12341, pid: 101, name: loop.tcl
2 job id: 12352, pid: 52, name: loop.tcl
3 job id: 12363, pid: 55, name: loop.tcl
4 job id: 12395, pid: 53, name: loop.tcl
5 job id: 12588, pid: 102, name: loop.tcl
2 Applet threads service class default
total: 32 running: 5 idle: 27
1 job id: 15585, pid: 104, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 job id: 15586, pid: 105, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 job id: 15587, pid: 106, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 job id: 15589, pid: 107, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 job id: 15590, pid: 80, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

To display the scheduler threads of a queue-type use the **queue-type** keyword. The following are the sample output for this keyword:

Example:

```
Device# show event manager sched thread queue-type applet
1 Applet threads service class default
total: 32 running: 7 idle: 25
Device# show event manager sched thread queue-type applet detailed
1 Applet threads service class default
total: 32 running: 5 idle: 27
1 job id: 15700, pid: 103, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 job id: 15701, pid: 104, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 job id: 15703, pid: 106, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 job id: 15704, pid: 107, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 job id: 15706, pid: 55, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

Verifying Class-Based Active EEM Policies

To verify the active or the running EEM policies, use the **show event manager policy active** command.

SUMMARY STEPS

1. **show event manager policy active** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager policy active [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

This command displays only the running EEM policies. This command includes **class**, **detailed** and **queue-type** optional keywords. The following is sample output from this command:

Example:

```
Device# show event manager policy active
no. job id p s status time of event event type name
1 12598 N A running Mon Oct29 20:49:37 2007 timer watchdog loop.tcl
2 12609 N A running Mon Oct29 20:49:42 2007 timer watchdog loop.tcl
3 12620 N A running Mon Oct29 20:49:46 2007 timer watchdog loop.tcl
4 12650 N A running Mon Oct29 20:49:59 2007 timer watchdog loop.tcl
5 12842 N A running Mon Oct29 20:51:13 2007 timer watchdog loop.tcl
default class - 6 applet events
no. job id p s status time of event event type name
1 15852 N A running Mon Oct29 21:11:09 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 15853 N A running Mon Oct29 21:11:09 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 15854 N A running Mon Oct29 21:11:10 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 15855 N A running Mon Oct29 21:11:10 2007 timer watchdog WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 15856 N A running Mon Oct29 21:11:11 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
6 15858 N A running Mon Oct29 21:11:11 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

Verifying Pending EEM Policies

To verify the EEM policies that are pending for execution, use the **show event manager policy pending** command. Use the optional keywords to specify EEM class-based scheduling options.

SUMMARY STEPS

1. **show event manager policy pending** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager policy pending [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

This command displays only the pending policies. This command includes **class**, **detailed** and **queue-type** optional keywords. The following is sample output from this command:

Example:

```
Device# show event manager policy pending
no. job id p s status time of event event type name
1    12851   N A    pend    Mon Oct29   20:51:18 2007   timer watchdog   loop.tcl
2    12868   N A    pend    Mon Oct29   20:51:24 2007   timer watchdog   loop.tcl
3    12873   N A    pend    Mon Oct29   20:51:27 2007   timer watchdog   loop.tcl
4    12907   N A    pend    Mon Oct29   20:51:41 2007   timer watchdog   loop.tcl
5    13100   N A    pend    Mon Oct29   20:52:55 2007   timer watchdog   loop.tcl
```

Configuring EEM Applet (Interactive CLI) Support

The synchronous applets are enhanced to support interaction with the local console (tty) using two commands, **action gets** and **action puts**, and these commands allow users to enter and display input directly on the console. The output for synchronous applets will bypass the system logger. The local console will be opened by the applets and serviced by the corresponding synchronous Event Detector pty. Synchronous output will be directed to the opened console.

Reading and Writing Input from the Active Console for Synchronous EEM Applets

Use the following tasks to implement EEM applet interactive CLI support:

Reading Input from the Active Console

When a synchronous policy is triggered, the related console is stored in the publish information specification. The policy director will query this information in an event_reqinfo call, and store the given console information for use by the **action gets** command.

The **action gets** command reads a line of the input from the active console and stores the input in the variable. The trailing new line will not be returned.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **gets** *variable*
6. **action** *label* **syslog** [**priority** *priority-level* **msg** *msg-text*

7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet action`	Registers the applet with the EEM and enters applet configuration mode.
Step 4	**event none** **Example:** `Device(config-applet)# event none`	Specifies that an EEM policy is to be registered with the EEM and can be run manually.
Step 5	**action** *label* **gets** *variable* **Example:** `Device(config-applet)# action label2 gets input`	Gets input from the local console in a synchronous applet and stores the value in the given variable when an EEM applet is triggered.
Step 6	**action** *label* **syslog** [**priority** *priority-level* **msg** *msg-text* **Example:** `Device(config-applet)# action label3 syslog msg "Input entered was \"$input\""`	Specifies the action to be taken when an EEM applet is triggered. • In this example, the action to be taken is to write the value of the variable specified in Step 5, to syslog.
Step 7	**exit** **Example:** `Device(config-applet)# exit`	Exits applet configuration mode and returns to privileged EXEC mode.

Example

The following example shows how to get the input from the local tty in a synchronous applet and store the value

```
Device(config)# event manager applet action
```

```
Device(config-applet)# event none
Device(config-applet)# action label2 gets input
```

Device(config-applet)# **action label3 syslog msg "Input entered was \"$input\""**

Writing Input to the Active Console

When a synchronous policy is triggered, the related console is stored in the publish information specification. The policy director will query this information in an event_reqinfo call, and store the given console information for use by the **action puts** command.

The **action puts** command will write the string to the active console. A new line will be displayed unless the **nonewline** keyword is specified. The output from the **action puts** command for a synchronous applet is displayed directly to the console, bypassing the system logger. The output of the **action puts** command for an asynchronous applet is directed to the system logger.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]
6. **action** *label* **puts** [**nonewline**] *string*
7. **exit**
8. **event manager run** *applet-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet action`	Registers the applet with the EEM and enters applet configuration mode.
Step 4	**event none** **Example:** `Device(config-applet)# event none`	Specifies that an EEM policy is to be registered with the EEM and can be run manually.

	Command or Action	Purpose
Step 5	**action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]] **Example:** `Device(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1`	Specifies the action to match the regular expression pattern on an input string when an EEM applet is triggered.
Step 6	**action** *label* **puts** [**nonewline**] *string* **Example:** `Device(config-applet)# action 2 puts "match is $_match"`	Specifies the action of printing data directly to the local console when an EEM applet is triggered. • The **nonewline** keyword is optional and is used to suppress the display of the new line character.
Step 7	**exit** **Example:** `Device(config-applet)# exit`	Exits applet configuration mode and returns to privileged EXEC mode.
Step 8	**event manager run** *applet-name* **Example:** `Device# event manager run action`	Manually runs a registered EEM policy. • In this example, the policy registered in Step 3 is triggered and the associated actions specified in Step 5 and Step 6 are executed.

Example

The following example shows how the **action puts** command prints data directly to the local console:

```
Device(config-applet)# event manager applet puts
Device(config-applet)# event none
Device(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
Device(config-applet)# action 2 puts "match is $_match"
Device(config-applet)# action 3 puts "submatch 1 is $_sub1"
Device# event manager run puts
match is one two three
submatch 1 is one
```

Configuring SNMP Library Extensions

Depending on your release, the SNMP Library Extensions feature allows you to perform the following configurations.

Prerequisites

To use this feature, you must be running Cisco IOS Release 12.4(22)T or a later release.

SNMP Get and Set Operations

The SNMP Library Extensions feature extends the EEM applet **action info** and Tcl **sys_reqinfo_snmp**commands to include functionality for SNMP get-one, get-next, getid and set-any operations.

SNMP Get Operation

The SNMP event manager performs the SNMP get operation to retrieve one or more variables for the managed objects. Using the **action info type snmp oid get-type**and **action info type snmp getid**commands, you can configure the SNMP event manager to send an SNMP get request by specifying the variables to retrieve, and the IP address of the agent.

For example, if you want to retrieve the variable with the OID value of 1.3.6.1.2.1.1.1, you should specify the variable value, that is 1.3.6.1.2.1.1.1. If the specified values do not match, a trap will be generated and an error message will be written to the syslog history.

The **action info type snmp oid get-type** command specifies the type of the get operation to be performed. To retrieve the exact variable, the get operation type should be specified as **exact**. To retrieve a lexicographical successor of the specified OID value, the get operation type should be set to **next**.

The table below shows the built-in variables, in which the values retrieved from SNMP get operation are stored.

Table 54: Built-in Variables for action info type snmp oid Command

Built-in Variable	Description
_info_snmp_oid	The SNMP object ID.
_info_snmp_value	The value string of the associated SNMP data element.

GetID Operation

The **action info type snmp getid** command retrieves the following variables from the SNMP entity:

- sysDescr.0
- sysObjectID.0
- sysUpTime.0
- sysContact.0
- sysName.0
- sysLocation.0

The table below shows the built-in variables, in which the values retrieved from the SNMP getID operation are stored.

Table 55: Built-in Variables for action info type snmp getid Command

Built-in Variable	Description
_info_snmp_syslocation_oid	The OID value of the sysLocation variable.
_info_snmp_syslocation_value	The value string for the sysLocation variable.

Built-in Variable	Description
_info_snmp_sysdescr_oid	The OID value of the sysDescr variable.
_info_snmp_sysdescr_value	The value string for the sysDescr variable.
_info_snmp_sysobjectid_oid	The OID value of the sysObjectID variable.
_info_snmp_sysobjectid_value	The value string for the sysObjectID variable.
_info_snmp_sysuptime_oid	The OID value of the sysUptime variable.
_info_snmp_sysuptime_value	The value string for the sysUptime variable.
_info_snmp_syscontact_oid	The OID value of the sysContact variable.
_info_snmp_syscontact_value	The value string for the sysContact variable.

The get operation requests can be sent to both local and remote hosts.

SNMP Set Operation

All SNMP variables are assigned a default value in the MIB view. The SNMP event manager can modify the value of these MIB variables through set operation. The set operation can be performed only on the system that allows read-write access.

To perform a set operation, you must specify the type of the variable and the value associated with it.

The table below shows the valid OID types and values for each OID type.

Table 56: OID Type and Value for Set Operation

OID Type	Description
counter32	A 32-bit number with a minimum value of 0. V value in the range from 0 to 4294967295 is v
gauge	A 32-bit number with a minimum value of 0. gauge object type. Integer value in the range
integer	A 32-bit number used to specify a numbered the operational status of a device interface, 1 from 0 to 4294967295 is valid.
ipv4	IP version 4 address. IPv4 address in dotted d
octet string	An octet string in hexadecimal notation used
string	An octet string in text notation used to repres
unsigned32	A 32-bit number used to represent decimal va valid.

The set operation can carried out on both local and remote hosts.

SNMP Traps and Inform Requests

Traps are SNMP notifications that alert the SNMP manager or the NMS to a network condition.

SNMP inform requests refer to the SNMP notifications that alert the SNMP manager to a network condition and request for confirmation of receipt from the SNMP manager.

An SNMP event occurs when SNMP MIB object ID values are sampled, or when the SNMP counter crosses a defined threshold. If the notifications are enabled and configured for such events, the SNMP traps or inform messages generated. An SNMP notification event is triggered when an SNMP trap or inform message is received by the event manager server.

To send an SNMP trap or inform message when an Embedded Event Manager (EEM) applet is triggered, the **action info type snmp trap** and **action info type snmp inform**commands are used. The CISCO-EMBEDDED-EVENT-MGR-MIB.my is used to define the trap and inform messages.

Configuring EEM Applet for SNMP Get and Set Operations

While registering a policy with the event manager server, the actions associated with an SNMP event can be configured.

Perform this task to configure EEM applet for SNMP set and get operations.

Before you begin

- SNMP event manager must be configured using the **snmp-server manager**command.
- The SNMP community string should be set by using the **snmp-server community** command to enable access to the SNMP entity.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. Do one of the following:
   - **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
5. **action label info type snmp oid** *oid-value* **get-type** {**exact**| **next**} [**community** *community-string*] [**ipaddr** *ip-address*]
6. **action label info type snmp oid** *oid-value* **set-type** *oid-type oid-type-value* **community** *community-string* [**ipaddr** *ip-address*]
7. **action label info type snmp getid** *oid-value* [**community** *community-string*] [**ipaddr** *ip-address*]
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Registers the applet with the event manager server and enters applet configuration mode.
Step 4	Do one of the following: • **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value* **Example:** `Device(config-applet)# event snmp oid` **Example:** `1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact` **Example:** `entry-op lt entry-val 5120000 poll-interval 90`	Specifies the event criteria that cause the EEM applet to run. • In this example, an EEM event is triggered when free memory falls below the value of 5120000. • Exit criteria are optional, and if not specified, event monitoring is reenabled immediately.
Step 5	**action label info type snmp oid** *oid-value* **get-type** {**exact**| **next**} [**community** *community-string*] [**ipaddr** *ip-address*] **Example:** `Device(config-applet)# action 1.3 info type` **Example:** `snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type` **Example:** `exact community public ipaddr 172.17.16.69`	Specifies the type of get operation to perform. • In this example, the type of get operation is specified as exact and community string is specified as public.
Step 6	**action label info type snmp oid** *oid-value* **set-type** *oid-type oid-type-value* **community** *community-string* [**ipaddr** *ip-address*] **Example:** `Device(config-applet)# action 1.4 info type`	(Optional) Specifies the variable to be set. • In this example, the sysName.0 variable is specified for the set operation and community string is specified as rw.

	Command or Action	Purpose
	Example: `snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 set-type` **Example:** `integer 42220 sysName.0 community rw ipaddr` **Example:** `172.17.16.69`	**Note** For set operation, you must specify the SNMP community string.
Step 7	**action label info type snmp getid** *oid-value* [**community** *community-string*] [**ipaddr** *ip-address*] **Example:** `Device(config-applet)# action 1.3 info type` **Example:** `snmp getid community public ipaddr 172.17.16.69`	(Optional) Specifies if the individual variables should be retrieved by the getid operation.
Step 8	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring EEM Applet for SNMP OID Notifications

Perform this task to configure SNMP notifications.

Before you begin

- SNMP event manager must be configured using the **snmp-server manager**command and SNMP agents must be configured to send and receive SNMP traps generated for an EEM policy.
- SNMP traps and informs must be enabled by using the **snmp-server enable traps event-manager** and **snmp-server enable traps** commands, to allow traps and inform requests to be sent from the device to the event manager server.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. Do one of the following:
   - **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*

5. **action label info type snmp var** *variable-name* **oid** *oid-value* *oid-type oid-type-value*
6. **action label info type snmp trap enterprise-oid** *enterprise-oid-value* **generic-trapnum** *generic-trap-number* **specific-trapnum** *specific-trap-number* **trap-oid** *trap-oid-value* **trap-var** *trap-variable*
7. **action label info type snmp inform trap-oid** *trap-oid-value* **trap-var** *trap-variable* **community** *community-string* **ipaddr** *ip-address*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Registers the applet with the event manager server and enters applet configuration mode.
Step 4	Do one of the following: • **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value* **Example:** `Device(config-applet)# event snmp oid` **Example:** `1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact` **Example:** `entry-op lt entry-val 5120000 poll-interval 90`	Specifies the event criteria that cause the EEM applet to run. • In this example, an EEM event is triggered when free memory falls below the value of 5120000. • Exit criteria are optional, and if not specified, event monitoring is reenabled immediately.
Step 5	**action label info type snmp var** *variable-name* **oid** *oid-value* *oid-type oid-type-value* **Example:** `Device(config-applet)# action 1.3 info type` **Example:**	Specifies the instance of a managed object and its value. • In this example, the sysDescr.0 variable is used.

	Command or Action	Purpose
	`snmp var sysDescr.0 oid` **Example:** `1.3.6.1.4.1.9.9.48.1.1.1.6.1 integer 4220`	
Step 6	**action label info type snmp trap enterprise-oid** *enterprise-oid-value* **generic-trapnum** *generic-trap-number* **specific-trapnum** *specific-trap-number* **trap-oid** *trap-oid-value* **trap-var** *trap-variable* **Example:** `Device(config-applet)# action 1.4 info type` **Example:** `snmp trap enterprise-oid 1.3.6.1.4.1.1` **Example:** `generic-trapnum 4 specific-trapnum 7 trap-oid` **Example:** `1.3.6.1.4.1.1.226.0.2.1 trap-var sysUpTime.0`	Generates an SNMP trap when the EEM applet is triggered. • In this example, the authenticationFailure trap is generated. **Note** The specific trap number refers to the enterprise-specific trap, which is generated when an enterprise event occurs. If the generic trap number is not set to 6, the specific trap number you specify will be used to generate traps.
Step 7	**action label info type snmp inform trap-oid** *trap-oid-value* **trap-var** *trap-variable* **community** *community-string* **ipaddr** *ip-address* **Example:** `Device(config-applet)# action 1.4 info type` **Example:** `snmp inform trap-oid 1.3.6.1.4.1.1.226.0.2.1` **Example:** `trap-var sysUpTime.0 community public ipaddr` **Example:** `172.69.16.2`	Generates an SNMP inform request when the EEM applet is triggered. • In this example, the inform request is generated for the sysUpTime.0 variable.
Step 8	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged mode.

Configuring Variable Logic for EEM Applets

The Variable Logic for EEM Applets feature adds the ability to apply conditional logic within EEM applets. Before variable logic is introduced, applets have a linear structure where each action is executed in the order in which they are configured when the event is triggered. Conditional logic introduces a control structure that can change the flow of actions within applets depending on conditional expressions. Each control structure can contain a list of applet actions including looping and if/else actions which determine if the structure is executed or not.

The information in applet configuration mode is presented as background to set the context for the action commands.

To provide a consistent user interface between the Tool Command Language (Tcl) and the applet (CLI) based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl based implementation.
- Event specification data is written using the CLI applet submode configuration statements in the applet-based implementation.

Applet configuration mode is entered using the event manager applet command. In applet configuration mode the config prompt changes to (config-applet)#. In applet configuration mode two types of config statements are supported:

- event - used to specify the event criteria to cause this applet to run.
- action - used to specify a built-in action to perform.

Multiple **action** applet config commands are allowed within an applet configuration. If no **action** applet config command is present, a warning is displayed, upon exit, stating no statements are associated with this applet. When no statements are associated with this applet, events get triggered but no action is taken. If no commands are specified in applet configuration mode, the applet will be removed upon exit. The exit applet config command is used to exit from applet configuration mode.

Depending on your release, the Variable Logic for EEM Applets feature allows you to perform the following configurations.

Prerequisites

To use this feature, you must be running Cisco IOS Release 12.4(22)T or a later release.

Configuring Variable Logic for EEM Applets

EEM 3.0 adds new applet action commands to permit simple variable logic within applets.

To configure the variable logic using action commands perform the following tasks.

Specifying a Loop of Conditional Blocks

To specify a loop of a conditional block when an EEM applet is triggered, perform this task. In this task, a conditional loop is set to check if the value of the variable is less than 10. If the value of the variable is less than 10, then the message ‘i is $_i’ is written to the syslog.

Note Depending on your release, the **set** (EEM) command is replaced by the **action set** command. See the **action** *label* **set** command for more information. If the set (EEM) command is entered in certain releases, the IOS parser translates the **set** command to the **action** *label* **set** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set**
5. **action** *label* **while** *string_op1 operator string_op2*
6. Add any action as required.
7. **action** *label* **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet condition`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**action** *label* **set** **Example:** `Device(config-applet)#action 1.0 set i 2`	Sets an action for the event. • In this example, the value of the variable i is set to 2.
Step 5	**action** *label* **while** *string_op1 operator string_op2* **Example:** `Device(config-applet)# action 2 while $i lt 10`	Specifies a loop of a conditional block. • In this example, a loop is set to check if the value of the variable i is less than 10.
Step 6	Add any action as required. **Example:**	Performs the action as indicated by the action command. • In this example, the message 'i is $_i' is written to the syslog.

	Command or Action	Purpose
	`Device(config-applet)# action 3 syslog msg "i is $i"`	
Step 7	**action** *label* **end** **Example:** `Device(config-applet)# action 3 end`	Exits from the running action.

Specifying if else Conditional Blocks

To specify the beginning of an if conditional statement followed by an else conditional statement, perform this task. The if or else conditional statements can be used in conjunction with each other or separately. In this task, the value of a variable is set to 5. An if conditional block is then specified to check if the value of the variable is less than 10. Provided the if conditional block is satisfied, an action command to output the message ‘x is less than 10’ is specified.

Following the if conditional block, an else conditional block is specified. Provided the if conditional block is not satisfied, an action command to output the message ‘x is greater than 10’ is specified.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set** *variable-name variable-value*
5. **action** *label* **if** [*stringop1*] {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} [*stringop2*]
6. Add any action as required.
7. **action** *label* **else**
8. Add any action as required.
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:**	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.

	Command or Action	Purpose
	`Device(config)# event manager applet ifcondition`	
Step 4	**action** *label* **set** *variable-name variable-value* **Example:** `Device(config-applet)# action 1.0 set x 5`	Sets an action for the event. • In this example, the value of the variable x is set to 5.
Step 5	**action** *label* **if** [*stringop1*] {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} [*stringop2*] **Example:** `Device(config-applet)# action 2.0 if $x lt 10`	Specifies an if conditional statement. • In this example, an if conditional statement to check if the value of the variable is less than 10.
Step 6	Add any action as required. **Example:** `Device(config-applet)# action 3.0 puts "$x is less than 10"`	Performs the action as indicated by the action command. • In this example, the message '5 is less than 10' is displayed on the screen.
Step 7	**action** *label* **else** **Example:** `Device(config-applet)# action 4.0 else`	Specifies an else conditional statement
Step 8	Add any action as required. **Example:** `Device(config-applet)# action 5.0`	Performs the action as indicated by the action command. • In this example, the message '5 is greater than 10' is displayed on the screen.
Step 9	**end** **Example:** `Device(config-applet)# end`	Exits from the running action.

Specifying foreach Iterating Statements

To specify a conditional statement that iterates over an input string using the delimiter as a tokenizing pattern, perform this task. The foreach iteration statement is used to iterate through a collection to get the desired information. The delimiter is a regular expression pattern string. The token found in each iteration is assigned to the given iterator variable. All arithmetic calculations are performed as long integers with out any checks for overflow. In this task, the value of the variable x is set to 5. An iteration statement is set to run through the input string red, blue, green, orange. For every element in the input string, a corresponding message is displayed on the screen.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **event manager applet** *applet-name*
4. **action** *label* **foreach** [*string-iterator*] [*string-input*] [*string-delimiter*]
5. Specify any action command
6. **action** *label* **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet iteration`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**action** *label* **foreach** [*string-iterator*] [*string-input*] [*string-delimiter*] **Example:** `Device(config-applet)# action 2.0 foreach iterator "red blue green orange"`	Iterates over an input string using the delimter as a tokenizing pattern. • In this example, the iteration is run through the elements of the input string - red, blue, green and orange.
Step 5	Specify any action command **Example:** `Device(config-applet)# action 3.0 puts "Iterator is $iterator"`	Performs the action as indicated by the action command. • In this example, the following message is displayed on the screen: Iterator is red Iterator is blue Iterator is green Iterator is orange
Step 6	**action** *label* **end** **Example:** `Device(config-applet)# action 4.0 end`	Exits from the running action.

Using Regular Expressions

To match a regular expression pattern with an input string, perform this task. Using regular expressions, you can specify the rules for a set of possible strings to be matched.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet regexp`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]] **Example:** `Device(config-applet)# action 2.0 regexp "(.*) (.*) (.*)" "red blue green" _match _sub1`	Specifies an expression pattern to match with an input string. • In this example, an input string of 'red blue green' is specified. When the expression pattern matches the input string, the entire result **red blue green** is stored in the variable **_match** and the submatch **red**is stored in the variable **_sub1**.

Incrementing the Values of Variables

To increment the value of variables, perform this task. In this task, the value of a variable is set to 20 and then the value is incremented by 12.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set**
5. **action** *label* **increment** *variable-name long-integer*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet increment`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**action** *label* **set** **Example:** `Device(config-applet)#action 1.0 set varname 20`	Sets an action for the event. • In this example, the value of the variable is set to 20.
Step 5	**action** *label* **increment** *variable-name long-integer* **Example:** `Device(config-applet)#action 2.0 increment varname 12`	Increments the value of variable by the specified long integer. • In this example, the value of the variable is incremented by 12.

Configuring Event SNMP Object

Perform this task to register the Simple Network Management Protocol (SNMP) object event for an Embedded Event Manager (EEM) applet that is run by sampling SNMP object.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event snmp-object oid** *oid-value* **type** *value* **sync** {**yes** | **no**} **skip** {**yes** | **no**} **istable** {**yes** | **no**} [**default** *seconds*] [**maxrun** *maxruntime-number*]

5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet manual-policy`	Registers the applet with the Embedded Event Manager and enters applet configuration mode.
Step 4	**event snmp-object oid** *oid-value* **type** *value* **sync** {**yes** | **no**} **skip** {**yes** | **no**} **istable** {**yes** | **no**} [**default** *seconds*] [**maxrun** *maxruntime-number*] **Example:** `Device(config-applet)# event snmp-object oid 1.9.9.9.9 type gauge sync yes` **Example:** `action 1 syslog msg "oid = $_snmp_oid"` **Example:** `action 2 syslog msg "request = $_snmp_request"` **Example:** `action 3 syslog msg "request_type = $_snmp_request_type"`	Registers the Simple Network Management Protocol (SNMP) object event for an Embedded Event Manager (EEM) applet to intercept SNMP GET and SET requests for an object. The default for this command is that it is not configured. If this command is configured the defaults are the same as in the description of the syntax options, • The **oid**keyword specifies the SNMP object identifier (object ID). • The *oid-value* argument can be the Object ID value of the data element, in SNMP dotted notation. An OID is defined as a type in the associated MIB, CISCO-EMBEDDED-EVENT-MGR-MIB, and each type has an object value. • The **istable** keyword specifies whether the OID is an SNMP table. • The sync keyword specifies that the applet is to run in synchronous mode. The return code from the applet indicates whether to reply to the SNMP request. The description for code 0 is “do not reply to the request” and the description for code 1 is “reply to the request”. When the return code from the applet replies to the request, a value is specified in the applet for the object using **action snmp-object-value** command. • The **type**keyword specifies the type of object. • The *value* argument is the value of the object.

	Command or Action	Purpose
		• Theskip keyword specifies whether to skip CLI command execution. • The **default** keyword specifies the time to process the SET or GET request normally by the applet. If the **default** keyword is not specified, the default time period is set to 30 seconds. • The *milliseconds* argument is the time period during which the SNMP Object event detector waits for the policy to exit. • The **maxrun** keyword specifies the maximum runtime of the applet. If the **maxrun** keyword is specified, the *maxruntime-number* value must be specified. If the **maxrun** keyword is not specified, the default applet run time is 20 seconds. • The *milliseconds* argument is the maximum runtime of the apple in milliseconds. If the argument is not specified, the default 20-second run-time limit is used.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Disabling AAA Authorization

Perform this task to allow EEM policies to bypass AAA authorization when triggered.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name* [**authorization bypass**] [**class** *class-options*] [**trap**]
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**event manager applet** *applet-name* [**authorization bypass**] [**class** *class-options*] [**trap**] **Example:** `Device(config)# event manager applet one class A authorization bypass`	Registers the applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**exit** **Example:** `Device(config-aaplet)# exit`	Exits device configuration applet mode and returns to privileged EXEC mode.

Configuring Description of an Embedded Event Manager Applet

Perform this task to describe an EEM applet. The description of an applet can be added in any order, before or after any other applet configuration. Configuring a new description for an applet that already has a description overwrites the current description. An applet description is optional.

Perform this task to configure a new description for an applet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **description** *line*
5. **event syslog pattern** *regular-expression*
6. **action** *label* **syslog msg** *msg-text*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet increment`	Registers the applet with the EEM and enters applet configuration mode.
Step 4	**description** *line* **Example:** `Device(config-applet)# description "This applet looks for the word count in syslog messages"`	Adds or modifies the description of an EEM applet that is run by sampling Simple Network Management Protocol (SNMP).
Step 5	**event syslog pattern** *regular-expression* **Example:** `Device(config-applet)# event syslog pattern "count"`	Specifies the event criteria for an Embedded Event Manager (EEM) applet that is run by matching syslog messages.
Step 6	**action** *label* **syslog msg** *msg-text* **Example:** `Device(config-applet)# action 1 syslog msg hi`	Specifies the action to be taken when an EEM applet is triggered. • In this example, the action taken is to write a message to syslog. • The *msg-text* argument can be character text, an environment variable, or a combination of the two.
Step 7	**end** **Example:** `Device(config-applet)# end`	Exits applet configuration mode and returns to privileged EXEC mode.

Configuration Examples for Writing EEM Policies Using the Cisco IOS CLI

Embedded Event Manager Applet Configuration Examples

The following examples show how to create an EEM applet for some of the EEM event detectors. These examples follow steps outlined in the Registering and Defining an Embedded Event Manager Applet, on page 531.

Application-Specific Event Detector

The following example shows how a policy named EventPublish_A runs every 20 seconds and publishes an event type numbered 1 to an EEM subsystem numbered 798. The subsystem value of 798 specifies that a publish event has occurred from an EEM policy. A second policy named EventPublish_B is registered to run

when the EEM event type 1 occurs with subsystem 798. When the EventPublish_B policy runs, it sends a message to syslog containing data passed as an argument from the EventPublish_A policy.

```
event manager applet EventPublish_A
 event timer watchdog time 20.0
 action 1.0 syslog msg "Applet EventPublish_A"
 action 2.0 publish-event sub-system 798 type 1 arg1 twenty
 exit
event manager applet EventPublish_B
 event application sub-system 798 type 1
 action 1.0 syslog msg "Applet EventPublish_B arg1 $_application_data1"
```

CLI Event Detector

The following example shows how to specify an EEM applet to run when the Cisco IOS **write memory** CLI command is run. The applet provides a notification that this event has occurred via a syslog message. In the example, the **sync** keyword is configured with the yes argument, and this means that the event detector is notified when this policy completes running. The exit status of the policy determines whether the CLI command will be executed. In this example, the policy exit status is set to one and the CLI command runs.

```
event manager applet cli-match
 event cli pattern "write mem.*" sync yes
 action 1.0 syslog msg "$_cli_msg Command Executed"
 set 2.0 _exit_status 1
```

The following example shows an applet which matches the **cli pattern** with the test argument. When **show access-list test** is entered, the CLI event detector matches the test argument, and the applet is triggered. The **debug event manager detector cli** output is added to show num_matches is set to one.

```
!
event manager applet EEM-PIPE-TEST
 event cli pattern "test" sync yes
 action 1.0 syslog msg "Pattern matched!"
!
*Aug 23 23:19:59.827: check_eem_cli_policy_handler: command_string=show access-lists test
*Aug 23 23:19:59.827: check_eem_cli_policy_handler: num_matches = 1, response_code = 4
*Aug 23 23:19:59.843: %HA_EM-6-LOG: EEM-PIPE-TEST: Pattern matched!
```

Note The functionality provided in the CLI event detector only allows a regular expression pattern match on a valid IOS CLI command itself. This does not include text after a pipe (|) character when redirection is used.

The following example shows that when **show version | include test** is entered, the applet fails to trigger because the CLI event detector does not match on characters entered after the pipe (|) character and the **debug event manager detector cli** output shows num_matches is set to zero.

```
*Aug 23 23:20:16.827: check_eem_cli_policy_handler: command_string=show version
*Aug 23 23:20:16.827: check_eem_cli_policy_handler: num_matches = 0, response_code = 1
```

Counter Event Detector and Timer Event Detector

The following example shows that the EventCounter_A policy is configured to run once a minute and to increment a well-known counter called critical_errors. A second policy--EventCounter_B--is registered to be triggered when the well-known counter called critical_errors exceeds a threshold of 3. When the EventCounter_B policy runs, it resets the counter to 0.

```
event manager applet EventCounter_A
 event timer watchdog time 60.0
 action 1.0 syslog msg "EventCounter_A"
 action 2.0 counter name critical_errors op inc value 1
 exit
event manager applet EventCounter_B
 event counter name critical_errors entry-op gt entry-val 3 exit-op lt exit-val 3
 action 1.0 syslog msg "EventCounter_B"
 action 2.0 counter name critical_errors op set value 0
```

Interface Counter Event Detector

The following example shows how a policy named EventInterface is triggered every time the receive_throttle counter for Fast Ethernet interface 0/0 is incremented by 5. The polling interval to check the counter is specified to run once every 90 seconds.

```
event manager applet EventInterface
 event interface name FastEthernet0/0 parameter receive_throttle entry-op ge entry-val 5
 entry-val-is-increment true poll-interval 90
 action 1.0 syslog msg "Applet EventInterface"
```

RF Event Detector

The RF event detector is only available on networking devices that contain dual Route Processors (RPs). The following example shows how to specify event criteria based on an RF state change notification:

```
event manager applet start-rf
 event rf event rf_prog_initialization
 action 1.0 syslog msg "rf state rf_prog_initialization reached"
```

RPC Event Detector

The RPC event detector allows an outside entity to make a Simple Object Access Protocol (SOAP) request to the device and invokes a defined EEM policy or script. The following example shows how an EEM applet called Event_RPC is being registered to run an EEM script:

```
event manager applet Event_RPC
 event rpc
 action print puts "hello there"
```

The following example shows the format of the SOAP request and reply message:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP:Envelope xmlns:SOAP="http://www.cisco.com/eem.xsd">
  <SOAP:Body>
    <run_eemscript>
      <script_name>Event_RPC</script_name>
    </run_eemscript>
  </SOAP:Body>
</SOAP:Envelope>
]]>]]>
<?xml version="1.0" encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.cisco.com/eem.xsd"><SOAP:Body>
<run_eemscript_response><return_code>0</return_code><output></output></run_eemscript_response></SOAP:Body></SOAP:Envelope>]]>]]>
```

SNMP Event Detector

The following example shows how to specify an EEM applet to run when the CPU usage is greater than 75 percent. When the EEM applet runs, the CLI commands **enable** and **show cpu processes** are run, and an e-mail containing the result of the **show cpu processes** command is sent to an engineer.

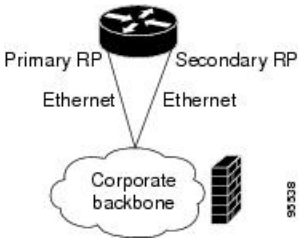
```
event manager applet snmpcpuge75
 event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3.1 get-type exact entry-op ge entry-val 75
poll-interval 10
 action 1.0 cli command "enable"
 action 2.0 cli command "show process cpu"
 action 3.0 mail server "192.168.1.146" to "engineer@cisco.com" from "devtest@cisco.com"
subject "B25 PBX Alert" body "$_cli_result"
```

The next example is more complex and shows how to configure an EEM applet that causes a switch to the secondary (redundant) Route Processor (RP) when the primary RP runs low on memory.

This example illustrates a method for taking preventative action against a software fault that causes a memory leak. The action taken here is designed to reduce downtime by switching over to a redundant RP when a possible memory leak is detected.

The figure below shows a dual RP device that is running an EEM image. An EEM applet has been registered through the CLI using the **event manager applet** command. The applet will run when the available memory on the primary RP falls below the specified threshold of 5,120,000 bytes. The applet actions are to write a message to syslog that indicates the number of bytes of memory available and to switch to the secondary RP.

Figure 15: Dual RP Topology



The commands used to register the policy are shown below.

```
event manager applet memory-demo
 event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val 5120000
poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory is
$_snmp_oid_val bytes"
 action 2.0 force-switchover
```

The registered applet is displayed using the **show event manager policy registered** command:

```
Device# show event manager policy registered
No.  Type    Event Type          Time Registered           Name
1    applet  snmp                Thu Jan30  05:57:16 2003  memory-demo
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory is
$_snmp_oid_val bytes"
 action 2.0 force-switchover
```

For the purpose of this example, a memory depletion is forced on the device, and a series of **show memory** commands are executed to watch the memory deplete:

```
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212348444   119523060    92825384    92825384    92365916
Fast        53565260      131080       70360       60720       60720       60668
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212364664   164509492    47855172    47855172    47169340
Fast        53565260      131080       70360       60720       60720       60668
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212369492   179488300    32881192    32881192    32127556
Fast        53565260      131080       70360       60720       60720       60668
```

When the threshold is reached, an EEM event is triggered. The applet named memory-demo runs, causing a syslog message to be written to the console and a switch to be made to the secondary RP. The following messages are logged:

```
00:08:31: %HA_EM-2-LOG: memory-demo: Memory exhausted; current available memory is
4484196 bytes
00:08:31: %HA_EM-6-FMS_SWITCH_HARDWARE: fh_io_msg: Policy has requested a hardware
switchover
```

The following is partial output from the **show running-config** command on both the primary RP and the secondary (redundant) RP:

```
redundancy
 mode sso
.
.
!
event manager applet memory-demo
 event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val
5120000 poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
```

SNMP Notification Event Detector

The following example shows how to configure the **snmp-server community** public RW and **snmp-server manager** commands before **event snmp-notification** is configured.

```
snmp-server community public RW
 snmp-server manager
```

The following example shows how an EEM applet called SNMP_Notification is being registered to run an EEM script when the device receives an SNMP notification on destination IP address 192.168.1.1 for object ID 1 whose value equals 10.

```
event manager applet SNMP_Notification
 event snmp-notification dest_ip_address 192.168.1.1 oid 1 op eq oid-value 10
 action 1 policy eem_script
```

Syslog Event Detector

The following example shows how to specify an EEM applet to run when syslog identifies that Ethernet interface 1/0 is down. The applet sends a message about the interface to syslog.

```
event manager applet interface-down
 event syslog pattern "\.*UPDOWN.*Ethernet1/0.*" occurs 4
 action 1.0 syslog msg "Ethernet interface 1/0 changed state 4 times"
```

Configuration Examples for Embedded Event Manager Applet

Example Identity Event Detector

The following example shows how a policy named "EventIdentity" is triggered every time the authentication on the Fast Ethernet interface 0 is success.

```
event manager applet EventIdentity
 event identity interface FastEthernet0 authc success
action 1.0 syslog msg "Applet EventIdentity"
```

Example MAT Event Detector

The following example shows how a policy named "EventMat" is triggered every time a mac-address is learned in the mac-address-table.

```
event manager applet EventMat
 event mat interface FastEthernet0
action 1.0 syslog msg "Applet EventMat"
```

Example Neighbor-Discovery Event Detector

The following example shows how a policy named "EventNeighbor" is triggered when a Cisco Discovery Protocol (CDP) cache entry changes.

```
event manager applet EventNeighbor
 event neighbor-discovery interface FastEthernet0 cdp all
action 1.0 syslog msg "Applet EventNeighbor"
```

Embedded Event Manager Manual Policy Execution Examples

The following examples show how to use the none event detector to configure an EEM policy (applet or script) to be run manually.

Using the event manager run Command

This example shows how to run a policy manually using the **event manager run** command. The policy is registered using the **event none** command under applet configuration mode and then run from global configuration mode using the **event manager run** command.

```
event manager applet manual-policy
 event none
 action 1.0 syslog msg "Manual-policy triggered"
 end
!
event manager run manual-policy
```

Using the action policy Command

This example shows how to run a policy manually using the **action policy** command. The policy is registered using the **event none** command under applet configuration mode, and then the policy is executed using the **action policy** command in applet configuration mode.

```
event manager applet manual-policy
 event none
 action 1.0 syslog msg "Manual-policy triggered"
 exit
!
event manager applet manual-policy-two
 event none
 action 1.0 policy manual-policy
 end
!
event manager run manual-policy-two
```

Embedded Event Manager Watchdog System Monitor (Cisco IOS) Event Detector Configuration Example

The following example shows how to configure three EEM applets to demonstrate how the Cisco IOS watchdog system monitor (IOSWDSysMon) event detector works.

Watchdog System Monitor Sample1 Policy

The first policy triggers an applet when the average CPU usage for the process named IP Input is greater than or equal to 1 percent for 10 seconds:

```
event manager applet IOSWD_Sample1
 event ioswdsysmon sub1 cpu-proc taskname "IP Input" op ge val 1 period 10
 action 1.0 syslog msg "IOSWD_Sample1 Policy Triggered"
```

Watchdog System Monitor Sample2 Policy

The second policy triggers an applet when the total amount of memory used by the process named Net Input is greater than 100 kb:

```
event manager applet IOSWD_Sample2
 event ioswdsysmon sub1 mem-proc taskname "Net Input" op gt val 100 is-percent false
 action 1.0 syslog msg "IOSWD_Sample2 Policy Triggered"
```

Watchdog System Monitor Sample3 Policy

The third policy triggers an applet when the total amount of memory used by the process named IP RIB Update has increased by more than 50 percent over the sample period of 60 seconds:

```
event manager applet IOSWD_Sample3
 event ioswdsysmon sub1 mem-proc taskname "IP RIB Update" op gt val 50 is-percent true
period 60
 action 1.0 syslog msg "IOSWD_Sample3 Policy Triggered"
```

The three policies are configured, and then repetitive large pings are made to the networking device from several workstations, causing the networking device to register some usage. This will trigger policies 1 and 2, and the console will display the following messages:

```
00:42:23: %HA_EM-6-LOG: IOSWD_Sample1: IOSWD_Sample1 Policy Triggered
00:42:47: %HA_EM-6-LOG: IOSWD_Sample2: IOSWD_Sample2 Policy Triggered
```

To view the policies that are registered, use the **show event manager policy registered** command:

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  ioswdsysmon         Off   Fri Jul 23 02:27:28 2004  IOSWD_Sample1
 sub1 cpu_util {taskname {IP Input} op ge val 1 period 10.000 }
 action 1.0 syslog msg "IOSWD_Sample1 Policy Triggered"
2    applet  system  ioswdsysmon         Off   Fri Jul 23 02:23:52 2004  IOSWD_Sample2
 sub1 mem_used {taskname {Net Input} op gt val 100 is_percent FALSE}
 action 1.0 syslog msg "IOSWD_Sample2 Policy Triggered"
3    applet  system  ioswdsysmon         Off   Fri Jul 23 03:07:38 2004  IOSWD_Sample3
 sub1 mem_used {taskname {IP RIB Update} op gt val 50 is_percent TRUE period 60.000 }
 action 1.0 syslog msg "IOSWD_Sample3 Policy Triggered"
```

Configuration SNMP Library Extensions Examples

SNMP Get Operations Examples

The following example shows how to send a get request to the local host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact
community
 public
Device(config-applet)# action 1.3 info type snmp oid
 1.3.6.1.2.1.1.4.0 get-type next community
public
```

The following log message will be written to the SNMP event manager log:

```
1d03h:%HA_EM-6-LOG: lg: 1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgn: 1.3.6.1.2.1.1.5.0
```

The following example shows how to send a get request to a remote host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 get-type next community
 public ipaddr
 172.17.16.69
Device(config-applet)# action 1.3 info type snmp getid
 1.3.6.1.2.1.1.1.0 community
 public ipaddr
 172.17.16.69
```

The following log message is written to the SNMP event manager log:

```
1d03h:%HA_EM-6-LOG: lg: 1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgn: 1.3.6.1.2.1.1.5.0
```

SNMP GetID Operations Examples

The following example shows how to send a getid request to the local host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp getid
 community
 public
```

The following log message is written to the SNMP event manager log:

```
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_oid=1.3.6.1.2.1.1.5.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_value=jubjub.cisco.com
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_oid=1.3.6.1.2.1.1.6.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_value=
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysdescr_oid=1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_oid=1.3.6.1.2.1.1.2.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_value=products.222
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=1.3.6.1.2.1.1.3.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=10131676
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_oid=1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_value=YYY
```

The following example shows how to send a getid request to a remote host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp getid
 1.3.6.1.2.1.1.1.0 community
 public ipaddr
 172.17.16.69
```

The following log message is written to the SNMP event manager log:

```
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_oid=1.3.6.1.2.1.1.5.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_value=jubjub.cisco.com
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_oid=1.3.6.1.2.1.1.6.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_value=
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysdescr_oid=1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_oid=1.3.6.1.2.1.1.2.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_value=products.222
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=1.3.6.1.2.1.1.3.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=10131676
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_oid=1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_value=YYY
```

Set Operations Examples

The following example shows how to perform a set operation on the local host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 set-type
 integer
 5 sysName.0 community
 public
```

The following log message is written to the SNMP event manager log:

```
1d04h:%HA_EM-6-LOG: lset: 1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lset: XXX
```

The following example shows how to perform a set operation on a remote host.

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 set-type integer
 5 sysName.0 community
 public ipaddr
172.17.16.69
```

The following log message is written to the SNMP event manager log:

```
1d04h:%HA_EM-6-LOG: lset: 1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lset: XXX
```

Generating SNMP Notifications Examples

The following example shows how to configure SNMP traps for the sysUpTime.0 variable:

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op
lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp var
 sysUpTime.0 oid
1.3.6.1.4.1.9.9.43.1.1.6.1.3.41 integer
 2
Device(config-applet)# action 1.4 info type snmp trap
 enterprise-oid
 ciscoSyslogMIB.2 generic-trapnum
6 specific-trapnum
1 trap-oid
```

```
 1.3.6.1.4.1.9.9.41.2.0.1 trap-var
sysUpTime.0
```

The following output is generated if the debug snmp packets command is enabled:

```
Device# debug snmp packets
1d04h: SNMP: Queuing packet to 172.69.16.2
1d04h: SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 172.19.rap 1
clogHistoryEntry.3 = 4
clogHistoryEntry.6 = 9999
1d04h: SNMP: Queuing packet to 172.19.208.130
1d04h: SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 172.19.rap 1
clogHistoryEntry.3 = 4
clogHistoryEntry.6 = 9999
1d04h: SNMP: Packet sent via UDP to 172.69.16.2
1d04h: SNMP: Packet sent via UDP to 172.69.16.2
infra-view10:
Packet Dump:
30 53 02 01 00 04 04 63 6f 6d 6d a4 48 06 09 2b
06 01 04 01 09 09 29 02 40 04 ac 13 d1 17 02 01
06 02 01 01 43 04 00 9b 82 5d 30 29 30 12 06 0d
2b 06 01 04 01 09 09 29 01 02 03 01 03 02 01 04
30 13 06 0d 2b 06 01 04 01 09 09 29 01 02 03 01
06 02 02 27 0f
Received SNMPv1 Trap:
Community: comm
Enterprise: ciscoSyslogMIBNotificationPrefix
Agent-addr: 172.19.209.23
Enterprise Specific trap.
Enterprise Specific trap: 1
Time Ticks: 10191453
clogHistSeverity = error(4)
clogHistTimestamp = 9999
```

The following example shows how to configure SNMP inform requests for the sysUpTime.0 variable:

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp var
 sysUpTime.0 oid
1.3.6.1.4.1.9.9.43.1.1.6.1.3.41 integer
 2
Device(config-applet)# action 1.4 info type snmp inform
trap-oid
 1.3.6.1.4.1.9.9.43.2.0.1 trap-var
 sysUpTime.0 community
 public ipaddr
 172.19.209.24
```

The following output is generated if the debug snmp packets command is enabled:

```
Device# debug snmp packets
1d04h: SNMP: Inform request, reqid 24, errstat 0, erridx 0
sysUpTime.0 = 10244391
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.40 = 1
1d04h: SNMP: Packet sent via UDP to 172.19.209.24.162
1d04h: SNMP: Packet received via UDP from 172.19.209.24 on FastEthernet0/0
```

```
1d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
1d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
1d04h: SNMP: Inform request, reqid 25, errstat 0, erridx 0
sysUpTime.0 = 10244396
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.41 = 2
1d04h: SNMP: Packet sent via UDP to 172.19.209.24.162
1d04h: SNMP: Packet received via UDP from 172.19.209.24 on FastEthernet0/0
1d04h: SNMP: Response, reqid 25, errstat 0, erridx 0
1d04h: SNMP: Response, reqid 25, errstat 0, erridx 0
Device# debug snmp packets
5d04h: SNMP: Packet received via UDP from 172.19.209.23 on FastEthernet0/0
5d04h: SNMP: Inform request, reqid 24, errstat 0, erridx 0
sysUpTime.0 = 10244391
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.40 = 1
5d04h: dest if_index = 1
5d04h: dest ip addr= 172.19.209.24
5d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
5d04h: SNMP: Packet sent via UDP to 172.19.209.23.57748
5d04h: SNMP: Packet received via UDP from 172.19.209.23 on FastEthernet0/0
5d04h: SNMP: Inform request, reqid 25, errstat 0, erridx 0
```

Configuring Variable Logic for EEM Applets Examples

The following sections provide examples on some selected action commands. For information on all the action commands supporting variable logic within applets, see the table below.

In this example, conditional loops **while**, **if** and **foreach** are used to print data. Other action commands such as **action divide**, **action increment** and **action puts** are used to define the actions to be performed when the conditions are met.

```
event manager applet printdata
event none
action 100 set colors "red green blue"
action 101 set shapes "square triangle rectange"
action 102 set i "1"
action 103 while $i lt 6
action 104  divide $i 2
action 105  if $_remainder eq 1
action 106   foreach _iterator "$colors"
action 107    puts nonewline "$_iterator "
action 108   end
action 109   puts ""
action 110  else
action 111   foreach _iterator "$shapes"
action 112    puts nonewline "$_iterator "
action 113   end
action 114   puts ""
action 115  end
action 116  increment i
action 117 end
```

When the event manager applet ex is run, the following output is obtained:

```
event manager run printdata
red green blue
square triangle rectange
red green blue
square triangle rectange
red green blue
```

In this example, two environment variables poll_interface and max_rx_rate are set to F0/0 and 3 respectively. Every 30 seconds there is a poll on an interface for rx rate. If the rx rate is greater than the threshold, a syslog message is displayed.

This applet makes use of the foreach conditional statement to poll the interface, the if conditional block to compare the value under RXPS with max_rx_rate that was set in the EEM environment variable.

```
event manager environment poll_interfaces F0/0
event manager environment max_rx_rate 3
ev man app check_rx_rate
ev timer watchdog name rx_timer time 30
action 100 foreach int $poll_interfaces
action 101  cli command "en"
action 102  cli command "show int $int summ | beg ------"
action 103  foreach line $_cli_result "\n"
action 105  regexp ".*[0-9]+\s+[0-9]+\s+[0-9]+\s+[0-9]+\s+[0-9]+\s+([0-9]+)\s+.*" $line
junk rxps
action 106   if $_regexp_result eq 1
action 107    if $rxps gt $max_rx_rate
action 108     syslog msg "Warning rx rate for $int is > than threshold. Current value is
$rxps
 (threshold is $max_rx_rate)"
action 109    end
action 110   end
action 111  end
action 112 end
```

Example syslog message:

```
Oct 16 09:29:26.153: %HA_EM-6-LOG: c: Warning rx rate for F0/0 is > than threshold.
 Current value is 4 (threshold is 3)
The output of show int F0/0 summ is of the format:

#show int f0/0 summ

 *: interface is up
 IHQ: pkts in input hold queue     IQD: pkts dropped from input queue
 OHQ: pkts in output hold queue    OQD: pkts dropped from output queue
 RXBS: rx rate (bits/sec)          RXPS: rx rate (pkts/sec)
 TXBS: tx rate (bits/sec)          TXPS: tx rate (pkts/sec)
 TRTL: throttle count

  Interface               IHQ   IQD  OHQ   OQD  RXBS RXPS  TXBS TXPS TRTL
------------------------------------------------------------------------
* FastEthernet0/0          0 87283    0     0     0    0     0    0    0
```

Note To use other action commands supporting variable logic within applets, use the commands listed in the table below.

Table 57: Available action commands

Action Commands	Purpose
action add	Adds the value of two variables when an EEM applet
action append	Appends the given value to the current value of a vari

Action Commands	Purpose
action break	Causes an immediate exit from a loop of actions w
action comment	Adds comments to an applet when an EEM applet
action context retrieve	Retrieves variables identified by a given set of cor
action context save	Saves information across multiple policy triggers
action continue	Continues with a loop of actions when an EEM ap
action decrement	Decrements the value of a variable when an EEM
action divide	Divides the dividend value by the given divisor va
action else	Specifies the beginning of else conditional action l triggered.
action elseif	Identifies the beginning of the else conditional act is triggered.
action end	Specifies the identification of the end of an condit when an EEM applet is triggered.
action exit	Specifies an immediate exit from the running appl
action foreach	Specifies the iteration of an input string using the
action gets	Gets an input from the local TTY in a synchronou is triggered.
action if	Specifies the identification of the beginning of an
action if goto	Instructs the applet to jump to a given label if the s
action increment	Increments the value of a variable when an EEM a
action info type interface-names	Specifies the action of obtaining interface names v
action info type snmp getid	Retrieves the individual variables from a Simple N operation.
action info type snmp inform	Sends an SNMP inform requests when an EEM ap
action info type snmp oid	Specifies the type of SNMP get operation and the c is triggered.
action info type snmp trap	Sends SNMP trap requests when an EEM applet is
action info type snmp var	Creates a variable for an SNMP object identifier (
action multiply	Specifies the action of multiplying the variable valu
action puts	Enables the action of printing data directly to the l

Action Commands	Purpose
action regexp	Specifies the action of matching a regular expression
action set (EEM)	Specifies the action of setting the value of a variable v
action string compare	Specifies the action of comparing two unequal strings
action string equal	Specifies the action of verifying whether or not two st
action string first	Specifies the action of returning the index on the first oc
action string index	Specifies the action of returning the characters specifi
action string last	Specifies the action of returning the index on the last oc
action string length	Specifies the action of returning the number of charac
action string match	Specifies the action of returning 1 to the $_string_resul
action string range	Specifies the action of storing a range of characters in
action string replace	Specifies the action of storing a new string by replacin is triggered.
action string tolower	Specifies the action of storing specific range of charac
action string toupper	Specifies the action of storing specific range of charac
action string trim	Specifies the action to trim a string when an EEM app
action string trimleft	Specifies the action to trim the characters of one string
action string trimright	Specifies the action to trim the characters one string fr
action subtract	Subtracts the value of a variable from another value w
action while	Specifies the action of identifying the beginning of a l

Configuring Event SNMP-Object Examples

The following example shows the SET operation and the value to set is in $_snmp_value and it is managed by the script. The example below saves the oid and its value as contexts to be retrieved later.

```
event manager applet snmp-object1
 description "APPLET SNMP-OBJ-1"
 event snmp-object oid 1.3.6.1.2.1.31.1.1.1.18 type string sync no skip no istable yes
default 0
 action 1 syslog msg "SNMP-OBJ1:TRIGGERED" facility "SNMP_OBJ"
 action 2 context save key myoid variable "_snmp_oid"
 action 3 context save key myvalue variable "_snmp_value"
```

Configuring Description of an EEM Applet Examples

The following example shows how to add or modify the description for an Embedded Event Manager (EEM) applet that is run by sampling Simple Network Management Protocol (SNMP):

```
event manager applet test
 description "This applet looks for the word count in syslog messages"
 event syslog pattern "count"
 action 1 syslog msg hi
```

Additional References

The following sections provide references related to writing EEM policies Using the Cisco IOS CLI.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
EEM commands: complete command syntax, defaults, command mode, command history, usage guidelines, and examples	Cisco IOS Embedded Event Manager Command Reference
Embedded Event Manager overview	Embedded Event Manager Overview module
Embedded Event Manager policy writing using Tcl	Writing Embedded Event Manager Policies Using Tcl module
Configuring enhanced object tracking	Configuring Enhanced Object Tracking module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
CISCO-EMBEDDED-EVENT-MGR-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Writing EEM 4.0 Policies Using the Cisco IOS CLI

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

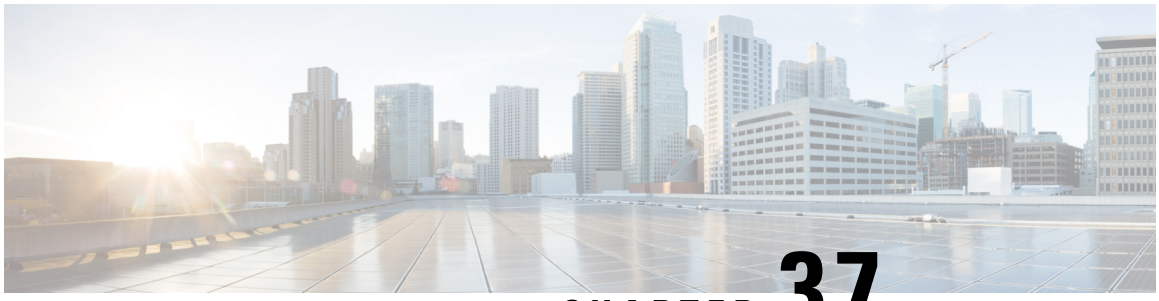
Table 58: Feature Information for Writing EEM 4.0 Policies Using the Cisco IOS CLI

Feature Name	Releases	Feature Information
Embedded Event Manager 1.0	12.0(26)S 12.3(4)T	EEM 1.0 introduced Embedded Event Manager applet creation with the SNMP and syslog event detectors. EEM 1.0 also introduced the following actions: generating prioritized syslog messages, generating a CNS event for upstream processing by Cisco CNS devices, reloading the Cisco software, and switching to a secondary processor in a fully redundant hardware configuration. The following commands were introduced by this feature: **action cns-event**, **action force-switchover**, **action reload**, **action syslog**, **debug event manager**, **event manager applet**, **event snmp**, **event syslog**, **show event manager policy registered**.

Feature Name	Releases	Feature Information
Embedded Event Manager 2.0	12.2(25)S	EEM 2.0 introduced the application-specific event detector, the counter event detector, the interface counter event detector, the timer event detector, and the watchdog event detector. New actions included modifying a named counter, publishing an application-specific event, and generating an SNMP trap. The ability to define environment variables and to run EEM policies written using Tcl was introduced, and two sample policies were included with the software. The following commands were introduced by this feature: **action counter**, **action publish-event**, **action snmp-trap**, **event application**, **event counter**, **event interface**, **event ioswdsysmon**, **event manager environment**, **event manager history size**, **event manager policy**, **event manager scheduler suspend**, **event timer**, **show event manager environment**, **show event manager history events**, **show event manager history traps**, **show event manager policy available**, **show event manager policy pending**.
Embedded Event Manager 2.1	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	EEM 2.1 introduced some new event detectors and actions with new functionality to allow EEM policies to be run manually and the ability to run multiple concurrent policies. Support for Simple Network Management Protocol (SNMP) event detector rate-based events was provided as was the ability to create policies using Tool Command Language (Tcl). The following commands were introduced or modified by this feature: **action cli**, **action counter**, **action info**, **action mail**, **action policy**, **debug event manager**, **event cli**, **event manager directory user**, **event manager policy**, **event manager run**, **event manager scheduler script**, **event manager session cli username**, **event none**, **event oir**, **event snmp**, **event syslog**, **set**(EEM), **show event manager directory user**, **show event manager policy registered**, **show event manager session cli username**.
Embedded Event Manager 2.1 (Software Modularity)	12.2(18)SXF4 Cisco IOS Software Modularity images	EEM 2.1 for Software Modularity images introduced the GOLD, system manager, and WDSysMon (Cisco IOS Software Modularity watchdog) event detectors, and the ability to display Cisco IOS Software Modularity processes and process metrics. The following commands were introduced by this feature: **event gold**, **event process**, **show event manager metric process**. **Note** EEM 2.1 for Software Modularity images also supports the resource and RF event detectors introduced in EEM 2.2, but it does not support the enhanced object tracking event detector or the actions to read and set tracked objects.

Feature Name	Releases	Feature Information
Embedded Event Manager 2.2	12.4(2)T 12.2(31)SB3 12.2(33)SRB	EEM 2.2 introduced the enhanced object tracking, resource, and RF event detectors. The actions of reading and setting the state of a tracked object were also introduced. The following commands were introduced or modified by this feature: **action track read**, **action track set**, **default-state**, **event resource**, **event rf**, **event track**, **show track**, **track stub-object**.
SNMP event detector delta environment variable	12.4(11)T	A new SNMP event detector environment variable, _snmp_oid_delta_val, was introduced. This is a minor enhancement. Minor enhancements are not typically listed in Feature Navigator.
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB 15.1(2)SY	EEM 2.3 introduced some new features relative to the Generic Online Diagnostics (GOLD) Event Detector on the Cisco Catalyst 6500 Series switches. The **event gold** command was enhanced in addition to the Tcl keywords--**action-notify**, **testing-type**, **test-name**, **test-id**, **consecutive-failure**, **platform-action**, and **maxrun**--for improved reaction to GOLD test failures and conditions Read-only variables were added under the **GOLD Event Detector** category to provide access to platform-wide and test-specific GOLD event detector information for a detected event.
Embedded Event Manager 2.4	12.4(20)T 12.2(33)SXI 12.2(33)SRE 15.1(2)SY	EEM 2.4 is supported in Cisco IOS Release 12.4(20)T and later releases, and introduced several new features. The following commands were introduced by this feature: **attribute (EEM)** , **correlate**, **event manager detector rpc**, **event manager directory user repository**, **event manager update user policy**, **event manager scheduler clear**, **event manager update user policy**, **event owner**, **event rpc**, **event snmp-notification**, **show event manager detector**, **show event manager version**, **trigger (EEM)**.

Feature Name	Releases	Feature Information
Embedded Event Manger 3.0	12.4(22)T 12.2(33)SRE 12.2(50)SY	EEM 3.0 is supported in Cisco IOS Release 12.4(22)T and later releases, and introduced several new features. The following commands were introduced or modified by this feature: **action add** , **action append**, **action break**, **action comment**, **action context retrieve**, **action context save**, **action continue**, **action decrement**, **action divide**, **action else**, **action elseif**, **action end**, **action exit**, **action foreach**, **action gets**, **action if**, **action if goto**, **action increment**, **action info type interface-names**, **action info type snmp getid**, **action info type snmp inform**, **action info type snmp oid**, **action info type snmp trap**, **action info type snmp var**, **action multiply**, **action puts**, **action regexp**, **action set (EEM)**, **action string compare**, **action string equal**, **action string first**, **action string index**, **action string last**, **action string length**, **action string match**, **action string range**, **action string replace**, **action string tolower**, **action string toupper**, **action string trim**, **action string trimleft**, **action string trimright**, **action subtract**, **action while**, **event cli**, **event ipsla**, **event manager detector routing**, **event manager scheduler**, **event manager scheduler clear**, **event manager scheduler hold**, **event manager scheduler modify**, **event manager scheduler release**, **event nf**, **event routing**, **show event manager policy active**, **show event manager policy pending**, and **show event manager scheduler**.
Embedded Event Manager 3.1	15.0(1)M 15.1(1)SY 15.1(2)SY	EEM 3.1 is supported in Cisco IOS Release 15.0(1)M and later releases, and introduced several new features. The following commands were introduced or modified by this feature: **action syslog**, **description (EEM)** , **event manager applet** , **event manager policy** , **event snmp-notification** , **event snmp-object** , **show event manager policy registered** , and **show event manager policy available** .
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG 15.1(3)T 15.1(1)SY 15.1(2)SY	EEM is a distributed and customized approach to event detection and recovery offered directly in a Cisco IOS device. The following sections provide information about this feature: The following commands were introduced or modified: **debug event manager**, **event identity**, **event mat**, **event neighbor-discovery**, **show event manager detector**.
Embedded Event Manager 4.0	15.2(2)T 15.1(1)SY 15.1(2)SY	EEM 4.0 is supported in 15.2(2)T and later releases, and introduced several new features. The following commands were introduced or modified: **action file**, **action mail**, **action syslog**, **clear event manager detector counters**, **clear event manager server counters**, **event cli**, **event manager policy**, **event manager scheduler**, **event syslog**, **show event manager detector**, **show event manager policy registered**, **show event manager statistics**.



CHAPTER 37

Writing Embedded Event Manager Policies Using Tcl

This module describes how software developers can write and customize Embedded Event Manager (EEM) policies using Tool command language (Tcl) scripts to handle Cisco software faults and events. EEM is a policy-driven process by means of which faults in the Cisco software system are reported through a defined application programing interface (API). The EEM policy engine receives notifications when faults and other events occur. EEM policies implement recovery on the basis of the current state of the system and the actions specified in the policy for a given event. Recovery actions are triggered when the policy is run.

Prerequisites for Writing Embedded Event Manager Policies Using Tcl

- Before writing EEM policies, you should be familiar with the “ Embedded Event Manager Overview ” module.
- If you want to write EEM policies using the command-line interface (CLI) commands, you should be familiar with the “ Writing Embedded Event Manager Policies Using the Cisco IOS CLI ” module.

Information About Writing Embedded Event Manager Policies Using Tcl

EEM Policies

EEM offers the ability to monitor events and take informational or corrective action when the monitored events occur or reach a threshold. An EEM policy is an entity that defines an event and the actions to be taken when that event occurs. There are two types of EEM policies: an applet or a script. An applet is a simple form of policy that is defined within the command-line interface (CLI) configuration. A script is a form of policy that is written in Tool Command Language (Tcl).

EEM Applet

An EEM applet is a concise method for defining event screening criteria and the actions to be taken when that event occurs. In EEM applet configuration mode, three types of configuration statements are supported. The event commands are used to specify the event criteria to trigger the applet to run, the action commands are used to specify an action to perform when the EEM applet is triggered, and the **set** command is used to set the value of an EEM applet variable. Currently only the _exit_status variable is supported for the **set** command.

Only one event configuration command is allowed within an applet configuration. When applet configuration submode is exited and no event command is present, a warning is displayed stating that no event is associated with the applet. If no event is specified, the applet is not considered registered. When no action is associated with the applet, events are still triggered but no actions are performed. Multiple action configuration commands are allowed within an applet configuration. Use the **show event manager policy registered** command to display a list of registered applets.

Before modifying an EEM applet, be aware that the existing applet is not replaced until you exit applet configuration mode. While you are in applet configuration mode modifying the applet, the existing applet may be executing. It is safe to modify the applet without unregistering it, because changes are written to a temporary file. When you exit applet configuration mode, the old applet is unregistered and the new version is registered.

Action configuration commands within an applet are uniquely identified using the *label* argument, which can be any string value. Actions are sorted within an applet in ascending alphanumeric key sequence using the *label* argument as the sort key, and they are run using this sequence. The same *label* argument can be used in different applets; the labels must be unique only within one applet.

The Embedded Event Manager schedules and runs policies on the basis of an event specification that is contained within the policy itself. When applet configuration mode is exited, EEM examines the event and action commands that are entered and registers the applet to be run when a specified event occurs.

For more details about writing EEM policies using the Cisco IOS CLI, see the “Writing Embedded Event Manager Policies Using the Cisco IOS CLI” module.

EEM Script

All Embedded Event Manager scripts are written in Tcl. Tcl is a string-based command language that is interpreted at run time. The version of Tcl supported is Tcl version 8.3.4 plus added script support. Scripts are defined using an ASCII editor on another device, not on the networking device. The script is then copied

to the networking device and registered with EEM. Tcl scripts are supported by EEM. As an enforced rule, Embedded Event Manager policies are short-lived run time routines that must be interpreted and executed in less than 20 seconds of elapsed time. If more than 20 seconds of elapsed time are required, the maxrun parameter may be specified in the event_register statement to specify any desired value.

EEM policies use the full range of the Tcl language's capabilities. However, Cisco provides enhancements to the Tcl language in the form of Tcl command extensions that facilitate the writing of EEM policies. The main categories of Tcl command extensions identify the detected event, the subsequent action, utility information, counter values, and system information.

EEM allows you to write and implement your own policies using Tcl. Writing an EEM script involves:

- Selecting the event Tcl command extension that establishes the criteria used to determine when the policy is run.
- Defining the event detector options associated with detecting the event.
- Choosing the actions to implement recovery or respond to the detected event.

EEM Policy Tcl Command Extension Categories

There are different categories of EEM policy Tcl command extensions.

Note The Tcl command extensions available in each of these categories for use in all EEM policies are described in later sections in this document.

Table 59: EEM Policy Tcl Command Extension Categories

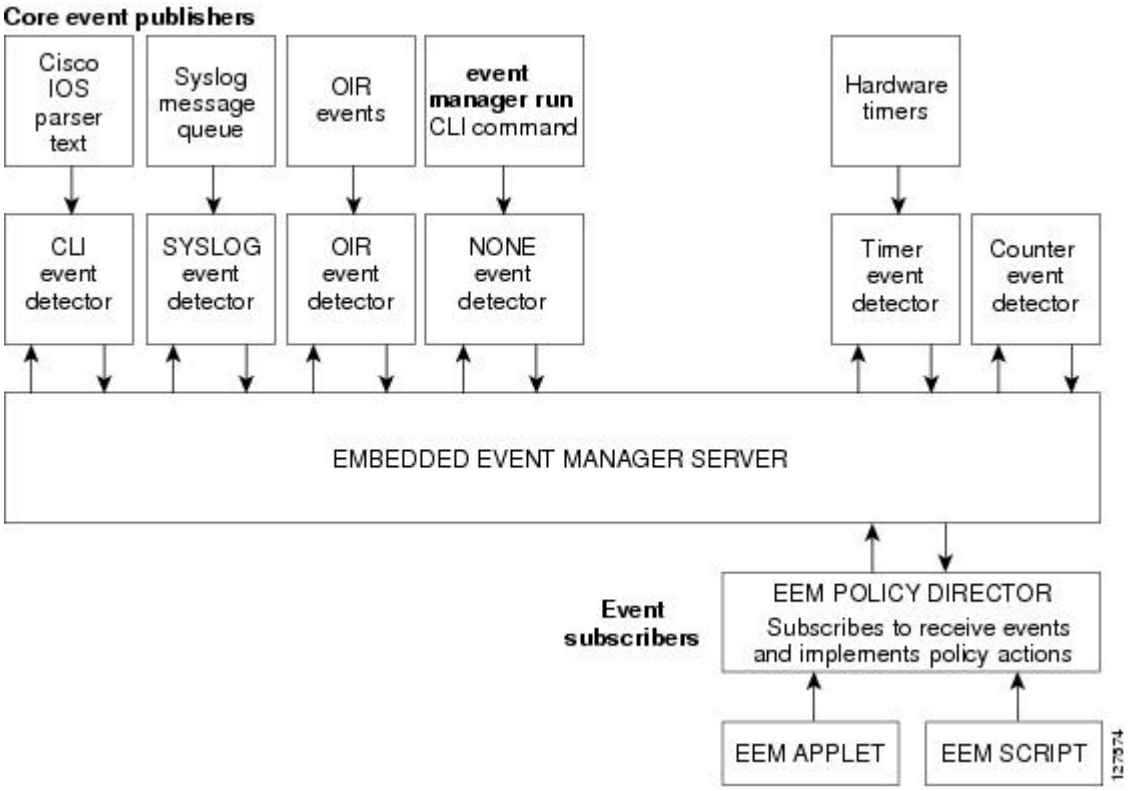
Category	Definition
EEM event Tcl command extensions (three types: event information, event registration, and event publish)	This category is represented by the **event_register_** *xxx* family of event-specific commands. There is a separate event information Tcl command extension in this category as well: **event_reqinfo**. This is the command used in policies to query the EEM for information about an event. There is also an EEM event publish Tcl command extension **event_publish**> *that publishes an application-specific event.*
EEM action Tcl command extensions	These Tcl command extensions (for example, **action_syslog**) are used by policies to respond to or recover from an event or fault. In addition to these extensions, developers can use the Tcl language to implement any action desired.
EEM utility Tcl command extensions	These Tcl command extensions are used to retrieve, save, set, or modify application information, counters, or timers.
EEM system information Tcl command extensions	This category is represented by the **sys_reqinfo _** *xxx* family of system-specific information commands. These commands are used by a policy to gather system information.
EEM context Tcl command extensions	These Tcl command extensions are used to store and retrieve a Tcl context (the visible variables and their values).

General Flow of EEM Event Detection and Recovery

EEM is a flexible, policy-driven framework that supports in-box monitoring of different components of the system with the help of software agents known as event detectors. The figure below shows the relationship between the EEM server, the core event publishers (event detectors), and the event subscribers (policies). Basically, event publishers screen events and publish them when there is a match on an event specification that is provided by the event subscriber. Event detectors notify the EEM server when an event of interest occurs.

When an event or fault is detected, Embedded Event Manager determines from the event publishers--an example would be the OIR events publisher in the figure below--if a registration for the encountered fault or event has occurred. EEM matches the event registration information with the event data itself. A policy registers for the detected event with the Tcl command extension **event_register _** *xxx*. The event information Tcl command extension **event_reqinfo** is used in the policy to query the Embedded Event Manager for information about the detected event.

Figure 16: Embedded Event Manager Core Event Detectors



Safe-Tcl

Safe-Tcl is a safety mechanism that allows untrusted Tcl scripts to run in an interpreter that was created in the safe mode. The safe interpreter has a restricted set of commands that prevent accessing some system resources and harming the host and other applications. For example, it does not allow commands to access critical Cisco IOS file system directories.

Cisco-defined scripts run in full Tcl mode, but user-defined scripts run in Safe-Tcl mode. Safe-Tcl allows Cisco to disable or customize individual Tcl commands. For more details about Tcl commands, go to http://www.tcl.tk/man/ .

The following list of Tcl commands are restricted with a few exceptions. Restrictions are noted against each command or command keyword:

- **cd** --Change directory is not allowed to one of the restricted Cisco directory names.
- **encoding** --The commands **encoding names**, **encoding convertfrom**, and **encoding convertto** are permitted. The **encoding system** command with no arguments is permitted, but the **encoding system** command with the **?encoding?** keyword is not permitted.
- **exec** --Not permitted.
- **fconfigure** --Permitted.
- **file** --The following are permitted:
  - **file dirname**
  - **file exists**
  - **file extension**
  - **file isdirectory**
  - **file join**
  - **file pathtype**
  - **file rootname**
  - **file split**
  - **file stat**
  - **file tail**
- **file** --The following are not permitted:
  - **file atime**
  - **file attributes**
  - **file channels**
  - **file copy**
  - **file delete**
  - **file executable**
  - **file isfile**
  - **file link**
  - **file lstat**
  - **file mkdir**
  - **file mtime**
  - **file nativename**
  - **file normalize**
  - **file owned**
  - **file readable**
  - **file readlink**
  - **file rename**
  - **file rootname**
  - **file separator**
  - **file size**

  - **file system**
  - **file type**
  - **file volumes**
  - **file writable**

- **glob** --The **glob** command is not permitted when searching in one of the restricted Cisco directories. Otherwise, it is permitted.
- **load** --Only files that are in the user policy directory or the user library directory are permitted to be loaded. Static packages (for example, libraries that consist of C code) are not permitted to be loaded with the **load** command.
- **open** --The **open** command is not allowed for a file that is located in one of the restricted Cisco directories.
- **pwd** --The **pwd** command is not permitted.
- **socket** --The **socket** command is permitted.
- **source** --The **source** command is permitted for files that are in the user policy directory or the user library directory.

Bytecode Support for EEM 2.4

EEM 2.4 introduces bytecode language (BCL) support by accepting files with the standard bytecode script extension .tbc. Tcl version 8.3.4 defines a BCL and includes a compiler that translates Tcl scripts into BCL. Valid EEM policy file extensions in EEM 2.4 for user and system policies are .tcl (Tcl Text files) and .tbc (Tcl bytecode files).

Storing Tcl scripts in bytecode improves the execution speed of the policy because the code is precompiled, creates a smaller policy size, and obscures the policy code. Obfuscation makes it a little more difficult to modify scripts and hides logic to preserve intellectual property rights.

Support for bytecode is being added to provide another option for release of supported and trusted code. We recommend that you only run well understood, or trusted and supported software on network devices. To generate Tcl bytecode for IOS EEM support, use TclPro versions 1.4 or 1.5.

To translate a Tcl script to bytecode you can use procomp, part of Free TclPro Compiler, or Active State Tcl Development Kit. When a Tcl script is compiled using procomp, the code is scrambled and a .tbc file is generated. The bytecode files are platform-independent and can be generated on any operating system on which TclPro is available, including Windows, Linux, and UNIX. Procomp is part of TclPro and available from http://www.tcl.tk/software/tclpro.

Registration Substitution

In addition to regular Tcl substitution, EEM 2.3 permits the substitution of an individual parameter in an EEM event registration statement line with an environment variable.

EEM 2.4 introduces the ability to replace multiple parameters in event registration statement lines with a single environment variable.

Note Only the first environment variable supports multiple parameter substitution. Individual parameters can still be specified with additional environment variables after the initial variable.

To illustrate the substitution, a single environment variable, $_eem_syslog_statement is configured as:

```
::cisco::eem::event_register_syslog pattern COUNT
```

Using the registration substitution, the $_eem_syslog_statement environment variable is used in the following EEM user policy:

```
$_eem_syslog_statement occurs $_eem_occurs_val
action_syslog "this is test 3"
```

Environment variables must be defined before a policy using them is registered. To define the $_eem_syslog_statement environment variable:

```
Device(config)# event manager environment eem_syslog_statement
::cisco::eem::event_register_syslog pattern COUNT
Device(config)# event manager environment eem_occurs_val 2
```

Cisco File Naming Convention for EEM

All Embedded Event Manager policy names, policy support files (for example, e-mail template files), and library filenames are consistent with the Cisco file naming convention. In this regard, Embedded Event Manager policy filenames adhere to the following specification:

- An optional prefix--Mandatory.--indicating, if present, that this is a system policy that should be registered automatically at boot time if it is not already registered. For example: Mandatory.sl_text.tcl.
- A filename body part containing a two-character abbreviation (see the table below) for the first event specified; an underscore part; and a descriptive field part that further identifies the policy.
- A filename suffix part defined as .tcl.

Embedded Event Manager e-mail template files consist of a filename prefix of email_template, followed by an abbreviation that identifies the usage of the e-mail template.

Embedded Event Manager library filenames consist of a filename body part containing the descriptive field that identifies the usage of the library, followed by _lib, and a filename suffix part defined as .tcl.

Table 60: Two-Character Abbreviation Specification

ap	event_register_appl
cl	event_register_cli
ct	event_register_counter
go	event_register_gold
if	event_register_interface
io	event_register_ioswdsysmon
la	event_register_ipsla
nf	event_register_nf
no	event_register_none

oi	event_register_oir
pr	event_register_process
rf	event_register_rf
rs	event_register_resource
rt	event_register_routing
rp	event_register_rpc
sl	event_register_syslog
sn	event_register_snmp
st	event_register_snmp_notification
so	event_register_snmp_object
tm	event_register_timer
tr	event_register_track
ts	event_register_timer_subscriber
wd	event_register_wdsysmon

How to Write Embedded Event Manager Policies Using Tcl

Registering and Defining an EEM Tcl Script

Perform this task to configure environment variables and register an EEM policy. EEM schedules and runs policies on the basis of an event specification that is contained within the policy itself. When an EEM policy is registered, the software examines the policy and registers it to be run when the specified event occurs.

Before you begin

You must have a policy available that is written in the Tcl scripting language. Sample policies are provided--see the details in the *Sample EEM Policies* task to see which policies are available for the Cisco IOS release image that you are using--and these sample policies are stored in the system policy directory.

SUMMARY STEPS

1. **enable**
2. **show event manager environment** [**all**| *variable-name*]
3. **configure terminal**
4. **event manager environment** *variable-name string*
5. Repeat Step 4 to configure all the environment variables required by the policy to be registered in Step 6.

6. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager environment** [**all**| *variable-name*] **Example:** `Device# show event manager environment all`	(Optional) Displays the name and value of EEM environment variables. • The optional **all** keyword displays all the EEM environment variables. • The optional *variable-name*argument displays information about the specified environment variable.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**event manager environment** *variable-name string* **Example:** `Device(config)# event manager environment _cron_entry 0-59/2 0-23/1 * * 0-6`	Configures the value of the specified EEM environment variable. • In this example, the software assigns a CRON timer environment variable to be set to the second minute of every hour of every day.
Step 5	Repeat Step 4 to configure all the environment variables required by the policy to be registered in Step 6.	--
Step 6	**event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**] **Example:** `Device(config)# event manager policy tm_cli_cmd.tcl type system`	Registers the EEM policy to be run when the specified event defined within the policy occurs. • Use the **system** keyword to register a Cisco-defined system policy. • Use the **user** keyword to register a user-defined system policy. • Use the **trap** keyword to generate an SNMP trap when the policy is triggered. • In this example, the sample EEM policy named tm_cli_cmd.tcl is registered as a system policy.
Step 7	**exit** **Example:**	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# exit`	

Examples

In the following example, the **show event manager environment** privileged EXEC command is used to display the name and value of all EEM environment variables.

```
Device# show event manager environment all
No.  Name                          Value
1    _cron_entry                   0-59/2 0-23/1 * * 0-6
2    _show_cmd                     show ver
3    _syslog_pattern               .*UPDOWN.*Ethernet1/0.*
4    _config_cmd1                  interface Ethernet1/0
5    _config_cmd2                  no shut
```

Displaying EEM Registered Policies

Perform this optional task to display EEM registered policies.

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]

Use this command with the **time-ordered** keyword to display information about currently registered policies sorted by time, for example:

Example:

```
Device# show event manager policy registered time-ordered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Wed May11  01:43:18 2005  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240
2    system  syslog              Off   Wed May11  01:43:28 2005  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90
3    system  proc abort          Off   Wed May11  01:43:38 2005  pr_cdp_abort.tcl
```

```
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20
```

Use this command with the **name-ordered** keyword to display information about currently registered policies sorted by name, for example:

Example:

```
Device# show event manager policy registered name-ordered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  proc abort          Off   Wed May11  01:43:38 2005  pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20
2    system  syslog              Off   Wed May11  01:43:28 2005  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90
3    system  timer cron          Off   Wed May11  01:43:18 2005  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240
```

Use this command with the **event-type** keyword to display information about currently registered policies for the event type specified in the *event-name* argument, for example:

Example:

```
Device# show event manager policy registered event-type syslog
No.  Type    Event Type          Time Registered           Name
1    system  syslog              Wed May11  01:43:28 2005  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90
```

Unregistering EEM Policies

Perform this task to remove an EEM policy from the running configuration file. Execution of the policy is canceled.

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]
3. **configure terminal**
4. **no event manager policy** *policy-filename*
5. **exit**
6. Repeat Step 2 to ensure that the policy has been removed.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*] **Example:** `Device# show event manager policy registered`	(Optional) Displays the EEM policies that are currently registered. • The optional **system**or **user** keyword displays the registered system or user policies. • If no keywords are specified, EEM registered policies for all event types are displayed in time order.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**no event manager policy** *policy-filename* **Example:** `Device(config)# no event manager policy pr_cdp_abort.tcl`	Removes the EEM policy from the configuration, causing the policy to be unregistered. • In this example, the **no** form of the command is used to unregister a specified policy.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	Repeat Step 2 to ensure that the policy has been removed. **Example:** `Device# show event manager policy registered`	--

Examples

In the following example, the **show event manager policy registered** privileged EXEC command is used to display the three EEM policies that are currently registered:

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Tue Oct11  01:43:18 2005 tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Tue Oct11  01:43:28 2005 sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
3    system  proc abort          Off   Tue Oct11  01:43:38 2005 pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20.000
```

After the current policies are displayed, it is decided to delete the pr_cdp_abort.tcl policy using the **no** form of the **event manager policy** command:

```
Device# configure terminal
Device(config)# no event manager policy pr_cdp_abort.tcl
Device(config)# exit
```

The **show event manager policy registered** privileged EXEC command is entered again to display the EEM policies that are currently registered. The policy pr_cdp_abort.tcl is no longer registered.

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Tue Oct11  01:45:17 2005 tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Tue Oct11  01:45:27 2005 sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
```

Suspending EEM Policy Execution

Perform this task to immediately suspend the execution of all EEM policies. Suspending policies, instead of unregistering them, might be necessary for reasons of temporary performance or security.

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]
3. **configure terminal**
4. **event manager scheduler suspend**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*] **Example:** `Device# show event manager policy registered`	(Optional) Displays the EEM policies that are currently registered. • The optional **system**or **user** keyword displays the registered system or user policies. • If no keywords are specified, EEM registered policies for all event types are displayed in time order.

	Command or Action	Purpose
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**event manager scheduler suspend** **Example:** `Device(config)# event manager scheduler suspend`	Immediately suspends the execution of all EEM policies.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the **show event manager policy registered** privileged EXEC command is used to display all the EEM registered policies:

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Sat Oct11  01:43:18 2003  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Sat Oct11  01:43:28 2003  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
3    system  proc abort          Off   Sat Oct11  01:43:38 2003  pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20.000
```

The **event manager scheduler suspend** command is entered to immediately suspend the execution of all EEM policies:

```
Device# configure terminal
Device(config)# event manager scheduler suspend
*Nov  2 15:34:39.000: %HA_EM-6-FMS_POLICY_EXEC: fh_io_msg: Policy execution has been
suspended
```

Managing EEM Policies

Perform this task to specify a directory to use for storing user library files or user-defined EEM policies.

Note This task applies only to EEM policies that are written using Tcl scripts.

SUMMARY STEPS

1. **enable**
2. **show event manager directory user** [**library**| **policy**]
3. **configure terminal**
4. **event manager directory user** {**library** *path*| **policy** *path*}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show event manager directory user** [**library**| **policy**] **Example:** `Device# show event manager directory user library`	(Optional) Displays the directory to use for storing EEM user library or policy files. • The optional **library**keyword displays the directory to use for user library files. • The optional **policy**keyword displays the directory to use for user-defined EEM policies.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**event manager directory user** {**library** *path*| **policy** *path*} **Example:** `Device(config)# event manager directory user library disk0:/user_library` `Device(config)# event manager directory user library bootflash:/user_library`	Specifies a directory to use for storing user library files or user-defined EEM policies. • Use the *path*argument to specify the absolute pathname to the user directory.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the **show event manager directory user** privileged EXEC command is used to display the directory, if it exists, to use for storing EEM user library files:

```
Device# show event manager directory user library
disk0:/user_library

Device# show event manager directory user library
bootflash:/user_library
```

Modifying History Table Size and Displaying EEM History Data

Perform this optional task to change the size of the history tables and to display EEM history data.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager history size** {**events** | **traps**} [*size*]
4. **exit**
5. **show event manager history events** [**detailed**] [**maximum** *number*]
6. **show event manager history traps** [**server** | **policy]**

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 configure terminal

Enters global configuration mode.

Example:

```
Device# configure terminal
```

Step 3 event manager history size {events | traps} [size]

Use this command to change the size of the EEM event history table or the size of the EEM SNMP trap history table. In the following example, the size of the EEM event history table is changed to 30 entries:

Example:

```
Device(config)# event manager history size events 30
```

Step 4 exit

Exits global configuration mode and returns to privileged EXEC mode.

Example:

```
Device(config)# exit
```

Step 5 **show event manager history events** [**detailed**] [**maximum** *number*]

Use this command to display information about each EEM event that has been triggered.

Example:

```
Device# show event manager history events
No.  Time of Event             Event Type          Name
1    Fri Sep  9 13:48:40 2005  syslog              applet: one
2    Fri Sep  9 13:48:40 2005  syslog              applet: two
3    Fri Sep  9 13:48:40 2005  syslog              applet: three
4    Fri Sep  9 13:50:00 2005  timer cron          script: tm_cli_cmd.tcl
5    Fri Sep  9 13:51:00 2005  timer cron          script: tm_cli_cmd.tcl
```

Step 6 **show event manager history traps** [**server** | **policy]**

Use this command to display the EEM SNMP traps that have been sent either from the EEM server or from an EEM policy.

Example:

```
Device# show event manager history traps
No.  Time                      Trap Type           Name
1    Fri Sep  9 13:48:40 2005  server              applet: four
2    Fri Sep  9 13:57:03 2005  policy              script: no_snmp_test.tcl
```

Displaying Software Modularity Process Reliability Metrics Using EEM

Perform this optional task to display reliability metrics for Cisco IOS Software Modularity processes. The **show event manager metric processes** command is supported only in Software Modularity images.

SUMMARY STEPS

1. **enable**
2. **show event manager metric process** {**all**| *process-name*}

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show event manager metric process** {**all**| *process-name*}

Use this command to display the reliability metric data for processes. The system keeps a record of when processes start and end, and this data is used as the basis for reliability analysis. In this partial example, the first and last entries showing the metric data for the processes on all the cards inserted in the system are displayed.

Example:

```
Device# show event manager metric process all
====================================
process name: devc-pty, instance: 1
sub_system id: 0, version: 00.00.0000
--------------------------------
last event type: process start
recent start time: Fri Oct10  20:34:40 2005
recent normal end time: n/a
recent abnormal end time: n/a
number of times started: 1
number of times ended normally: 0
number of times ended abnormally: 0
most recent 10 process start times:
--------------------------
Fri Oct10  20:34:40 2005
--------------------------
most recent 10 process end times and types:
cumulative process available time: 6 hours 30 minutes 7 seconds 378 milliseconds
cumulative process unavailable time: 0 hours 0 minutes 0 seconds 0 milliseconds
process availability:  0.100000000
number of abnormal ends within the past 60 minutes (since reload): 0
number of abnormal ends within the past 24 hours (since reload): 0
number of abnormal ends within the past 30 days (since reload): 0
.
.
.
====================================
process name: cdp2.iosproc, instance: 1
sub_system id: 0, version: 00.00.0000
--------------------------------
last event type: process start
recent start time: Fri Oct10  20:35:02 2005
recent normal end time: n/a
recent abnormal end time: n/a
number of times started: 1
number of times ended normally: 0
number of times ended abnormally: 0
most recent 10 process start times:
--------------------------
Fri Oct10  20:35:02 2005
--------------------------
most recent 10 process end times and types:

cumulative process available time: 6 hours 29 minutes 45 seconds 506 milliseconds
cumulative process unavailable time: 0 hours 0 minutes 0 seconds 0 milliseconds
process availability:  0.100000000
number of abnormal ends within the past 60 minutes (since reload): 0
number of abnormal ends within the past 24 hours (since reload): 0
number of abnormal ends within the past 30 days (since reload): 0
```

Troubleshooting Tips

Use the **debug event manager** command in privileged EXEC mode to troubleshoot EEM command operations. Use any debugging command with caution because the volume of output generated can slow or stop the device operations. We recommend that this command be used only under the supervision of a Cisco engineer.

Modifying the Sample EEM Policies

Perform this task to modify one of the sample policies. Cisco software contains some sample policies in the images that contain the Embedded Event Manager. Developers of EEM policies may modify these policies by customizing the event for which the policy is to be run and the options associated with logging and responding to the event. In addition, developers may select the actions to be implemented when the policy runs.

Sample EEM Policies

Cisco includes a set of sample policies shown in the table below. You can copy the sample policies to a user directory and then modify the policies, or you can write your own policies. Tcl is currently the only Cisco-supported scripting language for policy creation. Tcl policies can be modified using a text editor such as Emacs. Policies must execute within a defined number of seconds of elapsed time, and the time variable can be configured within a policy. The default is currently 20 seconds.

The table below describes the sample EEM policies.

Table 61: Sample EEM Policy Descriptions

Name of Policy	Description
pr_cdp_abort.tcl	Introduced with Cisco Software Modularity images. This policy monitors for cdp2.iosproc process terminates events. It will log a message to SYSLOG and send an e-mail with the details of the termination.
pr_crash_reporter.tcl	Introduced with Cisco Software Modularity images. This policy monitors for all process termination events. When an event occurs, the policy will send crash information, including the crashdump file, to the specified URL where a CGI script processes the data.
pr_iprouting_abort.tcl	Introduced with Cisco Software Modularity images. This policy monitors for iprouting.iosproc process termination events. It will log a message to SYSLOG and send an e-mail with the details of the termination.
sl_intf_down.tcl	This policy runs when a configurable syslog message is logged. It will execute a configurable CLI command and e-mail the results.
tm_cli_cmd.tcl	This policy runs using a configurable CRON entry. It will execute a configurable CLI command and e-mail the results.
tm_crash_history.tcl	Introduced with Cisco Software Modularity images. This policy runs at midnight every day and e-mails a process crash history report to a specified e-mail address.
tm_crash_reporter.tcl	This policy runs 5 seconds after it is registered. If the policy is saved in the configuration, it will also run each time that the device is reloaded. The policy will prompt for the reload reason. If the reload was due to a crash, the policy will search for the latest crashinfo file and send this information to a specified URL location.
tm_fsys_usage.tcl	Introduced with Cisco Software Modularity images. This policy runs using a configurable CRON entry and monitors disk space usage. A syslog message will be displayed if disk space usage crosses configurable thresholds.

Name of Policy	Description
wd_mem_reporter.tcl	Introduced with Cisco Software Modularity images. This policy reports on low system memory conditions when the amount of memory available falls below 20 percent of the initial available system memory. A syslog message will be displayed and, optionally, an e-mail will be sent.

SUMMARY STEPS

1. **enable**
2. **show event manager policy available detailed** *policy-filename*
3. Cut and paste the contents of the sample policy displayed on the screen to a text editor.
4. Edit the policy and save it with a new filename.
5. Copy the new file back to the device flash memory.
6. **configure terminal**
7. **event manager directory user** {**library** *path*| **policy** *path*}
8. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show event manager policy available detailed** *policy-filename*
detailed

Displays the actual specified sample policy including details about the environment variables used by the policy and instructions for running the policy. The keyword was introduced for the **show event manager policy available** and the **show event manager policy registered** commands. Depending on your release, you may need to copy one of the two Tcl scripts from the configuration examples section in this document. In the following example, details about the sample policy tm_cli_cmd.tcl are displayed on the screen.

Example:

```
Device# show event manager policy available detailed tm_cli_cmd.tcl
```

Step 3 Cut and paste the contents of the sample policy displayed on the screen to a text editor.

Use the edit and copy functions to move the contents from the device to a text editor on another device.

Step 4 Edit the policy and save it with a new filename.

Use the text editor to modify the policy as a Tcl script. For file naming conventions, see the Cisco File Naming Convention for EEM, on page 601.

Step 5 Copy the new file back to the device flash memory.

Copy the file to the flash file system on the device--typically disk0:. For more details about copying files, see the "Using the Cisco IOS File System" chapter in the *Configuration Fundamentals Configuration Guide*.

Copy the file to the flash file system on the device--typically bootflash:. For more details about copying files, see the "Using the Cisco IOS File System" chapter in the *Configuration Fundamentals Configuration Guide*.

Step 6 **configure terminal**

Enters global configuration mode.

Example:

```
Device# configure terminal
```

Step 7 **event manager directory user** {**library** *path*| **policy** *path*}

Specifies a directory to use for storing user library files or user-defined EEM policies. In the following example, the user_library directory on disk0 is specified as the directory for storing user library files.

Specifies a directory to use for storing user library files or user-defined EEM policies. In the following example, the user_library directory on bootflash is specified as the directory for storing user library files.

Example:

```
Device(config)# event manager directory user library disk0:/user_library
```

```
Device(config)# event manager directory user library bootflash:/user_library
```

Step 8 **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

Registers the EEM policy to be run when the specified event defined within the policy occurs. In the following example, the new EEM policy named test.tcl is registered as a user-defined policy.

Example:

```
Device(config)# event manager policy test.tcl type user
```

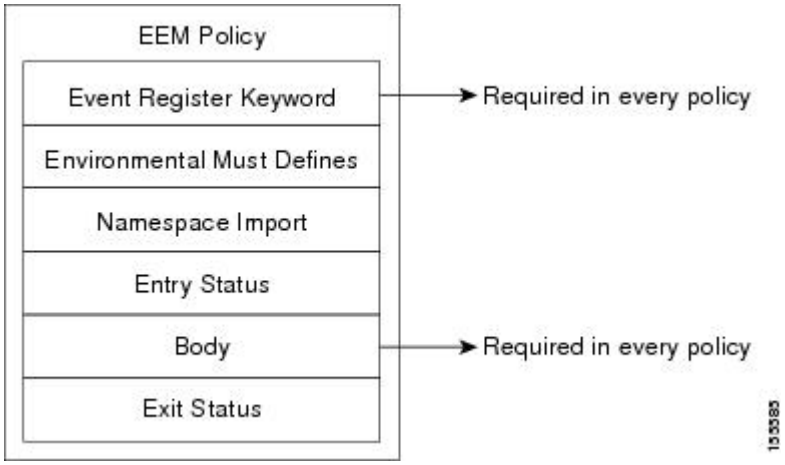
Programming EEM Policies with Tcl

Perform this task to help you program a policy using Tcl command extensions. We recommend that you copy an existing policy and modify it. There are two required parts that must exist in an EEM Tcl policy: the **event_register** Tcl command extension and the body. All other sections shown in the Tcl Policy Structure and Requirements, on page 615 concept are optional.

Tcl Policy Structure and Requirements

All EEM policies share the same structure, shown in the figure below. There are two parts of an EEM policy that are required: the **event_register** Tcl command extension and the body. The remaining parts of the policy are optional: environment must defines, namespace import, entry status, and exit status.

Figure 17: Tcl Policy Structure and Requirements



The start of every policy must describe and register the event to detect using an **event_register** Tcl command extension. This part of the policy schedules the running of the policy . The following example Tcl code shows how to register the **event_register_timer** Tcl command extension:

```
::cisco::eem::event_register_timer cron name crontimer2 cron_entry $_cron_entry maxrun 240
```

The environment must defines section is optional and includes the definition of environment variables. The following example Tcl code shows how to check for, and define, some environment variables.

```
# Check if all the env variables that we need exist.
# If any of them does not exist, print out an error msg and quit.
if {![info exists _email_server]} {
    set result \
      "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
      "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
      "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
```

The namespace import section is optional and defines code libraries. The following example Tcl code shows how to configure a namespace import section.

```
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
```

The body of the policy is a required structure and might contain the following:

- The **event_reqinfo**event information Tcl command extension that is used to query the EEM for information about the detected event.
- The action Tcl command extensions, such as **action_syslog**, that are used to specify EEM specific actions.

- The system information Tcl command extensions, such as **sys_reqinfo_routername**, that are used to obtain general system information.
- Use of the SMTP library (to send e-mail notifications) or the CLI library (to run CLI commands) from a policy.
- The **context_save** and **context_retrieve** Tcl command extensions that are used to save Tcl variables for use by other policies.

The following example Tcl code shows the code to query an event and log a message as part of the body section.

```
# Query the event info and log a message.
array set arr_einfo [event_reqinfo]

if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)

# Log a message.
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]

action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

EEM Entry Status

The entry status part of an EEM policy is used to determine if a prior policy has been run for the same event, and to determine the exit status of the prior policy. If the _entry_status variable is defined, a prior policy has already run for this event. The value of the _entry_status variable determines the return code of the prior policy.

Entry status designations may use one of three possible values: 0 (previous policy was successful), Not=0 (previous policy failed), and Undefined (no previous policy was executed).

EEM Exit Status

When a policy finishes running its code, an exit value is set. The exit value is used by the Embedded Event Manager to determine whether or not to apply the default action for this event, if any. A value of zero means do not perform the default action. A value of nonzero means perform the default action. The exit status will be passed to subsequent policies that are run for the same event.

EEM Policies and Cisco Error Number

Some EEM Tcl command extensions set a Cisco Error Number Tcl global variable _cerrno. Whenever _cerrno is set, four other Tcl global variables are derived from _cerrno and are set along with it (_cerr_sub_num, _cerr_sub_err, _cerr_posix_err, and _cerr_str).

For example, the **action_syslog** command in the example below sets these global variables as a side effect of the command execution:

```
action_syslog priority warning msg "A sample message generated by action_syslog"
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

_cerrno: 32-Bit Error Return Values

The _cerrno set by a command can be represented as a 32-bit integer of the following form:

```
XYSSSSSSSSSSSSSEEEEEEEEPPPPPPPPP
```

For example, the following error return value might be returned from an EEM Tcl command extension:

```
862439AE
```

This number is interpreted as the following 32-bit value:

```
10000110001001000011100110101110
```

This 32-bit integer is divided up into the five variables shown in the table below.

Table 62: _cerrno: 32-Bit Error Return Value Variables

Variable	Description
XY	The error class (indicates the severity of the error). This variable corresponds to the first two bits in the 32-bit error return value; 10 in the case above, which indicates CERR_CLASS_WARNING: See the table below for the four possible error class encodings specific to this variable.
SSSSSSSSSSSSS	The subsystem number that generated the most recent error (13 bits = 8192 values). This is the next 13 bits of the 32-bit sequence, and its integer value is contained in $_cerr_sub_num.
Variable	Description
EEEEEEEE	The subsystem specific error number (8 bits = 256 values). This segment is the next 8 bits of the 32-bit sequence, and the string corresponding to this error number is contained in $_cerr_sub_err.
PPPPPPPP	The pass-through POSIX error code (9 bits = 512 values). This represents the last of the 32-bit sequence, and the string corresponding to this error code is contained in $_cerr_posix_err.

Error Class Encodings for XY

The first variable, XY, references the possible error class encodings shown in the table below.

Table 63: Error Class Encodings

00	CERR_CLASS_SUCCESS
01	CERR_CLASS_INFO
10	CERR_CLASS_WARNING
11	CERR_CLASS_FATAL

An error return value of zero means SUCCESS.

SUMMARY STEPS

1. **enable**
2. **show event manager policy available detailed** *policy-filename*
3. Cut and paste the contents of the sample policy displayed on the screen to a text editor.
4. Define the required **event_register** Tcl command extension.
5. Add the appropriate namespace under the ::cisco hierarchy.
6. Program the must defines section to check for each environment variable that is used in this policy.
7. Program the body of the script.
8. Check the entry status to determine if a policy has previously run for this event.
9. Check the exit status to determine whether or not to apply the default action for this event, if a default action exists.
10. Set Cisco Error Number (_cerrno) Tcl global variables.
11. Save the Tcl script with a new filename, and copy the Tcl script to the device.
12. **configure terminal**
13. **event manager directory user** {**library** *path*| **policy** *path*}
14. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]
15. Cause the policy to execute, and observe the policy.
16. Use debugging techniques if the policy does not execute correctly.

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 show event manager policy available detailed *policy-filename*

Displays the actual specified sample policy including details about the environment variables used by the policy and instructions for running the policy. The **detailed** keyword was introduced for the **show event manager policy available** and the **show event manager policy registered** commands. Depending on your release, you must copy one of the two Tcl scripts from the configuration examples section in this document. In the following example, details about the sample policy tm_cli_cmd.tcl are displayed on the screen.

Example:

```
Device# show event manager policy available detailed tm_cli_cmd.tcl
```

Step 3 Cut and paste the contents of the sample policy displayed on the screen to a text editor.

Use the edit and copy functions to move the contents from the device to a text editor on another device. Use the text editor to edit the policy as a Tcl script.

Step 4 Define the required **event_register** Tcl command extension.

Choose the appropriate **event_register** Tcl command extension from the table below for the event that you want to detect, and add it to the policy.

Table 64: EEM Event Registration Tcl Command Extensions

Event Registration Tcl Command Extensions
event_register_appl
event_register_cli
event_register_counter
event_register_gold
event_register_interface
event_register_ioswdsysmon
event_register_ipsla
event_register_nf
event_register_none
event_register_oir
event_register_process
event_register_resource
event_register_rf
event_register_routing
event_register_rpc
event_register_snmp
event_register_snmp_notification
event_register_snmp_object
event_register_syslog
event_register_timer

Event Registration Tcl Command Extensions
event_register_timer_subscriber
event_register_track
event_register_wdsysmon

Step 5 Add the appropriate namespace under the ::cisco hierarchy.

Policy developers can use the new namespace ::cisco in Tcl policies in order to group all the extensions used by Cisco IOS EEM. There are two namespaces under the ::cisco hierarchy, and the table below shows which category of EEM Tcl command extension belongs under each namespace.

Table 65: Cisco IOS EEM Namespace Groupings

Namespace	Category of Tcl Command Extension
::cisco::eem	EEM event registration
	EEM event information
	EEM event publish
	EEM action
	EEM utility
	EEM context library
	EEM system information
	CLI library
::cisco::lib	SMTP library

Note Make sure that you import the appropriate namespaces or use the qualified command names when using the above commands.

Step 6 Program the must defines section to check for each environment variable that is used in this policy.

This is an optional step. Must defines are a section of the policy that tests whether any EEM environment variables that are required by the policy are defined before the recovery actions are taken. The must defines section is not required if the policy does not use any EEM environment variables. EEM environment variables for EEM scripts are Tcl global variables that are defined external to the policy before the policy is run. To define an EEM environment variable, use the Embedded Event Manager configuration command **event manager environment** CLI command. By convention all Cisco EEM environment variables begin with " _ " (an underscore). In order to avoid future conflict, customers are urged not to define new variables that start with " _ ".

Note You can display the Embedded Event Manager environment variables set on your system by using the **show event manager environment** privileged EXEC command.

For example, Embedded Event Manager environment variables defined by the sample policies include e-mail variables. The sample policies that send e-mail must have the variables shown in the table below set in order to function properly.

The table below describes the e-mail-specific environment variables used in the sample EEM policies.

Table 66: E-mail-Specific Environmental Variables Used by the Sample Policies

Environment Variable	Description	Example
_email_server	A Simple Mail Transfer Protocol (SMTP) mail server used to send e-mail.	The e-mail server name can be in any one of the following template formats: • username:password@host • username@host • host
_email_to	The address to which e-mail is sent.	engineering@example.com
_email_from	The address from which e-mail is sent.	devtest@example.com
_email_cc	The address to which the e-mail must be copied.	manager@example.com

The following example of a must define section shows how to program a check for e-mail-specific environment variables.

Example of Must Defines

Example:

```
if {![info exists _email_server]} {
    set result \
        "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
        "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
        "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
}
if {![info exists _email_cc]} {
    set result \
        "Policy cannot be run: variable _email_cc has not been set"
    error $result $errorInfo
}
```

Step 7 Program the body of the script.

In this section of the script, you can define any of the following:

- The **event_reqinfo**event information Tcl command extension that is used to query the EEM for information about the detected event.
- The action Tcl command extensions, such as **action_syslog**, that are used to specify EEM specific actions.
- The system information Tcl command extensions, such as **sys_reqinfo_routername**, that are used to obtain general system information.

- The **context_save** and **context_retrieve** Tcl command extensions that are used to save Tcl variables for use by other policies.
- Use of the SMTP library (to send e-mail notifications) or the CLI library (to run CLI commands) from a policy.

Step 8 Check the entry status to determine if a policy has previously run for this event.

If the prior policy is successful, the current policy may or may not require execution. Entry status designations may use one of three possible values: 0 (previous policy was successful), Not=0 (previous policy failed), and Undefined (no previous policy was executed).

Step 9 Check the exit status to determine whether or not to apply the default action for this event, if a default action exists.

A value of zero means do not perform the default action. A value of nonzero means perform the default action. The exit status will be passed to subsequent policies that are run for the same event.

Step 10 Set Cisco Error Number (_cerrno) Tcl global variables.

Some EEM Tcl command extensions set a Cisco Error Number Tcl global variable _cerrno. Whenever _cerrno is set, four other Tcl global variables are derived from _cerrno and are set along with it (_cerr_sub_num, _cerr_sub_err, _cerr_posix_err, and _cerr_str).

For example, the **action_syslog** command in the example below sets these global variables as a side effect of the command execution:

Example:

```
action_syslog priority warning msg "A sample message generated by action_syslog
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

Step 11 Save the Tcl script with a new filename, and copy the Tcl script to the device.

Embedded Event Manager policy filenames adhere to the following specification:

- An optional prefix--Mandatory.--indicating, if present, that this is a system policy that should be registered automatically at boot time if it is not already registered. For example: Mandatory.sl_text.tcl.
- A filename body part containing a two-character abbreviation for the first event specified; an underscore character part; and a descriptive field part further identifying the policy.
- A filename suffix part defined as .tcl.

For more details, see the *Cisco File Naming Convention for EEM*.

Copy the file to the flash file system on the device--typically disk0:. For more details about copying files, see the "Using the Cisco IOS File System" chapter in the Cisco IOS Configuration Fundamentals Configuration Guide .

Copy the file to the flash file system on the device--typically bootflash:. For more details about copying files, see the "Using the Cisco IOS File System" chapter in the Cisco IOS Configuration Fundamentals Configuration Guide .

Step 12 **configure terminal**

Enters global configuration mode.

Example:

```
Device# configure terminal
```

Step 13 **event manager directory user** {**library** *path*| **policy** *path*}

Specifies a directory to use for storing user library files or user-defined EEM policies. In the following example, the user_library directory on disk0 is specified as the directory for storing user library files.

Specifies a directory to use for storing user library files or user-defined EEM policies. In the following example, the user_library directory on bootflash is specified as the directory for storing user library files.

Example:

```
Device(config)# event manager directory user library disk0:/user_library
```

```
Device(config)# event manager directory user library bootflash:/user_library
```

Step 14 **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

Registers the EEM policy to be run when the specified event defined within the policy occurs. In the following example, the new EEM policy named cl_mytest.tcl is registered as a user-defined policy.

Example:

```
Device(config)# event manager policy cl_mytest.tcl type user
```

Step 15 Cause the policy to execute, and observe the policy.

To test that the policy runs, generate the conditions that will cause the policy to execute and observe that the policy runs as expected.

Step 16 Use debugging techniques if the policy does not execute correctly.

Use the Cisco IOS **debug event manager** CLI command with its various keywords to debug issues. Refer to the *Troubleshooting Tips* section for details about using Tcl-specific keywords.

Troubleshooting Tips

- Use the **debug event manager tcl commands** CLI command to debug issues with Tcl extension commands. When enabled, this command displays all data that is passed in and read back from the TTY session that handles the CLI interactions. This data helps ensure users that the commands they are passing to the CLI are valid.
- The CLI library allows users to run CLI commands and obtain the output of commands in Tcl. Use the **debug event manager tcl cli-library** CLI command to debug issues with the CLI library.
- The SMTP library allows users to send e-mail messages to an SMTP e-mail server. Use the **debug event manager tcl smtp_library** CLI command to debug issues with the SMTP library. When enabled, this command displays all data that is passed in and read back from the SMTP library routines. This data helps ensure users that the commands they are passing to the SMTP library are valid.
- Tcl is a flexible language that allows you to override commands. For example, you can modify the **set** command and create a version of the **set** command that displays a message when a scalar variable is set. When the **set** command is entered in a policy, a message is displayed anytime a scalar variable is set,

and this provides a way to debug scalar variables. To view an example of this debugging technique, see the Tracing Tcl set Command Operations Example, on page 653.

To view examples of the some of these debugging techniques, see the Debugging Embedded Event Manager Policies Examples, on page 650.

Creating an EEM User Tcl Library Index

Perform this task to create an index file that contains a directory of all the procedures contained in a library of Tcl files. This task allows you to test library support in EEM Tcl. In this task, a library directory is created to contain the Tcl library files, the files are copied into the directory, and an index tclIndex) is created that contains a directory of all the procedures in the library files. If the index is not created, the Tcl procedures will not be found when an EEM policy is run that references a Tcl procedure.

SUMMARY STEPS

1. On your workstation (UNIX, Linux, PC, or Mac) create a library directory and copy the Tcl library files into the directory.
2. **tclsh**
3. **auto_mkindex** *directory_name* ***.tcl**
4. Copy the Tcl library files, and the tclIndex file to the directory used for storing user library files on the target device.
5. Copy a user-defined EEM policy file written in Tcl to the directory used for storing user-defined EEM policies on the target device.
6. **enable**
7. **configure terminal**
8. **event manager directory user library** *path*
9. **event manager directory user policy** *path*
10. **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap]**
11. **event manager run** *policy-name*

DETAILED STEPS

Step 1 On your workstation (UNIX, Linux, PC, or Mac) create a library directory and copy the Tcl library files into the directory.

The following example files can be used to create a tclIndex on a workstation running the Tcl shell:

lib1.tcl

Example:

```
proc test1 {} {
    puts "In procedure test1"
}

proc test2 {} {
    puts "In procedure test2"
}
```

lib2.tcl

Example:

```
proc test3 {} {
    puts "In procedure test3"
}
```

Step 2 **tclsh**

Use this command to enter the Tcl shell.

Example:

```
workstation% tclsh
```

Step 3 **auto_mkindex** *directory_name* ***.tcl**

Use the **auto_mkindex** command to create the tclIndex file. The tclIndex file that contains a directory of all the procedures contained in the Tcl library files. We recommend that you run auto_mkindex inside a directory because there can only be a single tclIndex file in any directory and you may have other Tcl files to be grouped together. Running auto_mkindex in a directory determines which tcl source file or files are indexed using a specific tclIndex.

Example:

```
workstation% auto_mkindex eem_library *.tcl
```

The following example TclIndex is created when the lib1.tcl and lib2.tcl files are in a library file directory and the **auto_mkindex** command is run.

tclIndex

Example:

```
# Tcl autoload index file, version 2.0
# This file is generated by the "auto_mkindex" command
# and sourced to set up indexing information for one or
# more commands.  Typically each line is a command that
# sets an element in the auto_index array, where the
# element name is the name of a command and the value is
# a script that loads the command.

set auto_index(test1) [list source [file join $dir lib1.tcl]]
set auto_index(test2) [list source [file join $dir lib1.tcl]]
set auto_index(test3) [list source [file join $dir lib2.tcl]]
```

Step 4 Copy the Tcl library files, and the tclIndex file to the directory used for storing user library files on the target device.

Step 5 Copy a user-defined EEM policy file written in Tcl to the directory used for storing user-defined EEM policies on the target device.

The directory for storing user-defined EEM policies can be the same directory used in Step 4. The following example user-defined EEM policy can be used to test the Tcl library support in EEM.

libtest.tcl

Example:

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
```

```
global auto_index auto_path

puts [array names auto_index]

if { [catch {test1} result]} {
    puts "calling test1 failed result = $result $auto_path"
}

if { [catch {test2} result]} {
    puts "calling test2 failed result = $result $auto_path"
}
if { [catch {test3} result]} {
    puts "calling test3 failed result = $result $auto_path"
}
```

Step 6 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 7 **configure terminal**

Enables global configuration mode.

Example:

```
Device# configure terminal
```

Step 8 **event manager directory user library** *path*

Use this command to specify the EEM user library directory; this is the directory to which the files were copied.

Example:

```
Device(config)# event manager directory user library disk2:/eem_library
```

Step 9 **event manager directory user policy** *path*

Use this command to specify the EEM user policy directory; this is the directory to which the file were copied.

Example:

```
Device(config)# event manager directory user policy disk2:/eem_policies
```

Step 10 **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap]**

Use this command to register a user-defined EEM policy. In this example, the policy named libtest.tcl is registered.

Example:

```
Device(config)# event manager policy libtest.tcl
```

Step 11 **event manager run** *policy-name*

Use this command to manually run an EEM policy. In this example, the policy named libtest.tcl is run to test the Tcl support in EEM. The example output shows that the test for Tcl support in EEM was successful.

Example:

```
Device(config)# event manager run libtest.tcl
The following output is displayed:
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test1
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test2
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test3
```

Creating an EEM User Tcl Package Index

Perform this task to create a Tcl package index file that contains a directory of all the Tcl packages and version information contained in a library of Tcl package files. Tcl packages are supported, depending on your release, using the Tcl **package** keyword.

Tcl packages are located in either the EEM system library directory or the EEM user library directory. When a **package require** Tcl command is executed, the user library directory is searched first for a pkgIndex.tcl file. If the pkgIndex.tcl file is not found in the user directory, the system library directory is searched. In this task, a Tcl package directory--the pkgIndex.tcl file--is created in the appropriate library directory using the **pkg_mkIndex** command to contain information about all of the Tcl packages contained in the directory along with version information. If the index is not created, the Tcl packages will not be found when an EEM policy is run that contains a **package require** Tcl command.

Using the Tcl package support in EEM, users can gain access to packages such as XML_RPC for Tcl. When the Tcl package index is created, a Tcl script can easily make an XML-RPC call to an external entity.

Note Packages implemented in C programming code are not supported in EEM.

SUMMARY STEPS

1. On your workstation (UNIX, Linux, PC, or Mac) create a library directory and copy the Tcl package files into the directory.
2. **tclsh**
3. **pkg_mkindex** *directory_name* ***.tcl**
4. Copy the Tcl library files and the pkgIndex file to the directory used for storing user library files on the target device.
5. Copy a user-defined EEM policy file written in Tcl to the directory used for storing user-defined EEM policies on the target device. The directory can be the same directory used.
6. **enable**
7. **configure terminal**
8. **event manager directory user library** *path*
9. **event manager directory user policy** *path*
10. **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap**]
11. **event manager run** *policy-name*

DETAILED STEPS

Step 1 On your workstation (UNIX, Linux, PC, or Mac) create a library directory and copy the Tcl package files into the directory.

Step 2 **tclsh**

Use this command to enter the Tcl shell.

Example:

```
workstation% tclsh
```

Step 3 **pkg_mkindex** *directory_name* ***.tcl**

Use the **pkg_mkindex** command to create the pkgIndex file. The pkgIndex file contains a directory of all the packages contained in the Tcl library files. We recommend that you run pkg_mkindex inside a directory because there can only be a single pkgIndex file in any directory and you may have other Tcl files to be grouped together. Running pkg_mkindex in a directory determines which Tcl package file or files are indexed using a specific pkgIndex.

Example:

```
workstation% pkg_mkindex eem_library *.tcl
```

The following example pkgIndex is created when some Tcl package files are in a library file directory and the **pkg_mkindex** command is run.

pkgIndex

Example:

```
# Tcl package index file, version 1.1
# This file is generated by the "pkg_mkIndex" command
# and sourced either when an application starts up or
# by a "package unknown" script. It invokes the
# "package ifneeded" command to set up package-related
# information so that packages will be loaded automatically
# in response to "package require" commands. When this
# script is sourced, the variable $dir must contain the
# full path name of this file's directory.
package ifneeded xmlrpc 0.3 [list source [file join $dir xmlrpc.tcl]]
```

Step 4 Copy the Tcl library files and the pkgIndex file to the directory used for storing user library files on the target device.

Step 5 Copy a user-defined EEM policy file written in Tcl to the directory used for storing user-defined EEM policies on the target device. The directory can be the same directory used.

The directory for storing user-defined EEM policies can be the same directory. The following example user-defined EEM policy can be used to test the Tcl package support in EEM.

packagetest.tcl

Example:

```
::cisco::eem::event_register_none maxrun 1000000.000
#
# test if xmlrpc available
#
#
# Namespace imports
```

```
#
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
#
package require xmlrpc
puts "Did you get an error?"
```

Step 6 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 7 **configure terminal**

Enables global configuration mode.

Example:

```
Device# configure terminal
```

Step 8 **event manager directory user library** *path*

Use this command to specify the EEM user library directory; this is the directory to which the files were copied.

Example:

```
Device(config)# event manager directory user library disk2:/eem_library
```

Step 9 **event manager directory user policy** *path*

Use this command to specify the EEM user policy directory; this is the directory to which the file was copied.

Example:

```
Device(config)# event manager directory user policy disk2:/eem_policies
```

Step 10 **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap**]

Use this command to register a user-defined EEM policy. In this example, the policy named packagetest.tcl is registered.

Example:

```
Device(config)# event manager policy packagetest.tcl
```

Step 11 **event manager run** *policy-name*

Use this command to manually run an EEM policy. In this example, the policy named packagetest.tcl is run to test the Tcl package support in EEM.

Example:

```
Device(config)# event manager run packagetest.tcl
```

Configuration Examples for Writing Embedded Event Manager Policies Using Tcl

Assigning a Username for a Tcl Session Examples

The following example shows how to set a username to be associated with a Tcl session. If you are using authentication, authorization, and accounting (AAA) security and implement authorization on a command basis, you should use the **event manager session cli username** command to set a username to be associated with a Tcl session. The username is used when a Tcl policy executes a CLI command. TACACS+ verifies each CLI command using the username associated with the Tcl session that is running the policy. Commands from Tcl policies are not usually verified because the device must be in privileged EXEC mode to register the policy. In the example, the username is yourname, and this is the username that is used whenever a CLI command session is initiated from within an EEM policy.

```
configure terminal
 event manager session cli username yourname
 end
```

EEM Event Detector Demo Examples

EEM Sample Policy Descriptions

This configuration example features some of the sample EEM policies:

- ap_perf_test_base_cpu.tcl--Is run to measure the the CPU performance of EEM policies.
- no_perf_test_init.tcl--Is run to measure the CPU performance of EEM policies.
- sl_intf_down.tcl--Is run when a configurable syslog message is logged. It executes up to two configurable CLI commands and e-mails the results.
- tm_cli_cmd.tcl--Is run using a configurable CRON entry. It executes a configurable CLI command and e-mails the results.
- tm_crash_reporter.tcl--Is run 5 seconds after it is registered and 5 seconds after the device boots up. When triggered, the script attempts to find the reload reason. If the reload reason was due to a crash, the policy searches for the related crashinfo file and sends this information to a URL location specified by the user in the environment variable _crash_reporter_url.
- tm_fsys_usage.tcl--This policy runs using a configurable CRON entry and monitors disk space usage. A syslog message is displayed if disk space usage crosses configurable thresholds.

Event Manager Environment Variables for the Sample Policies

Event manager environment variables are Tcl global variables that are defined external to the EEM policy before the policy is registered and run. The sample policies require three of the e-mail environment variables to be set (see above section for a list of the e-mail variables); only _email_cc is optional. Other required and optional variable settings are outlined in the following tables.

The table below describes the EEM environment variables that must be set before the ap_perf_test_base_cpu.tcl sample policy is run.

Table 67: Environment Variables Used in the ap_perf_test_base_cpu.tcl Policy

Environment Variable	Description	Example
_perf_iterations	The number of iterations over which to run the measurement.	**100**
_perf_cmd1	The first non interactive CLI command that is executed as part of the measurement test. This variable is optional and need not be specified.	**enable**
_perf_cmd2	The second non interactive CLI command that is as part of the measurement test. To use _perf_cmd2, _perf_cmd1 must be defined. This variable is optional and need not be specified.	**show version**
_perf_cmd3	The third non interactive CLI command that is as part of the measurement test. To use _perf_cmd3, _perf_cmd1 must be defined. This variable is optional and need not be specified.	**show interface counters protocol status**

The table below describes the EEM environment variables that must be set before the no_perf_test_init.tcl sample policy is run.

Table 68: Environment Variables Used in the no_perf_test_init.tcl Policy

Environment Variable	Description	Example
_perf_iterations	The number of iterations over which to run the measurement.	**100**
_perf_cmd1	The first non interactive CLI command that is executed as part of the measurement test. This variable is optional and need not be specified.	**enable**
_perf_cmd2	The second non interactive CLI command that is as part of the measurement test. To use _perf_cmd2, _perf_cmd1 must be defined. This variable is optional and need not be specified.	**show version**
_perf_cmd3	The third non interactive CLI command that is as part of the measurement test. To use _perf_cmd3, _perf_cmd1 must be defined. This variable is optional and need not be specified.	**show interface counters protocol status**

The table below describes the EEM environment variables that must be set before the sl_intf_down.tcl sample policy is run.

Table 69: Environment Variables Used in the sl_intf_down.tcl Policy

Environment Variable	Description	Example
_config_cmd1	The first configuration command that is executed.	**interface Ethernet1/0**

Environment Variable	Description	Example
_config_cmd2	The second configuration command that is executed. This variable is optional and need not be specified.	**no shutdown**
_syslog_pattern	A regular expression pattern match string that is used to compare syslog messages to determine when the policy runs.	.*UPDOWN.*FastEthernet0/0.*

The table below describes the EEM environment variables that must be set before the tm_cli_cmd.tcl sample policy is run.

Table 70: Environment Variables Used in the tm_cli_cmd.tcl Policy

Environment Variable	Description	Example
_cron_entry	A CRON specification that determines when the policy will run.	0-59/1 0-23/1 * * 0-7
_show_cmd	The CLI command to be executed when the policy is run.	**show version**

The table below describes the EEM environment variables that must be set before the tm_crash_reporter.tcl sample policy is run.

Table 71: Environment Variables Used in the tm_crash_reporter.tcl Policy

Environment Variable	Description	Example
_crash_reporter_debug	A value that identifies whether debug information for tm_crash_reporter.tcl will be enabled. This variable is optional and need not be specified.	1
_crash_reporter_url	The URL location to which the crash report is sent.	http://www.example.com/fm/interface_tm.cgi

The table below describes the EEM environment variables that must be set before the tm_fsys_usage.tcl sample policy is run.

Table 72: Environment Variables Used in the tm_fsys_usage.tcl Policy

Environment Variable	Description	Example
_tm_fsys_usage_cron	A CRON specification that is used in the **event_register** Tcl command extension. If unspecified, the tm_fsys_usage.tcl policy is triggered once per minute. This variable is optional and need not be specified.	0-59/1 0-23/1 * * 0-7
_tm_fsys_usage_debug	When this variable is set to a value of 1, disk usage information is displayed for all entries in the system. This variable is optional and need not be specified.	1

Environment Variable	Description	Example
_tm_fsys_usage_ freebytes	Free byte threshold for systems or specific prefixes. If free space falls below a given value, a warning is displayed. This variable is optional and need not be specified.	disk2:98000000
_tm_fsys_usage_percent	Disk usage percentage thresholds for systems or specific prefixes. If the disk usage percentage exceeds a given percentage, a warning is displayed. If unspecified, the default disk usage percentage is 80 percent for all systems. This variable is optional and need not be specified.	nvram:25 disk2:5

Registration of Some EEM Policies

Some EEM policies must be unregistered and then reregistered if an EEM environment variable is modified after the policy is registered. The event_register_*xxx* statement that appears at the start of the policy contains some of the EEM environment variables, and this statement is used to establish the conditions under which the policy is run. If the environment variables are modified after the policy has been registered, the conditions may become invalid. To avoid any errors, the policy must be unregistered and then reregistered. The following variables are affected:

- _cron_entry in the tm_cli_cmd.tcl policy
- _syslog_pattern in the sl_intf_down.tcl policy

Basic Configuration Details for All Sample Policies

To allow e-mail to be sent from the Embedded Event Manager, the **hostname** and **ip domain-name** commands must be configured. The EEM environment variables must also be set. After a Cisco IOS image has been booted, use the following initial configuration, substituting appropriate values for your network. The environment variables for the tm_fsys_usage sample policy (see the table above) are all optional and are not listed here:

```
hostname cpu
ip domain-name example.com
event manager environment _email_server ms.example.net
event manager environment _email_to username@example.net
event manager environment _email_from engineer@example.net
event manager environment _email_cc projectgroup@example.net
event manager environment _cron_entry 0-59/2 0-23/1 * * 0-7
event manager environment _show_cmd show event manager policy registered
event manager environment _syslog_pattern .*UPDOWN.*FastEthernet0/0
event manager environment _config_cmd1 interface Ethernet1/0
event manager environment _config_cmd2 no shutdown
event manager environment _crash_reporter_debug 1
event manager environment _crash_reporter_url
http://www.example.com/fm/interface_tm.cgi
end
```

Using the Sample Policies

This section contains the following configuration scenarios to demonstrate how to use the some sample Tcl policies:

Running the Mandatory.go_*.tcl Sample Policy

There are GOLD TCL scripts for each test which runs as a part of GOLD EEM Policy. You can modify the TCL script for the test, specify the consecutive failure count, and also change the default corrective action. For example, one could chose to power down a linecard card, instead of reset or other CLI based actions.

For each registered test, a default TCL script is available, which can be registered with the system, and matches with the default action. This can be then overridden by modifying these scripts.

The following table shows a list of the mandatory polices that GOLD installed into EEM. Each of the policies performs some sort of action such as resetting the card or disabling the port.

GOLD Tcl Scripts	Test
Mandatory.go_asicsync.tcl	TestAsicSync
Mandatory.go_bootup.tcl	Common for all bootup tests.
Mandatory.go_fabric.tcl	TestFabricHealth
Mandatory.go_fabrich0.tcl	TestFabricCh0Health
Mandatory.go_fabrich1.tcl	TestFabricCh1Health
Mandatory.go_ipsec.tcl	TestIPSecEncrypDecrypPkt
Mandatory.go_mac.tcl	TestMacNotification
Mandatory.go_nondislp.tcl	TestNonDisruptiveLoopback
Mandatory.go_scratchreg.tcl	TestScratchRegister
Mandatory.go_sprping.tcl	TestSPRPInbandPing

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode, where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the mandatory.go_*.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command again to verify that the policy has been registered.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy Mandatory.go_spuriousisr.tcl
 end
show event manager policy registered
show event manager environment
```

Running the ap_perf_test_base_cpu.tcl and no_perf_test_init.tcl Sample Policies

These sample policies measures the CPU performance of EEM policies. The policies help find the average execution time of each EEM policy and uses the CLI library to execute the configuration commands specified in the EEM environment variables _perf_cmd1 and, optionally, _perf_cmd2 and _perf_cmd3.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode, where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, enter the **service timestamps debug datetime msec** command and then you can register the ap_perf_test_base_cpu.tcl and no_perf_test_init.tcl policies with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command again to verify that the policy has been registered.

The policies ap_perf_test_base_cpu.tcl and no_perf_test_init.tcl need to be registered together, as they run as a test suite. You can run the no_perf_test_init.tcl policy to start the tests. Analyze the results using the syslog messages from each iteration. The total number of iteration is specified by the variable _perf_iterations. Take the time difference and divide it by the total number of iterations to get the average execution time of each EEM policy.

```
enable
show event manager policy registered
show event manager policy available
show event manager environment
configure terminal
 service timestamps debug datetime msec
 event manager environment _perf_iterations 100
 event manager policy ap_perf_test_base_cpu.tcl
 event manager policy no_perf_test_init.tcl
 end
show event manager policy registered
show event manager policy available
show event manager environment
event manager run no_perf_test_init.tcl
```

Running the no_perf_test_init.tcl Sample Policy

This sample policy measures the the cpu performance of EEM policies. The policy helps to find the average execution time of each EEM policy and uses the CLI library to execute the configuration commands specified in the EEM environment variables _perf_cmd1 and, optionally, _perf_cmd2 and _perf_cmd3.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode, where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the no_perf_test_init.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command again to verify that the policy has been registered.

Analyze the results using the syslog messages from each iteration. The total number of iteration is specified by the variable _perf_iterations. Take the time difference and divide it by the total number of iterations to get the average execution time of each EEM policy.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy no_perf_test_init.tcl
 end
```

```
show event manager policy registered
show event manager environment
```

Running the sl_intf_down.tcl Sample Policy

This sample policy demonstrates the ability to modify the configuration when a syslog message with a specific pattern is logged. The policy gathers detailed information about the event and uses the CLI library to execute the configuration commands specified in the EEM environment variables _config_cmd1 and, optionally, _config_cmd2. An e-mail message is sent with the results of the CLI command.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode, where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the sl_intf_down.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command again to verify that the policy has been registered.

The policy runs when an interface goes down. Enter the **show event manager environment** command to display the current environment variable values. Unplug the cable (or configure a shutdown) for the interface specified in the _syslog_pattern EEM environment variable. The interface goes down, prompting the syslog daemon to log a syslog message about the interface being down, and the syslog event detector is called.

The syslog event detector reviews the outstanding event specifications and finds a match for interface status change. The EEM server is notified, and the server runs the policy that is registered to handle this event--sl_intf_down.tcl.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy sl_intf_down.tcl
 end
show event manager policy registered
show event manager environment
```

Running the tm_cli_cmd.tcl Sample Policy

This sample policy demonstrates the ability to periodically execute a CLI command and to e-mail the results. The CRON specification “0-59/2 0-23/1 * * 0-7” causes this policy to be run on the second minute of each hour. The policy gathers detailed information about the event and uses the CLI library to execute the configuration commands specified in the EEM environment variable _show_cmd. An e-mail message is sent with the results of the CLI command.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the tm_cli_cmd.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command to verify that the policy has been registered.

The timer event detector triggers an event for this case periodically according to the CRON string set in the EEM environment variable _cron_entry. The EEM server is notified, and the server runs the policy that is registered to handle this event--tm_cli_cmd.tcl.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_cli_cmd.tcl
 end
show event manager policy registered
```

Running the tm_crash_reporter.tcl Sample Policy

This sample policy demonstrates the ability to send an HTTP-formatted crash report to a URL location. If the policy registration is saved in the startup configuration file, the policy is triggered 5 seconds after bootup. When triggered, the script attempts to find the reload reason. If the reload reason was due to a crash, the policy searches for the related crashinfo file and sends this information to a URL location specified by the user in the environment variable _crash_reporter_url. A CGI script, interface_tm.cgi, has been created to receive the URL from the tm_crash_reporter.tcl policy and save the crash information in a local database on the target URL machine.

A Perl CGI script, interface_tm.cgi, has been created and is designed to run on a machine that contains an HTTP server and is accessible by the device that runs the tm_crash_reporter.tcl policy. The interface_tm.cgi script parses the data passed into it from tm_crash_reporter.tcl and appends the crash information to a text file, creating a history of all crashes in the system. Additionally, detailed information on each crash is stored in three files in a crash database directory that is specified by the user. Another Perl CGI script, crash_report_display.cgi, has been created to display the information stored in the database created by the interface_tm.cgi script. The crash_report_display.cgi script should be placed on the same machine that contains interface_tm.cgi. The machine should be running a web browser such as Internet Explorer or Netscape. When the crash_report_display.cgi script is run, it displays the crash information in a readable format.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the tm_crash_reporter.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command to verify that the policy has been registered.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_crash_reporter.tcl
 end
show event manager policy registered
```

Running the tm_fsys_usage.tcl Sample Policy

This sample policy demonstrates the ability to periodically monitor disk space usage and report through syslog when configurable thresholds have been crossed.

The following sample configuration demonstrates how to use this policy. Starting in user EXEC mode, enter the **enable** command at the device prompt. The device enters privileged EXEC mode, where you can enter the **show event manager policy registered** command to verify that no policies are currently registered. The next command is the **show event manager policy available** command to display which policies are available to be installed. After you enter the **configure terminal** command to reach global configuration mode, you can register the tm_fsys_usage.tcl policy with EEM using the **event manager policy** command. Exit from global configuration mode and enter the **show event manager policy registered** command again to verify that the policy has been registered. If you had configured any of the optional environment variables that are used in the tm_fsys_usage.tcl policy, the **show event manager environment** command displays the configured variables.

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_fsys_usage.tcl
 end
show event manager policy registered
show event manager environment
```

Programming Policies with Tcl Sample Scripts Example

This section contains some of the sample policies that are included as EEM system policies. For more details about these policies, see the EEM Event Detector Demo Examples, on page 631.

Mandatory.go_ipsec.tcl Sample Policy

The following sample policy for the TestIPSecEncrypDecrypPkt Test.

```
::cisco::eem::event_register_gold card all testing_type monitoring test_name Tes
tIPSecEncrypDecrypPkt consecutive_failure 6 platform_action 0 queue_priority las
t
#
#  GOLD TestIPSecEncrypDecrypPkt Test TCL script
#
#  March 2005, Hai Qiu
#
#  Copyright (c) 2005-2007 by cisco Systems, Inc.
#  All rights reserved.
#
#
# Register for TestIPSecEncrypDecrypPkt test even
# the elements for register the event
# card [all | card #]
# sub_card [all | sub_card #]
# severity_major | severity_minor | severity_normal  default :  severity_normal
# new_failure [true | false]  default: dont_care
# testing_type [bootup | ondemand | schedule | monitoring]
# test_name [ test name ]
# test_id [ test # ]
# consecutive_failure [ consecutive_failure # ]
# platform_action [action_flag]
# action_flag [ 0 | 1 | 2 ]
# queue_priority [ normal | low | high | last]   default: normal
#
# Note:
#    1: "card" element is required. If other elements are not specified,
#       treat them as dont care, or default.
```

```
#
#    2: action_flag is platform specific. It is up to platform to
#       determine what action need to be taken based on the value
#       For Cat6k platform
#       action_flag 0 : TCL script take action to reset card
#       action_flag 1 : TCL script doesn't take action to reset card
#       action_flag 2 : TCL script takes action to reset card for bootup diag
#                       when there is major error
#       action_flag 3 : TCL script doesn't take action to reset card for
#                       bootup diag when there is major error
#
#    3: "queue_priority last" would guarantee this policy will be executed last
#       if there are other EEM events in queue with queue priority other
#       than "last"
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# 1. query the information of latest triggered eem event
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
puts "GOLD EEM TCL policy for TestIPSecEncrypDecrypPkt"
#set msg [format "array=%s", array names arr_einfo]
#puts "msg $msg"
#set msg $arr_einfo(msg)
set card $arr_einfo(card)
set sub_card $arr_einfo(sub_card)
#set overall_result $arr_einfo(overall_result)
#puts "GOLD event msg recieved: $card/$sub_card overall_result= $overall_result"
# 2. execute the user-defined config commands
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# Use "diagn action mod mod# test testname default" command
# for default platform action
if [catch {cli_exec $cli1(fd) "diagnostic action mod $card test TestIPSecEncrypD
ecrypPkt default"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}
```

ap_perf_test_base_cpu.tcl Sample Policy

The following sample policy measures the CPU performance of EEM policies.

```
::cisco::eem::event_register_appl sub_system 798 type 9999
#----------------------------------
# EEM policy used for measuring the cpu performance of EEM policies.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005, 2006 by cisco Systems, Inc.
```

```
# All rights reserved.
#------------------
###
### Input arguments:
###
### arg1 $iter                    - current iteration count
###
### The following EEM environment variables are used:
###
### _perf_iterations (mandatory) - number of iterations over which we
###                                will run our measurement.
### Example:
### event manager environment _perf_iterations 100
###
### _perf_cmd1 (optional)        - optional non interactive cli command
###                                to be executed as part of the
###                                measurement test.
### Example:
### event manager environment _perf_cmd1 enable
###
### _perf_cmd2 (optional)        - optional non interactive cli command
###                to be executed as part of the
###                                measurement test.
###                                To use _perf_cmd2, _perf_cmd1 MUST
###                be defined.
### Example:
### event manager environment _perf_cmd2 show ver
###
### _perf_cmd3 (optional)        - optional non interactive cli command
###                                to be executed as part of the
###                                measurement test.
###                                To use _perf_cmd3, _perf_cmd1 MUST
###                                be defined.
### Example:
### event manager environment _perf_cmd3 show int counters protocol status
###
### Description:
###   Iterate through _perf_iterations of this policy.
###   It is up to the user to calculate the average
###   execution time based on the system timestamps.
###   Optional commands _perf_cmd1,
###   _perf_cmd2 and _perf_cmd3 are executed if defined.
###
###   A value of 100 is a good starting point.
###
### Outputs:
###   Console output.
###
### Usage example:
###   >conf t
###   >service timestamps debug datetime msec
###   >event manager environment _perf_iterations 100
###   >event manager policy ap_perf_base_cpu.tcl
###   >event manager policy no_perf_test_init.tcl
###   >end
###   2d19h: %SYS-5-CONFIG_I: Configured from console by console
###   >event manager run no_perf_test_init.tcl
###
###   Oct 16 14:57:17.284: %SYS-5-CONFIG_I: Configured from console by console
###   >event manager run no_perf_test_init.tcl
###
###   Oct 16 19:32:02.772: %HA_EM-6-LOG:
###     eem_policy/no_perf_test_init.tcl: EEM performance test start
###   Oct 16 19:32:03.115: %HA_EM-6-LOG:
```

```
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 1
###   Oct 16 19:32:03.467: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 2
###   ...
###   Oct 16 19:32:36.936: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 100
###   Oct 16 19:32:36.936: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test end
###
###   The user must calculate execution time and average time of execution.
###   In this example, total time = 19:32:36.936 - 19:32:02.772 = 34.164
###   Average script execution time = 341.64 milliseconds
###
# check if all the env variables we need exist
# If any of them doesn't exist, print out an error msg and quit
if {![info exists _perf_iterations]} {
    set result \
        "Policy cannot be run: variable _perf_iterations has not been set"
    error $result $errorInfo
}
# ensure our target iteration count > 0
if {$_perf_iterations <= 0} {
    set result \
        "Policy cannot be run: variable _perf_iterations <= 0"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
set iter $arr_einfo(data1)
set iter [expr $iter + 1]
# if _perf_cmd1 is defined
if {[info exists _perf_cmd1]} {
    # open the cli library
    if [catch {cli_open} result] {
        error $result $errorInfo
    } else {
        array set cli1 $result
    }
    # execute the comamnd defined in _perf_cmd1
    if [catch {cli_exec $cli1(fd) $_perf_cmd1} result] {
        error $result $errorInfo
    }
    # if _perf_cmd2 is defined
    if {[info exists _perf_cmd2]} {
        # execute the comamnd defined in _perf_cmd2
        if [catch {cli_exec $cli1(fd) $_perf_cmd2} result] {
            error $result $errorInfo
        } else {
            set cmd_output $result
        }
    }
    # if _perf_cmd3 is defined
    if {[info exists _perf_cmd3]} {
        # execute the comamnd defined in _perf_cmd3
        if [catch {cli_exec $cli1(fd) $_perf_cmd3} result] {
            error $result $errorInfo
        } else {
```

```
            set cmd_output $result
        }
    }
    # close the cli library
    if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
        error $result $errorInfo
    }
}

# log a message
set msg [format "EEM performance test iteration %s" $iter]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# use the context info from the previous run to determine when to end
if {$iter >= $_perf_iterations} {
    #log the final messages
    action_syslog priority info msg "EEM performance test end"
    if {$_cerrno != 0} {
        set result [format \
                "component=%s; subsys err=%s; posix err=%s;\n%s" \
                $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
        error $result
    }
    exit 0
}
# cause the next iteration to run
event_publish sub_system 798 type 9999 arg1 $iter
if {$_cerrno != 0} {
    set result [format \
            "component=%s; subsys err=%s; posix err=%s;\n%s" \
            $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

tm_cli_cmd.tcl Sample Policy

The following sample policy runs a configurable CRON entry. The policy executes a configurable Cisco IOS CLI command and e-mails the results. An optional log file can be defined to which the output is appended with a timestamp.

```
::cisco::eem::event_register_timer cron name crontimer2 cron_entry $
_cron_entry maxrun 240
#-----------------------------------------------------------------------
# EEM policy that will periodically execute a cli command and email the
# results to a user.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#-----------------------------------------------------------------------
### The following EEM environment variables are used:
###
### _cron_entry (mandatory)            - A CRON specification that determines
###                                      when the policy will run. See the
###                                      IOS Embedded Event Manager
###                                      documentation for more information
###                                      on how to specify a cron entry.
```

```
### Example: _cron_entry                0-59/1 0-23/1 * * 0-7
###
### _log_file (mandatory without _email_....)
###                                     - A filename to append the output to.
###                                       If this variable is defined, the
###                                       output is appended to the specified
###                                       file with a timestamp added.
### Example: _log_file                  disk0:/my_file.log
###
### _email_server (mandatory without _log_file)
###                                     - A Simple Mail Transfer Protocol (SMTP)
###                                       mail server used to send e-mail.
### Example: _email_server              mailserver.example.com
###
### _email_from (mandatory without _log_file)
###                                     - The address from which e-mail is sent.
### Example: _email_from                devtest@example.com
###
### _email_to (mandatory without _log_file)
###                                     - The address to which e-mail is sent.
### Example: _email_to                  engineering@example.com
###
### _email_cc (optional)                - The address to which the e-mail must
###                                       be copied.
### Example: _email_cc                  manager@example.com
###
### _show_cmd (mandatory)               - The CLI command to be executed when
###                                       the policy is run.
### Example: _show_cmd                  show version
###
# check if all required environment variables exist
# If any required environment variable does not exist, print out an error msg and quit
if {![info exists _log_file]} {
    if {![info exists _email_server]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_server has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_from]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_from has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_to]} {
 set result \
  "Policy cannot be run: variable _log_file ore _email_to has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_cc]} {
 #_email_cc is an option, must set to empty string if not set.
 set _email_cc ""
    }
}
if {![info exists _show_cmd]} {
    set result \
        "Policy cannot be run: variable _show_cmd has not been set"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info and log a message
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
```

```
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)
# log a message
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
 $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 1. execute the command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# save exact execution time for command
set time_now [clock seconds]
# execute command
if [catch {cli_exec $cli1(fd) $_show_cmd} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
    # format output: remove trailing router prompt
    regexp {\n*(.*\n)([^\n]*)$} $result dummy cmd_output
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}

# 2. log the success of the CLI command
set msg [format "Command \"%s\" executed successfully" $_show_cmd]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 3. if _log_file is defined, then attach it to the file
if {[info exists _log_file]} {
    # attach output to file
    if [catch {open $_log_file a+} result] {
        error $result
    }
    set fileD $result
    # save timestamp of command execution
    #      (Format = 00:53:44 PDT Mon May 02 2005)
    set time_now [clock format $time_now -format "%T %Z %a %b %d %Y"]
    puts $fileD "%%% Timestamp = $time_now"
    puts $fileD $cmd_output
    close $fileD
}
# 4. if _email_server is defined send the email out
if {[info exists _email_server]} {
    set routername [info hostname]
    if {[string match "" $routername]} {
```

```
 error "Host name is not configured"
    }
    if [catch {smtp_subst [file join $tcl_library email_template_cmd.tm]} \
     result] {
 error $result $errorInfo
    }
    if [catch {smtp_send_email $result} result] {
 error $result $errorInfo
    }
}


::cisco::eem::event_register_timer cron name crontimer2 cron_entry $
_cron_entry maxrun 240
#-----------------------------------------------------------------
# EEM policy that will periodically execute a cli command and email the
# results to a user.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#-----------------------------------------------------------------
### The following EEM environment variables are used:
###
### _cron_entry (mandatory)            - A CRON specification that determines
###                                      when the policy will run. See the
###                                      IOS Embedded Event Manager
###                                      documentation for more information
###                                      on how to specify a cron entry.
### Example: _cron_entry                 0-59/1 0-23/1 * * 0-7
###
### _log_file (mandatory without _email_....)
###                                    - A filename to append the output to.
###                                      If this variable is defined, the
###                                      output is appended to the specified
###                                      file with a timestamp added.
### Example: _log_file                   bootflash:/my_file.log
###
### _email_server (mandatory without _log_file)
###                                    - A Simple Mail Transfer Protocol (SMTP)
###                                      mail server used to send e-mail.
### Example: _email_server               mailserver.example.com
###
### _email_from (mandatory without _log_file)
###                                    - The address from which e-mail is sent.
### Example: _email_from                 devtest@example.com
###
### _email_to (mandatory without _log_file)
###                                    - The address to which e-mail is sent.
### Example: _email_to                   engineering@example.com
###
### _email_cc (optional)               - The address to which the e-mail must
###                                      be copied.
### Example: _email_cc                   manager@example.com
###
### _show_cmd (mandatory)              - The CLI command to be executed when
###                                      the policy is run.
### Example: _show_cmd                   show version
###
# check if all required environment variables exist
# If any required environment variable does not exist, print out an error msg and quit
if {![info exists _log_file]} {
    if {![info exists _email_server]} {
 set result \
```

```
  "Policy cannot be run: variable _log_file or _email_server has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_from]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_from has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_to]} {
 set result \
  "Policy cannot be run: variable _log_file ore _email_to has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_cc]} {
 #_email_cc is an option, must set to empty string if not set.
 set _email_cc ""
    }
}
if {![info exists _show_cmd]} {
    set result \
        "Policy cannot be run: variable _show_cmd has not been set"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info and log a message
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)
# log a message
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
 $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 1. execute the command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# save exact execution time for command
set time_now [clock seconds]
# execute command
if [catch {cli_exec $cli1(fd) $_show_cmd} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
    # format output: remove trailing router prompt
    regexp {\n*(.*\n)([^\n]*)$} $result dummy cmd_output
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
```

```
    error $result $errorInfo
}

# 2. log the success of the CLI command
set msg [format "Command \"%s\" executed successfully" $_show_cmd]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 3. if _log_file is defined, then attach it to the file
if {[info exists _log_file]} {
    # attach output to file
    if [catch {open $_log_file a+} result] {
        error $result
    }
    set fileD $result
    # save timestamp of command execution
    #       (Format = 00:53:44 PDT Mon May 02 2005)
    set time_now [clock format $time_now -format "%T %Z %a %b %d %Y"]
    puts $fileD "%%% Timestamp = $time_now"
    puts $fileD $cmd_output
    close $fileD
}
# 4. if _email_server is defined send the email out
if {[info exists _email_server]} {
    set routername [info hostname]
    if {[string match "" $routername]} {
 error "Host name is not configured"
    }
    if [catch {smtp_subst [file join $tcl_library email_template_cmd.tm]} \
     result] {
 error $result $errorInfo
    }
    if [catch {smtp_send_email $result} result] {
 error $result $errorInfo
    }
}
```

sl_intf_down.tcl Sample Policy

The following sample policy runs when a configurable syslog message is logged. The policy executes a configurable CLI command and e-mails the results.

```
::cisco::eem::event_register_syslog occurs 1 pattern $_syslog_pattern maxrun 90

#-------------------------------------------------------------------
# EEM policy to monitor for a specified syslog message.
# Designed to be used for syslog interface-down messages.
# When event is triggered, the given config commands will be run.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#-------------------------------------------------------------------

### The following EEM environment variables are used:
###
### _syslog_pattern (mandatory)        - A regular expression pattern match string
###                                      that is used to compare syslog messages
###                                      to determine when policy runs
```

```
### Example: _syslog_pattern             .*UPDOWN.*FastEthernet0/0.*
###
### _email_server (mandatory)          - A Simple Mail Transfer Protocol (SMTP)
###                                      mail server used to send e-mail.
### Example: _email_server               mailserver.example.com
###
### _email_from (mandatory)            - The address from which e-mail is sent.
### Example: _email_from                 devtest@example.com
###
### _email_to (mandatory)              - The address to which e-mail is sent.
### Example: _email_to                   engineering@example.com
###
### _email_cc (optional)               - The address to which the e-mail must
###                                      be copied.
### Example: _email_cc                   manager@example.com
###
### _config_cmd1 (optional)            - The first configuration command that
###                                      is executed.
### Example: _config_cmd1                interface Ethernet1/0
###
### _config_cmd2 (optional)            - The second configuration command that
###                                      is executed.
### Example: _config_cmd2                no shutdown
###

# check if all the env variables we need exist
# If any of them doesn't exist, print out an error msg and quit
if {![info exists _email_server]} {
    set result \
        "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
        "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
        "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
}
if {![info exists _email_cc]} {
     #_email_cc is an option, must set to empty string if not set.
     set _email_cc ""
}

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

# 1. query the information of latest triggered eem event
array set arr_einfo [event_reqinfo]

if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

set msg $arr_einfo(msg)
set config_cmds ""

# 2. execute the user-defined config commands
if [catch {cli_open} result] {
```

```
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
if [catch {cli_exec $cli1(fd) "config t"} result] {
    error $result $errorInfo
}

if {[info exists _config_cmd1]} {
    if [catch {cli_exec $cli1(fd) $_config_cmd1} result] {
        error $result $errorInfo
    }
    append config_cmds $_config_cmd1
}

if {[info exists _config_cmd2]} {
    if [catch {cli_exec $cli1(fd) $_config_cmd2} result] {
        error $result $errorInfo
    }
    append config_cmds "\n"
    append config_cmds $_config_cmd2
}

if [catch {cli_exec $cli1(fd) "end"} result] {
    error $result $errorInfo
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}

after 60000
# 3. send the notification email
set routername [info hostname]
if {[string match "" $routername]} {
    error "Host name is not configured"
}

if [catch {smtp_subst [file join $tcl_library email_template_cfg.tm]} result] {
    error $result $errorInfo
}
if [catch {smtp_send_email $result} result] {
    error $result $errorInfo
}
```

The following e-mail template file is used with the EEM sample policy above:

```
email_template_cfg.tm
Mailservername: $_email_server
From: $_email_from
To: $_email_to
Cc: $_email_cc
Subject: From router $routername: Periodic $_show_cmd Output
$cmd_output
```

Debugging Embedded Event Manager Policies Examples

The following examples show how to debug the CLI library and the SMTP library.

Debugging the CLI Library

The CLI library allows users to run CLI commands and obtain the output of commands in Tcl. An Embedded Event Manager **debug** command has been provided for users of this library. The command to enable CLI library debugging is **debug event manager tcl cli_library**. When enabled, this command displays all data that is passed in and read back from the TTY session that handles the CLI interactions. This data helps ensure users that the commands that they are passing to the CLI are valid.

Example of the debug event manager tcl cli_library Command

This example uses the sample policy sl_intf_down.tcl. When triggered, sl_intf_down.tcl passes a configuration command to the CLI through the CLI library. The command passed in below is **show event manager environment**. This command is not a valid command in configuration mode. Without the **debug** command enabled, the output is shown below:

```
00:00:57:sl_intf_down.tcl[0]:config_cmds are show eve man env
00:00:57:%SYS-5-CONFIG_I:Configured from console by vty0
```

Notice that with the output above the user would not know whether or not the command succeeded in the CLI. With the **debug event manager tcl cli_library** command enabled, the user sees the following:

```
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : CTL : cli_open called.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson>
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson>enable
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson#configure terminal
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : Enter configuration commands, one
per line.  End with CNTL/Z.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson(config)#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson(config)#show event manager
environment
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT :                 ^
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : % Invalid input detected at '^'
marker.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson(config)#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson(config)#end
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : CTL : cli_close called.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson#exit
01:17:07: sl_intf_down.tcl[0]: config_cmds are show event manager environment
01:17:07: %SYS-5-CONFIG_I: Configured from console by vty0
```

The output above shows that **show event manager environment** is an invalid command in configuration mode. The IN keyword signifies all data passed in to the TTY through the CLI library. The OUT keyword signifies all data read back from the TTY through the CLI library. The CTL keyword signifies helper functions used in the CLI library. These helper functions are used to set up and remove connections to the CLI.

Debugging the SMTP Library

The SMTP library allows users to send e-mail messages to an SMTP e-mail server. An Embedded Event Manager **debug** command has been provided for users of this library. The command to enable SMTP library debugging is **debug event manager tcl smtp_library**. When enabled, this command displays all data that is passed in and read back from the SMTP library routines. This data helps ensure users that the commands that they are passing to the SMTP library are valid.

Example of the debug event manager tcl smtp_library Command

This example uses the sample policy tm_cli_cmd.tcl. When triggered, tm_cli_cmd.tcl runs the command **show event manager policy available system** through the CLI library. The result is then mailed to a user through the SMTP library. The output will help debug any issues related to using the SMTP library.

With the **debug event manager tcl smtp_library** command enabled, the users see the following on the console:

```
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 220 XXXX.example.com ESMTP XXXX
 1.1.0; Tue,
25 Jun 2002 14:20:39 -0700 (PDT)
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : HELO XXXX.example.com
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 XXXX.example.com Hello
XXXX.example.com [XXXX],
pleased to meet you
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : MAIL FROM:<XX@example.com>
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>... Sender
 ok
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : RCPT TO:<XX@example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>... Recipient
 ok
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : RCPT TO:<XX@example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>... Recipient
 ok
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : DATA
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 354 Enter mail, end with "."
on a line by itself
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Date: 25 Jun 2002 14:35:00 UTC

00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Message-ID:
<20020625143500.2387058729877@XXXX.example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : From: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : To: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Cc: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Subject: From router nelson:
Periodic show eve man po ava system Output
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : No. Type    Time Created
       Name
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 1   system  Fri May3   20:42:34
 2002  pr_cdp_abort.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 2   system  Fri May3   20:42:54
 2002  pr_iprouting_abort.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 3   system  Wed Apr3   02:16:33
 2002  sl_intf_down.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 4   system  Mon Jun24  23:34:16
 2002  tm_cli_cmd.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 5   system  Wed Mar27  05:53:15
 2002  tm_crash_hist.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : nelson#
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write :
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : .
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 ADE90179 Message accepted
for delivery
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : QUIT
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 221 XXXX.example.com closing
connection
```

Tracing Tcl set Command Operations Example

Tcl is a flexible language. One of the flexible aspects of Tcl is that you can override commands. In this example, the Tcl **set** command is renamed as _set and a new version of the **set** command is created that displays a message containing the text "setting" and appends the scalar variable that is being set. This example can be used to trace all instances of scalar variables being set.

```
rename set _set
proc set {var args} {
   puts [list setting $var $args]
   uplevel _set $var $args
};
```

When this is placed in a policy, a message is displayed anytime a scalar variable is set, for example:

```
02:17:58: sl_intf_down.tcl[0]: setting test_var 1
```

RPC Event Detector Example

```
TCL script (rpccli.tcl):
::cisco::eem::event_register_rpc
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
proc run_cli { clist } {
    set rbuf ""
    if {[llength $clist] < 1} {
    return -code ok $rbuf
    }
    if {[catch {cli_open} result]} {
        return -code error $result
    } else {
    array set cliarr $result
    }
    if {[catch {cli_exec $cliarr(fd) "enable"} result]} {
        return -code error $result
    }
    if {[catch {cli_exec $cliarr(fd) "term length 0"} result]} {
        return -code error $result
    }
    foreach cmd $clist {
    if {[catch {cli_exec $cliarr(fd) $cmd} result]} {
            return -code error $result
    }
    append rbuf $result
    }
    if {[catch {cli_close $cliarr(fd) $cliarr(tty_id)} result]} {
        puts "WARNING: $result"
    }
    return -code ok $rbuf
}
proc run_cli_interactive { clist } {
    set rbuf ""
    if {[llength $clist] < 1} {
    return -code ok $rbuf
    }
    if {[catch {cli_open} result]} {
        return -code error $result
    } else {
    array set cliarr $result
```

```
    }
    if {[catch {cli_exec $cliarr(fd) "enable"} result]} {
        return -code error $result
    }
    if {[catch {cli_exec $cliarr(fd) "term length 0"} result]} {
        return -code error $result
    }
    foreach cmd $clist {
        array set sendexp $cmd
    if {[catch {cli_write $cliarr(fd) $sendexp(send)} result]} {
            return -code error $result
    }
    foreach response $sendexp(responses) {
        array set resp $response
        if {[catch {cli_read_pattern $cliarr(fd) $resp(expect)} result]} {
                return -code error $result
        }
        if {[catch {cli_write $cliarr(fd) $resp(reply)} result]} {
                return -code error $result
        }
    }
    if {[catch {cli_read $cliarr(fd)} result]} {
            return -code error $result
    }
    append rbuf $result
    }
    if {[catch {cli_close $cliarr(fd) $cliarr(tty_id)} result]} {
        puts "WARNING: $result"
    }
    return -code ok $rbuf
}
array set arr_einfo [event_reqinfo]
set args $arr_einfo(argc)
set cmds [list]
for { set i 0 } { $i < $args } { incr i } {
    set arg "arg${i}"
    # Split each argument on the '^' character.  The first element is
    # the command, and each subsequent element is a prompt followed by
    # a response to that prompt.
    set cmdlist [split $arr_einfo($arg) "^"]
    set cmdarr(send) [lindex $cmdlist 0]
    set cmdarr(responses) [list]
    if { [expr ([llength $cmdlist] - 1) % 2] != 0 } {
    return -code 88
    }
    set cmdarr(responses) [list]
    for { set j 1 } { $j < [llength $cmdlist] } { incr j 2 } {
    set resps(expect) [lindex $cmdlist $j]
    set resps(reply) [lindex $cmdlist [expr $j + 1]]
    lappend cmdarr(responses) [array get resps]
    }
    lappend cmds [array get cmdarr]
}
set rc [catch {run_cli_interactive $cmds} output]
if { $rc != 0 } {
    error $output $errorInfo
    return -code 88
}
puts $output
```

Additional References

The following sections provide references related to writing Embedded Event Manager policies using Tcl.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
EEM commands: complete command syntax, defaults, command mode, command history, usage guidelines, and examples	Cisco IOS Embedded Event Manager Command Reference
Embedded Event Manager overview	Embedded Event Manager Overview module.
Embedded Event Manager policy writing using the CLI	Writing Embedded Event Manager Policies Using the Cisco IOS CLI module
Embedded Resource Manager	Embedded Resource Manager module

MIBs

MIB	MIBs Link
CISCO-EMBEDDED-EVENT-MGR-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Writing Embedded Event Manager 4.0 Policies Using Tcl

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

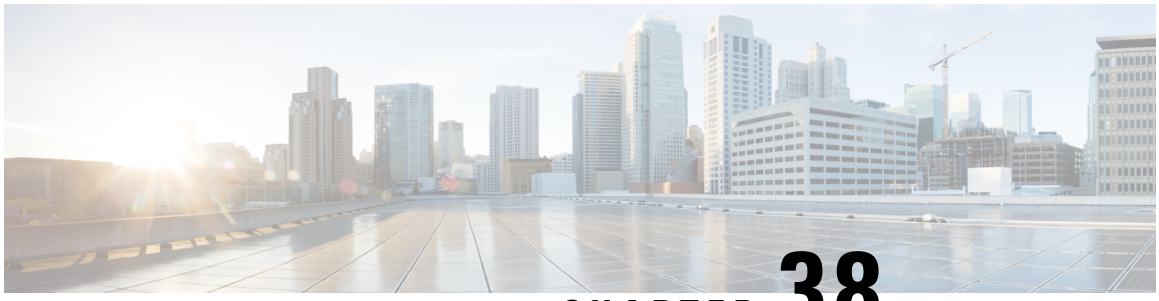
Table 73: Feature Information for Writing Embedded Event Manager 4.0 Policies Using Tcl

Feature Name	Releases	Feature Information
Embedded Event Manager 1.0	12.0(26)S 12.3(4)T	EEM 1.0 introduced Embedded Event Manager applet creation with the SNMP and syslog event detectors. EEM 1.0 also introduced the following actions: generating prioritized syslog messages, generating a CNS event for upstream processing by Cisco CNS devices, reloading the Cisco software, and switching to a secondary processor in a fully redundant hardware configuration. The following commands were introduced by this feature: **action cns-event**, **action force-switchover**, **action reload**, **action syslog**, **debug event manager**, **event manager applet**, **event snmp**, **event syslog**, **show event manager policy registered**.
Embedded Event Manager 2.0	12.2(25)S	EEM 2.0 introduced the application-specific event detector, the counter event detector, the interface counter event detector, the timer event detector, and the watchdog event detector. New actions included modifying a named counter, publishing an application-specific event, and generating an SNMP trap. The ability to define environment variables and to run EEM policies written using Tcl was introduced, and two sample policies were included with the software. The following commands were introduced by this feature: **action counter**, **action publish-event**, **action snmp-trap**, **event application**, **event counter**, **event interface**, **event ioswdsysmon**, **event manager environment**, **event manager history size**, **event manager policy**, **event manager scheduler suspend**, **event timer**, **show event manager environment**, **show event manager history events**, **show event manager history traps**, **show event manager policy available**, **show event manager policy pending**.

Feature Name	Releases	Feature Information
Embedded Event Manager 2.1	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	EEM 2.1 introduced some new event detectors and actions with new functionality to allow EEM policies to be run manually and the ability to run multiple concurrent policies. Support for Simple Network Management Protocol (SNMP) event detector rate-based events was provided as was the ability to create policies using Tool Command Language (Tcl). The following commands were introduced or modified by this feature: **action cli**, **action counter**, **action info**, **action mail**, **action policy**, **debug event manager**, **event cli**, **event manager directory user**, **event manager policy**, **event manager run**, **event manager scheduler script**, **event manager session cli username**, **event none**, **event oir**, **event snmp**, **event syslog**, **set**(EEM), **show event manager directory user**, **show event manager policy registered**, **show event manager session cli username**.
Embedded Event Manager 2.1 (Software Modularity)	12.2(18)SXF4 Cisco IOS Software Modularity images	EEM 2.1 for Software Modularity images introduced the GOLD, system manager, and WDSysMon (Cisco IOS Software Modularity watchdog) event detectors, and the ability to display Cisco IOS Software Modularity processes and process metrics. The following commands were introduced by this feature: **event gold**, **event process**, **show event manager metric process**. **Note** EEM 2.1 for Software Modularity images also supports the resource and RF event detectors introduced in EEM 2.2, but it does not support the enhanced object tracking event detector or the actions to read and set tracked objects.
Embedded Event Manager 2.2	12.4(2)T 12.2(31)SB3 12.2(33)SRB	EEM 2.2 introduced the enhanced object tracking, resource, and RF event detectors. The actions of reading and setting the state of a tracked object were also introduced. The following commands were introduced or modified by this feature: **action track read**, **action track set**, **default-state**, **event resource**, **event rf**, **event track**, **show track**, **track stub-object**.
SNMP event detector delta environment variable	12.4(11)T	A new SNMP event detector environment variable, _snmp_oid_delta_val, was introduced. This is a minor enhancement. Minor enhancements are not typically listed in Feature Navigator.

Feature Name	Releases	Feature Information
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB 15.1(2)SY	EEM 2.3 introduced some new features relative to the Generic Online Diagnostics (GOLD) Event Detector on the Cisco Catalyst 6500 Series switches. The **event gold** command was enhanced in addition to the Tcl keywords--**action-notify**, **testing-type**, **test-name**, **test-id**, **consecutive-failure**, **platform-action**, and **maxrun**--for improved reaction to GOLD test failures and conditions Read-only variables were added under the **GOLD Event Detector** category to provide access to platform-wide and test-specific GOLD event detector information for a detected event.
Embedded Event Manager 2.4	12.4(20)T 12.2(33)SXI 12.2(33)SRE 15.1(2)SY	EEM 2.4 introduced several new features. The following commands were introduced by this feature: **attribute (EEM)** , **correlate**, **event manager detector rpc**, **event manager directory user repository**, **event manager update user policy**, **event manager scheduler clear**, **event manager update user policy**, **event owner**, **event rpc**, **event snmp-notification**, **show event manager detector**, **show event manager version**, **trigger (EEM)**.
Embedded Event Manger 3.0	12.4(22)T 12.2(33)SRE 12.2(50)SY	EEM 3.0 introduced several new features. The following commands were introduced or modified by this feature: **action add** , **action append**, **action break**, **action comment**, **action context retrieve**, **action context save**, **action continue**, **action decrement**, **action divide**, **action else**, **action elseif**, **action end**, **action exit**, **action foreach**, **action gets**, **action if**, **action if goto**, **action increment**, **action info type interface-names**, **action info type snmp getid**, **action info type snmp inform**, **action info type snmp oid**, **action info type snmp trap**, **action info type snmp var**, **action multiply**, **action puts**, **action regexp**, **action set (EEM)**, **action string compare**, **action string equal**, **action string first**, **action string index**, **action string last**, **action string length**, **action string match**, **action string range**, **action string replace**, **action string tolower**, **action string toupper**, **action string trim**, **action string trimleft**, **action string trimright**, **action subtract**, **action while**, **event cli**, **event ipsla**, **event manager detector routing**, **event manager scheduler**, **event manager scheduler clear**, **event manager scheduler hold**, **event manager scheduler modify**, **event manager scheduler release**, **event nf**, **event routing**, **show event manager policy active**, **show event manager policy pending**, and **show event manager scheduler**.
Embedded Event Manager 3.1	15.0(1)M 15.1(1)SY 15.1(2)SY	EEM 3.1 introduced several new features. The following commands were introduced or modified by this feature: **action syslog**, **description (EEM)** , **event manager applet** , **event manager policy** , **event snmp-notification** , **event snmp-object** , **show event manager policy registered** , and **show event manager policy available** .

Feature Name	Releases	Feature Information
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG 15.1(3)T 15.1(1)SY 15.1(2)SY	EEM is a distributed and customized approach to event detection and recovery offered directly in a Cisco IOS device. The following sections provide information about this feature: The following commands were introduced or modified: **debug event manager**, **event identity**, **event mat**, **event neighbor-discovery**, **show event manager detector**.
Embedded Event Manager 4.0	15.2(2)T 15.1(1)SY 15.1(2)SY 12.2(2)E	EEM 4.0 introduced several new features. The following commands were introduced or modified: **action file**, **action mail**, **action syslog**, **clear event manager detector counters**, **clear event manager server counters**, **event cli**, **event manager policy**, **event manager scheduler**, **event syslog**, **show event manager detector**, **show event manager policy registered**, **show event manager statistics**.



CHAPTER 38

Signed Tcl Scripts

The Signed Tcl Scripts feature allows you to create a certificate to generate a digital signature and sign a Tool Command Language (Tcl) script with that digital signature. This feature also allows you to work with existing scripts and certificates. The digital signature is verified for authentication and then run with trusted access to the Tcl interpreter. If the script does not contain the digital signature, the script may run in a limited mode for untrusted scripts, or may not run at all.

Prerequisites for Signed Tcl Scripts

For this feature to work, the Cisco public key infrastructure (PKI) configuration trustpoint commands must be enabled.

Restrictions for Signed Tcl Scripts

For this feature to work, you must be running the following:

- Cisco IOS Crypto image
- OpenSSL Version 0.9.7a or above
- Expect

Information About Signed Tcl Scripts

The Signed Tcl Scripts feature introduces security for the Tcl scripts. This feature allows you to create a certificate to generate a digital signature and sign a Tcl script with that digital signature. This certificate examines the Tcl scripts prior to running them. The script is checked for a digital signature from Cisco. In addition, third parties may also sign a script with a digital signature. You may wish to sign your own internally developed Tcl scripts or you could use a script developed by a third party. If the script contains the correct digital signature, it is believed to be authentic and runs with full access to the Tcl interpreter. If the script does not contain the digital signature, the script may be run in a limited mode, known as Safe Tcl mode, or may not run at all.

To create and use signed Tcl scripts, you should understand the following concepts:

Cisco PKI

Cisco PKI provides certificate management to support security protocols such as IP security (IPsec), secure shell (SSH), and secure socket layer (SSL). A PKI is composed of the following entities:

- Peers communicating on a secure network
- At least one certification authority (CA) that grants and maintains certificates
- Digital certificates, which contain information such as the certificate validity period, peer identity information, encryption keys that are used for secure communication, and the signature of the issuing CA
- An optional registration authority (RA) to offload the CA by processing enrollment requests
- A distribution mechanism (such as Lightweight Directory Access Protocol [LDAP] or HTTP) for certificate revocation lists (CRLs)

PKI provides you with a scalable, secure mechanism for distributing, managing, and revoking encryption and identity information in a secured data network. Every routing device participating in the secured communication is enrolled in the PKI in a process where the routing device generates a Rivest, Shamir, and Adelman (RSA) key pair (one private key and one public key) and has its identity validated by a trusted routing device (also known as a CA or trustpoint).

After each routing device enrolls in a PKI, every peer (also known as an end host) in a PKI is granted a digital certificate that has been issued by a CA. When peers must negotiate a secured communication session, they exchange digital certificates. Based on the information in the certificate, a peer can validate the identity of another peer and establish an encrypted session with the public keys contained in the certificate.

RSA Key Pair

An RSA key pair consists of a public key and a private key. When setting up your PKI, you must include the public key in the certificate enrollment request. After the certificate has been granted, the public key is included in the certificate so that peers can use it to encrypt data that is sent to the device. The private key is kept on the device and used both to decrypt the data sent by peers and to digitally sign transactions when negotiating with peers.

RSA key pairs contain a key modulus value. The modulus determines the size of the RSA key. The larger the modulus, the more secure the RSA key. However, keys with large modulus values take longer to generate, and encryption and decryption operations take longer with larger keys.

Certificate and Trustpoint

A certification authority (CA), also known as a trustpoint, manages certificate requests and issues certificates to participating network devices. These services (managing certificate requests and issuing certificates) provide centralized key management for the participating devices and are explicitly trusted by the receiver to validate identities and to create digital certificates. Before any PKI operations can begin, the CA generates its own public key pair and creates a self-signed CA certificate; thereafter, the CA can sign certificate requests and begin peer enrollment for the PKI.

You can use a CA provided by a third-party CA vendor, or you can use an internal CA, which is the Cisco Certificate Server.

How to Configure Signed Tcl Scripts

Generating a Key Pair

The key pair consists of a private key and a public key. The private key is intended to be kept private, accessible only to the creator. The public key is generated from the private key and is intended to be known to the public.

To generate a key pair, use the **openssl genrsa** command and then the **openssl rsa** command.

SUMMARY STEPS

1. **openssl genrsa -out** *private-key-file bit-length*
2. **ls -l**
3. **openssl rsa -in** *private-key-file* **-pubout -out** *public-key-file*
4. **ls -l**

DETAILED STEPS

Step 1 **openssl genrsa -out** *private-key-file bit-length*

This command generates a private key that is *bit-length* bits long and writes the key to the *private-key-file*file.

```
Host% openssl genrsa -out privkey.pem 2048
```

Example:

```
Generating RSA private key, 2048 bit long modulus
.........+++
.....................................................................................+++
e is 65537 (0x10001)
```

Step 2 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 8
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
```

The privkey.pem file contains the private key generated using the **openssl genrsa** command.

Step 3 **openssl rsa -in** *private-key-file* **-pubout -out** *public-key-file*

This command generates a public key based on the specified private key in the *private-key-file* file and writes the public key to the *public-key-file*file.

Example:

```
Host% openssl rsa -in privkey.pem -pubout -out pubkey.pem

writing RSA key
```

Step 4 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 16
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

The pubkey.pem file contains the public key generated from the private key using the **openssl rsa** command.

Generating a Certificate

Perform this task to generate a certificate. To generate an X.509 certificate, use the **openssl req** command.

SUMMARY STEPS

1. **openssl req -new -x509 -key** *private-key-file* **-out** *certificate-file* **-days** *expiration-days*
2. **ls -l**

DETAILED STEPS

Step 1 **openssl req -new -x509 -key** *private-key-file* **-out** *certificate-file* **-days** *expiration-days*

This command creates an X.509 certificate, with full access to a private key that is stored in the *private-key-file* file, and stores the certificate in the *certificate-file*file. The certificate is configured to expire in *expiration-days* days.

To complete the command, enter the following Distinguished Name (DN) information when prompted:

- Country name
- State or province name
- Organization name
- Organizational unit name
- Common name
- Email address

At each prompt, text enclosed in square brackets indicates the default value that will be used if you do not enter a value before you press Enter.

This example shows how to create an X.509 certificate that has full access to the private key in the privkey.pem file. The certificate is written to the cert.pem file and will expire 1095 days after the creation date.

Example:

```
Host% openssl req -new -x509 -key privkey.pem -out cert.pem -days 1095

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value, If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:US

State or Province Name (full name) [Berkshire]:California

Locality Name (eg, city) [Newbury]:San Jose

Organization Name (eg, company) [My Company Ltd]:Cisco Systems, Inc.

Organizational Unit Name (eg, section) []:DEPT_ACCT

Common Name (eg, your name or your server's hostname) []:Jane

Email Address []:janedoe@company.com
```

Step 2 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 24
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

The cert.pem file contains the X.509 certificate created using the **openssl req** command.

Signing the Tcl Scripts

Perform this task to sign the Tcl scripts. You will need to sign the Tcl file and output in OpenSSL document in pkcs7 (PKCS#7) format.

To sign the Tcl file, use the **openssl smime** command with the **-sign** keyword.

SUMMARY STEPS

1. **openssl smime -sign -in** *tcl-file* **-out** *signed-tcl-file* **-signer** *certificate-file* **-inkey** *private-key-file* **-outform DER -binary**
2. **ls -l**

DETAILED STEPS

Step 1 **openssl smime -sign -in** *tcl-file* **-out** *signed-tcl-file* **-signer** *certificate-file* **-inkey** *private-key-file* **-outform DER -binary**

This command signs the Tcl filename *tcl-file* using the certificate stored in *certificate-file* and the private key stored in *private-key-file* file and then writes the signed Tcl file in DER PKCS#7 format to the *signed-tcl-file*file.

Example:

```
Host% openssl smime -sign -in hello -out hello.pk7 -signer cert.pem -inkey privkey.pem -outform DER
 -binary
```

Step 2 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

The hello.pk7 file contains the signed Tcl file created by the **openssl smime** command from the unsigned Tcl file named hello and using the X.509 certificate in the cert.pem file.

Verifying the Signature

Perform this task to verify that the signature matches the data, use the **openssl smime** command with the **-verify** keyword. The original Tcl content must be provided in the input file, because the file does not have the original content.

SUMMARY STEPS

1. **openssl smime -verify -in** *signed-tcl-file* **-CAfile** *certificate-file* **-inform DER -content** *tcl-file*
2. **ls -l**

DETAILED STEPS

Step 1 **openssl smime -verify -in** *signed-tcl-file* **-CAfile** *certificate-file* **-inform DER -content** *tcl-file*

This command verifies the signed Tcl file stored in DER PKCS#7 format in *signed-tcl-file* using the trusted Certificate Authority (CA) certificates in *certificate-file* and then writes the detached content to the file *tcl-file*.

The following example shows how to verify the signature with the input file hello.pk7:

Example:

```
Host% openssl smime -verify -in hello.pk7 -CAfile cert.pem -inform DER -content hello

puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
Verification successful
```

Note The SSL command page describes **-in** *filename* as the input message to be encrypted or signed or the MIME message to be decrypted or verified. For more information, go to http://www.openssl.org/ .

Step 2 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 13 10:18 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:17 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

The hello file contains the content detached from the signed Tcl file hello.pk7 by running the **openssl smime** command with the **-verify** keyword. If the verification was successful, the signer's certificates are written to the X.509 certificate in the cert.pem file.

Converting the Signature into Nonbinary Data

Perform this task to convert the signature from binary to nonbinary data.

SUMMARY STEPS

1. **xxd -ps** *signed-tcl-file* > *nonbinary-signature-file*
2. Create a script that displays **#Cisco Tcl Signature V1.0** in the first line andinserts a comment character (#) at the beginning of each line of the input file and writes each line to a file whose name is formed by appending the text string "_sig" to the name of the input file.
3. Run the script, supplying the name of the file containing the nonbinary signature file (*nonbinary-signature-file*) as the input argument.
4. **ls -l**
5. **cat** *signed-tcl-file* *commented-nonbinary-signature-file* > *signed-tcl-script*
6. **cat** *signed-tcl-script*

DETAILED STEPS

Step 1 **xxd -ps** *signed-tcl-file* > *nonbinary-signature-file*

This command converts the signature in *signed-tcl-file* from binary to nonbinary data and stores it as a hexadecimal dump in the file *nonbinary-signature-file*.

Example:

```
Host% xxd -ps hello.pk7 > hello.hex
```

Step 2 Create a script that displays **#Cisco Tcl Signature V1.0** in the first line andinserts a comment character (#) at the beginning of each line of the input file and writes each line to a file whose name is formed by appending the text string "_sig" to the name of the input file.

In this example the **cat** command is used to display the contents of the script file named my_append.

Example:

```
Host% cat my_append

#!/usr/bin/env expect
set my_first {#Cisco Tcl Signature V1.0}
set newline {}
set my_file [lindex $argv 0]
set my_new_file ${my_file}_sig
set my_new_handle [open $my_new_file w]
set my_handle [open $my_file r]
puts $my_new_handle $newline
puts $my_new_handle $my_first
foreach line [split [read $my_handle] "\n"]  {
   set new_line {#}
   append new_line $line
   puts $my_new_handle $new_line
}

close $my_new_handle
close $my_handle
```

Step 3 Run the script, supplying the name of the file containing the nonbinary signature file (*nonbinary-signature-file*) as the input argument.

In this example, the my_append script is run with the nonbinary signature file hello.hex specified as input. The output file will be named hello.hex_sig.

Example:

```
Host% my_append hello.hex
```

Step 4 **ls -l**

This command displays detailed information about each file in the current directory, including the permissions, owners, size, and when last modified.

Example:

```
Host% ls -l

total 80
-rw-r--r--   1 janedoe eng12       1659 Jun 13 10:18 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:17 hello
-rw-r--r--   1 janedoe eng12       3815 Jun 13 10:20 hello.hex
-rw-r--r--   1 janedoe eng12       3907 Jun 13 10:22 hello.hex_sig
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rwxr--r--   1 janedoe eng12        444 Jun 13 10:22 my_append
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

The hello.hex file contains nonbinary data (stored as a hexadecimal dump) converted from the binary signature in the signed Tcl file hello.pk7. The my_append file contains the script that inserts a comment character at the beginning of each line of the input file. The hello.hex_sig file is the file created by running the my_append script on the nonbinary signature file.

Step 5 **cat** *signed-tcl-file commented-nonbinary-signature-file* **>** *signed-tcl-script*

This command appends the contents of the nonbinary signature file (*commented-nonbinary-signature-file*) to the signed Tcl file stored in DER PKCS#7 format (in the *signed-tcl-file* file). The concatenated output is written to the file *signed-tcl-script*.

Example:

```
Host% cat hello hello.hex_sig > hello.tcl
```

Step 6 **cat** *signed-tcl-script*

This command displays the contents of the file *signed-tcl-script*, which is the concatenation of content detached from the signed Tcl file and the nonbinary signature file.

Example:

```
Host% cat hello.tcl

puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
#Cisco Tcl Signature V1.0
#3082075006092a864886f70d010702a08207413082073d020101310b3009
#06052b0e03021a0500300b06092a864886f70d010701a08204a13082049d
#30820385a003020102020100300d06092a864886f70d0101040500308195
#310b300906035504061302555331133011060355040813a43616c69666f
#726e69613111300f060355040713085616e204a6f7365311c301a060355
#040a1313436973636f2053797374656d732c20496e632e310e300c060355
#040b13054e53535447310d300b060355040313044a6f686e3121301f0609
#2a864886f70d01090116126a6c6175746d616e40636973636f2e636f6d30
```

```
#1e170d3037303631323232303134335a170d3130303631313232303133
#5a308195310b3009060355040613025553311330110603550408130a4361
#6c69666f726e69613111300f060355040713085361616e204a6f7365311c30
#1a060355040a1313436973636f2053797374656d732c20496e632e310e30
#0c060355040b13054e53535447310d300b060355040313044a6f686e3121
#301f06092a864886f70d01090116126a6c6175746d616e40636973636f2e
#636f6d30820122300d06092a864886f70d01010105000382010f00308201
#0a0282010100a751eb5ec1f3009738c88a55987c07b759c36f3386342283
#67ea20a89d9483ae85e0c63eeded8ab3eb7a08006689f09136f172183665
#c971099ba54e77ab47706069bbefaaab8c50184396350e4cc870c4c3f477
#88c55c52e2cf411f05b59f0eaec0678ff5cc238fdce2263a9fc6b6c244b8
#ffaead865c19c3d3172674a13b24c8f2c01dd8b1bd491c13e84e29171b85
#f28155d81ac8c69bb25ca23c2921d85fbf745c106e7aff93c72316cbc654
#4a34ea88174a8ba7777fa60662974e1fbac85a0f0aeac925dba6e5e850b8
#7caffce2fe8bb04b61b62f532b5893c081522d538005df81670b931b0ad0
#e1e76ae648f598a9442d5d0976e67c8d55889299147d0203010001a381f5
#3081f2301d0603551d0e04160414bc34132be952ff8b9e1af3b93140a255
#e54a667c3081c20603551d230481ba3081b78014bc34132be952ff8b9e1a
#f3b93140a255e54a667ca1819ba48198308195310b300906035504061302
#5553311330110603550408130a43616c69666f726e69613111300f060355
#0407130853616e204a6f7365311c301a060355040a1313436973636f2053
#797374656d732c20496e632e310e300c060355040b13054e53535447310d
#300b060355040313044a6f686e3121301f06092a864886f70d0109011612
#6a6c6175746d616e40636973636f2e636f6d820100300c0603551d130405
#30030101ff300d06092a864886f70d01010405000382010100c83c1b074
#6720929c9514af6d5df96f0a95639f047c40a607c83d8362507c58fa7f84
#aa699ec5e5bef61b2308297a0662c653ff446acfbb6f5cb2dd162d939338
#a5e4d78a5c45021e5d4dbabb8784efbf50cab0f5125d164487b31f5cf933
#a9f68f82cd111cbab1739d7f372ec460a7946882874b0a0f22dd53acbd62
#a944a15e52e54a24341b3b8a820f23a5bc7ea7b2278bb56838b8a4051926
#af9c167274ff8449003a4e012bcf4f4b3e280f85209249a390d14df47435
#35efabce720ea3d56803a84a2163db4478ae19d7d987ef6971c8312e280a
#aac0217d4fe620c6582a48faa8ea5e3726a99012e1d55f8d61b066381f77
#4158d144a43fb536c77d6a3182027730820273020101301819b308195310b
#300906035504061302555331133011060355040813a43616c69666f726e
#69613111300f060355040713085361616e204a6f7365311c301a060355040a
#1313436973636f2053797374656d732c20496e632e310e300c060355040b
#13054e53535447310d300b060355040313044a6f686e3121301f06092a86
#4886f70d010901161261a6c6175746d616e40636973636f2e636f6d020100
#300906052b0e03021a0500a081b1301806092a864886f70d010903310b06
#092a864886f70d010701301c06092a864886f70d010905310f170d303730
#363133313731363438 5a302306092a864886f70d01090431160414372cb3
#72dc607990577fd0426104a42ee4158d2b305206092a864886f70d01090f
#31453043300a06082a864886f70d0307300e06082a864886f70d03020202
#0080300d06082a864886f70d0302020140300706052b0e030207300d0608
#2a864886f70d0302020128300d06092a864886f70d010101050004820100
#72db6898742f449b26d3ac18f43a1e7178834fb05ad13951bf042e127eea
#944b72b96f3b8ecf7eb52f3d0e383bf63651750223efe69eae04287c9dae
#b1f31209444108b31d34e46654c6c3cc10b5baba887825c224ec6f376d49
#00ff7ab2d9f88402dab9a2c2ab6aa3ecceeaf5a594bdc7d3a822c55e7daa
#aa0c2b067e06967f22a20e406fe21d9013ecc6bd9cd6d402c2749f8bea61
#9f8f87acfbc9e10d6ce91502e34629adca6ee855419afafe6a8233333e14
#ad4c107901d1f2bca4d7ffaadddbc54192a25da662f8b8509782c76977b8
#94879453fbb00486ccc55f88db50fcc149bae066916b350089cde51a6483
#2ec14019611720fc5bbe2400f24225fc
```

Configuring the Device with a Certificate

Perform this task to configure the device with a certificate.

Before you begin

You must already have a Cisco IOS Crypto image; otherwise you cannot configure a certificate.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal**
5. **exit**
6. **crypto pki authenticate** *name*
7. At the prompt, enter the base-encoded CA certificate.
8. **scripting tcl secure-mode**
9. **scripting tcl trustpoint name** *name*
10. **scripting tcl trustpoint untrusted** {**execute** | **safe-execute** | **terminate**}
11. **exit**
12. **tclsafe**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **configure terminal**

Enters global configuration mode.

Example:

```
Device# configure terminal
```

Step 3 **crypto pki trustpoint** *name*

Declares the device is to use the Certificate Authority (CA) *mytrust* and enters ca-trustpoint configuration mode.

Example:

```
Device(config)# crypto pki trustpoint mytrust
```

Step 4 **enrollment terminal**

Specifies manual cut-and-paste certificate enrollment. When this command is enabled, the device displays the certificate request on the console terminal, allowing you to enter the issued certificate on the terminal.

Example:

```
Device(ca-trustpoint)# enrollment terminal
```

Step 5 **exit**

Exits ca-trustpoint configuration mode and returns to global configuration mode.

Example:

```
Device(ca-trustpoint)# exit
```

Step 6 **crypto pki authenticate** *name*

Retrieves the CA certificate and authenticates it. Check the certificate fingerprint if prompted.

Note Because the CA signs its own certificate, you should manually authenticate the public key of the CA by contacting the CA administrator when you perform this command.

Example:

```
Device(config)# crypto pki authenticate mytrust
```

Step 7 At the prompt, enter the base-encoded CA certificate.

Example:

```
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIEuDCCA6CgAwIBAgIBADANBgkqhkiG9w0BAQQFADCBnjELMAkGA1UEBhMCVVMx
EzARBgNVBAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMRwwGgYDVQQK
ExNDaXNjbyBTeXN0ZW1zLCBJbmMuMQ4wDAYDVQQLEwVOU1NURzEWMBQGA1UEAxMN
Sm9obiBMYXV0bWFubjEhMB8GCSqGSIb3DQEJARYSamxhdXRtYW5AY2lzY28uY29t
MB4XDTA2MTExNzE3NTgwMVoXDTA5MTExNjE3NTgwMVowgZ4xCzAJBgNVBAYTAlVT
MRMwEQYDVQQIEwpDYWxpZm9ybmlhMREwDwYDVQQHEwhTYW4gSm9zZTEcMBoGA1UE
ChMTQ2lzY28gU3lzdGVtcywgSW5jLjEOMAwGA1UECxMFTlNTVEcxFjAUBgNVBAMT
DUpvaG4gTGF1dG1hbm4xITAfBgkqhkiG9w0BCQEWEmpsYXV0bWFuQGNpc2NvLmNv
bTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALxtqTMCirMb+CdyWLuH
oWAM8CEJDwQggL7MWBhoi3TSMd/ww2XBB9biBtdlH6jHsjCiOwAR5OorakwfPyf7
mvRJ2PqJALs+Vn93VBKIG6rZUl4+wdOx686BVddIZvEJQPbROiYTzfazWV70aLMV
bd7/B7vF1SG1YK9y1tX9p9nZyZ0x47OAXetwOaGinvlG7VNuTXaASBLUjCRZsIlz
SBrXXedBzZ6+BuoWm1FK45EYSlag5Rt9RGXXMBqzx91iyhrJ3zDDmkExa45yKJET
mAgDVMcpeteJtif47UDZJK30g4MbMyx/c8WGhmJ54qRL9BZEPmDxMQkNP10l8MAl
Q8sCAwEAAaOB/jCB+zAdBgNVHQ4EFgQU9/ToDvbMR3JfJ4xEa4X47oNFq5kwgcsG
A1UdIwSBwzCBwIAU9/ToDvbMR3JfJ4xEa4X47oNFq5mhgaSkgaEwgZ4xCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMREwDwYDVQQHEwhTYW4gSm9zZTEc
MBoGA1UEChMTQ2lzY28gU3lzdGVtcywgSW5jLjEOMAwGA1UECxMFTlNTVEcxFjAU
BgNVBAMTDUpvaG4gTGF1dG1hbm4xITAfBgkqhkiG9w0BCQEWEmpsYXV0bWFuQGNp
c2NvLmNvbYIBADAMBgNVHRMEBTADAQH/MA0GCSqGSIb3DQEBBAUAA4IBAQBtEs/4
MQeN9pT+XPCPg2ObQU8y2AadI+I34YK+fDHsFOh68hZhpszTN2VpNEvkFXpADhgr
7DkNGtwTCla481v70iNFViQVL+inNrZwWMxoTnUNCK7Hc5kHkXt6cj0mvsefVUzx
Xl70mauhESRVlmYWrJxSsrEILerZYsuv5HbFdand+/rErmP2HVyfdntLnKdSzmXJ
5lwE/Et2QtYNGor0OBlLesowfslR3LhHi4wn+5is7mALgNw/NuTiUr1zH18OeB4m
wcpBIJsLaJu6ZUJQl7IqdswSa3fHd5qq0/k8P9z0YAYrf3+MFQr4ibvsYvHlO087
o2Js1gW4qz34pqNh
Certificate has the following attributes:
       Fingerprint MD5: 1E327DBB 330936EB 2FB8EACB 4FD1133E
      Fingerprint SHA1: EE7FF9F4 05148842 B9D50FAC D76FDC9C E0703246
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Step 8 **scripting tcl secure-mode**

Enables signature verification of the interactive Tcl scripts.

```
Device(config)# scripting tcl secure-mode
```

Step 9 **scripting tcl trustpoint name** *name*

Associates an existing configured trustpoint name with a certificate to verify Tcl scripts.

```
Device(config)# scripting tcl trustpoint name mytrust
```

Step 10 **scripting tcl trustpoint untrusted** {**execute** | **safe-execute** | **terminate**}

(Optional) Allows the interactive Tcl scripts to run regardless of the scripts failing in the signature check or in untrusted mode using one of the three keywords: **execute**, **safe-execute**, or **terminate**.

- **execute** --Executes Tcl scripts even if the signature verification fails. If the **execute** keyword is configured, signature verification is not at all performed.

Note Use of this keyword is usually not recommended because the signature verification is not at all performed.

The **execut**e keyword is provided for internal testing purposes and to provide flexibility. For example, in a situation where a certificate has expired but the other configurations are valid and you want to work with the existing configuration, then you can use the execute keyword to work around the expired certificate.

- **safe-execute** --Allows the script to run in safe mode. You can use the tclsafe command and also enter the interactive Tcl shell safe mode to explore the safe mode Tcl commands that are available. In order to get a better understanding of what is available in this limited safe mode, use the tclsafe Exec command to explore the options.
- **terminate** --Stops any script from running and reverts to default behavior. The default policy is to terminate. When the last trustpoint name is removed, the untrusted action is also removed. The untrusted action cannot be entered until at least one trustpoint name is configured for Tcl.

The following example shows how to execute the Tcl script in safe mode using the **safe-execute** keyword when the signature verification fails.

```
Device(config)# scripting tcl trustpoint untrusted safe-execute
```

Step 11 **exit**

Exits global configuration mode and returns to privileged EXEC mode.

```
Device(config)# exit
```

Step 12 **tclsafe**

(Optional) Enables the interactive Tcl shell untrusted safe mode. This allows you to manually run Tcl commands from the Cisco command line interface in untrusted safe mode.

```
Device# tclsafe
```

Example:

Verifying the Trustpoint

To display the trustpoints that are configured in the device, use the **show crypto pki trustpoints** command.

SUMMARY STEPS

1. **enable**
2. **show crypto pki trustpoints**

DETAILED STEPS

Step 1 **enable**

This command enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show crypto pki trustpoints**

This command displays the trustpoints that are configured in the device.

Example:

```
Device# show
 crypto pki trustpoints

Trustpoint mytrust:
    Subject Name:
    ea=janedoe@cisco.com
    cn=Jane
    ou=DEPT_ACCT
    o=Cisco
    l=San Jose
    st=California
    c=US
          Serial Number: 00
    Certificate configured.
```

Verifying the Signed Tcl Script

To verify that the Signed Tcl Script is properly running, use the **debug crypto pki transactions** command and the **tclsh**command.

SUMMARY STEPS

1. **enable**

2. **debug crypto pki transactions**
3. **tclsh** *flash:signed-tcl-file*

DETAILED STEPS

Step 1 **enable**

This command enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **debug crypto pki transactions**

This command display debugging messages for the trace of interaction (message type) between the CA and the device.

Example:

```
Device# debug crypto pki transactions

Crypto PKI Trans debugging is on
```

Step 3 **tclsh** *flash:signed-tcl-file*

This command executes the Tcl script in Tcl shell.

Note The file should be a signed Tcl file.

Example:

```
Device# tclsh flash:hello.tcl

hello
argc = 0
argv =
argv0 = flash:hello.tcl
tcl_interactive = 0
device#
*Apr 21 04:46:18.563: CRYPTO_PKI: locked trustpoint mytrust, refcount is 1
*Apr 21 04:46:18.563: The PKCS #7 message has 0 verified signers.
*Apr 21 04:46:18.563: CRYPTO_PKI: Success on PKCS7 verify!
*Apr 21 04:46:18.563: CRYPTO_PKI: unlocked trustpoint mytrust, refcount is 0
```

What to Do Next

- To get an overview of Crypto, refer to the “Part 5: Implementing and Managing a PKI” section of the *Security Configuration Guide*.

Configuration Examples for Signed Tcl Script

Generating a Key Pair Example

The following example shows how to generate the key pair--a private key and a public key:

Generate a Private Key: Example

```
Host% openssl genrsa -out privkey.pem 2048
Generating RSA private key, 2048 bit long modulus
.........+++
.....................................................................................+++
e is 65537 (0x10001)
Host% ls -l
total 8
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
Host%
```

Generate a Public Key from the Private Key

```
Host% openssl rsa -in privkey.pem -pubout -out pubkey.pem
writing RSA key
Host% ls -l
total 16
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

Generating a Certificate Example

The following example shows how to generate a certificate:

```
Host% openssl req -new -x509 -key privkey.pem -out cert.pem -days 1095
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value, If you enter '.', the field will be left
blank.
-----
Country Name (2 letter code) [GB]:US
State or Province Name (full name) [Berkshire]:California
Locality Name (eg, city) [Newbury]:San Jose
Organization Name (eg, company) [My Company Ltd]:Cisco Systems, Inc.
Organizational Unit Name (eg, section) []:DEPT_ACCT
Common Name (eg, your name or your server's hostname) []:Jane
Email Address []:janedoe@company.com
Host% ls -l
total 24
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

Signing the Tcl Scripts Example

The following example shows how to sign the Tcl scripts:

```
Host% openssl smime -sign -in hello -out hello.pk7 -signer cert.pem -inkey privkey.pem
-outform DER -binary
Host% ls -l
total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

Verifying the Signature Example

The following example shows how to verify the signature:

```
Host% openssl smime -verify -in hello.pk7 -CAfile cert.pem -inform DER -content hello
puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
Verification successful
```

Converting the Signature with Nonbinary Data Example

The following example shows how to convert the Tcl signature with nonbinary data:

```
#Cisco Tcl Signature V1.0
Then append the signature file to the end of the file.
Host% xxd -ps hello.pk7 > hello.hex
Host% cat my_append
#!/usr/bin/env expect
set my_first {#Cisco Tcl Signature V1.0}
set newline {}
set my_file [lindex $argv 0]
set my_new_file ${my_file}_sig
set my_new_handle [open $my_new_file w]
set my_handle [open $my_file r]

puts $my_new_handle $newline
puts $my_new_handle $my_first
foreach line [split [read $my_handle] "\n"]  {
   set new_line {#}
   append new_line $line
   puts $my_new_handle $new_line
}

close $my_new_handle
close $my_handle
Host% my_append hello.hex
Host% ls -l
total 80
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
-rw-r--r--   1 janedoe eng12       3815 Jun 13 10:20 hello.hex
```

```
-rw-r--r--   1 janedoe eng12       3907 Jun 13 10:22 hello.hex_sig
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rwxr--r--   1 janedoe eng12        444 Jun 13 10:22 my_append
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
Host% cat hello hello.hex_sig > hello.tcl
Host% cat hello.tcl
puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
#Cisco Tcl Signature V1.0
#3082075006092a864886f70d010702a08207413082073d020101310b3009
#06052b0e03021a0500300b06092a864886f70d010701a08204a13082049d
#30820385a003020102020100300d06092a864886f70d0101040500308195
#310b300906035504061302555331133011060355040813a43616c69666f
#726e69613111300f060355040713085361e204a6f7365311c301a060355
#040a13134369736f2053797374656d732c20496e632e310e300c060355
#040b13054e53535447310d300b060355040313044a6f686e3121301f0609
#2a864886f70d01090116126a6c6175746d616e40636973636f2e636f6d30
#1e170d3037303631323232303134335a170d3130303631313232303133
#5a308195310b300906035504061302555331133011060355040813a4361
#6c69666f726e69613111300f060355040713085361e204a6f7365311c30
#1a060355040a13134369736f2053797374656d732c20496e632e310e30
#0c060355040b13054e53535447310d300b060355040313044a6f686e3121
#301f06092a864886f70d01090116126a6c6175746d616e40636973636f2e
#636f6d30820122300d06092a864886f70d01010105000382010f00308201
#0a0282010100a751eb5ec1f3009738c88a55987c07b759c36f3386342283
#67ea20a89d9483ae85e0c63eeded8ab3eb7a08006689f09136f172183665
#c971099ba54e77ab47706069bbefaaab8c50184396350e4cc870c4c3f477
#88c55c52e2cf411f05b59f0eaec0678ff5cc238fdce2263a9fc6b6c244b8
#ffaead865c19c3d3172674a13b24c8f2c01dd8b1bd491c13e84e29171b85
#f28155d81ac8c69bb25ca23c2921d85fbf745c106e7aff93c72316cbc654
#4a34ea88174a8ba7777fa60662974e1fbac85a0f0aeac925dba6e5e850b8
#7caffce2fe8bb04b61b62f532b5893c081522d538005df81670b931b0ad0
#e1e76ae648f598a9442d5d0976e67c8d55889299147d0203010001a381f5
#3081f2301d0603551d0e04160414bc34132be952ff8b9e1af3b93140a255
#e54a667c3081c20603551d230481ba3081b78014bc34132be952ff8b9e1a
#f3b93140a255e54a667ca1819ba48198308195310b300906035504061302
#5553311330110603550408130a43616c69666f726e69613111300f060355
#0407130853616e204a6f7365311c301a060355040a1313436973636f2053
#797374656d732c20496e632e310e300c060355040b13054e53535447310d
#300b060355040313044a6f686e3121301f06092a864886f70d0109011612
#6a6c6175746d616e40636973636f2e636f6d820100300c0603551d130405
#30030101ff300d06092a864886f70d01010405000382010100c83c1b074
#6720929c9514af6d5df96f0a95639f047c40a607c83d8362507c58fa7f84
#aa699ec5e5bef61b2308297a0662c653ff446acfbb6f5cb2dd162d939338
#a5e4d78a5c45021e5d4dbabb8784efbf50cab0f5125d164487b31f5cf933
#a9f68f82cd111cbab1739d7f372ec460a7946882874b0a0f22dd53acbd62
#a944a15e52e54a24341b3b8a820f23a5bc7ea7b2278bb56838b8a4051926
#af9c167274ff8449003a4e012bcf4f4b3e280f85209249a390d14df47435
#35efabce720ea3d56803a84a2163db4478ae19d7d987ef6971c8312e280a
#aac0217d4fe620c6582a48faa8ea5e3726a99012e1d55f8d61b066381f77
#4158d144a43fb536c77d6a3182027730820273020101308 19b308195310b
#300906035504061302555331133011060355040813a43616c69666f726e
#69613111300f060355040713085361e204a6f7365311c301a060355040a
#1313436973636f2053797374656d732c20496e632e310e300c060355040b
#13054e53535447310d300b060355040313044a6f686e3121301f06092a86
#4886f70d01090116126a6c6175746d616e40636973636f2e636f6d020100
#300906052b0e03021a0500a081b1301806092a864886f70d010903310b06
#092a864886f70d010701301c06092a864886f70d010905310f170d303730
#3631333137313634385a302306092a864886f70d01090431160414372cb3
#72dc607990577fd0426104a42ee4158d2b305206092a864886f70d01090f
```

```
#31453043300a06082a864886f70d0307300e06082a864886f70d03020202
#0080300d06082a864886f70d0302020140300706052b0e030207300d0608
#2a864886f70d0302020128300d06092a864886f70d010101050004820100
#72db6898742f449b26d3ac18f43a1e7178834fb05ad13951bf042e127eea
#944b72b96f3b8ecf7eb52f3d0e383bf63651750223efe69eae04287c9dae
#b1f31209444108b31d34e46654c6c3cc10b5baba887825c224ec6f376d49
#00ff7ab2d9f88402dab9a2c2ab6aa3ecceeaf5a594bdc7d3a822c55e7daa
#aa0c2b067e06967f22a20e406fe21d9013ecc6bd9cd6d402c2749f8bea61
#9f8f87acfbc9e10d6ce91502e34629adca6ee855419afafe6a8233333e14
#ad4c107901d1f2bca4d7ffaadddbc54192a25da662f8b8509782c76977b8
#94879453fbb00486ccc55f88db50fcc149bae066916b350089cde51a6483
#2ec14019611720fc5bbe2400f24225fc
```

Configuring the Device with a Certificate Example

The following example shows how to configure the device with a certificate:

```
crypto pki trustpoint mytrust
 enrollment terminal
!
!
crypto pki authentication mytrust
crypto pki certificate chain mytrust
 certificate ca 00
  308204B8 308203A0 A0030201 02020100 300D0609 2A864886 F70D0101 04050030
  819E310B 30090603 55040613 02555331 13301106 03550408 130A4361 6C69666F
  726E6961 3111300F 06035504 07130853 616E204A 6F736531 1C301A06 0355040A
  13134369 73636F20 53797374 656D732C 20496E63 2E310E30 0C060355 040B1305
  4E535354 47311630 14060355 0403130D 4A6F686E 204C6175 746D616E 6E312130
  1F06092A 864886F7 0D010901 16126A6C 6175746D 616E4063 6973636F 2E636F6D
  301E170D 30363131 31373137 35383031 5A170D30 39313131 36313735 3830315A
  30819E31 0B300906 03550406 13025553 31133011 06035504 08130A43 616C6966
  6F726E69 61311130 0F060355 04071308 53616E20 4A6F7365 311C301A 06035504
  0A131343 6973636F 20537973 74656D73 2C20496E 632E310E 300C0603 55040B13
  054E5353 54473116 30140603 55040313 0D4A6F68 6E204C61 75746D61 6E6E3121
  301F0609 2A864886 F70D0109 0116126A 6C617574 6D616E40 63697363 6F2E636F
  6D308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201 0A028201
  0100BC6D A933028A B31BF827 7258BB87 A1600CF0 21090F04 2080BECC 5818688B
  74D231DF F0C365C1 07D6E206 D7651FA8 C7B230A2 3B0011E4 EA2B6A4C 1F3F27FB
  9AF449D8 FA8900BB 3E567F77 5412881B AAD9525E 3EC1D3B1 EBCE8155 D74866F1
  0940F6D1 3A2613CD F6B3595E F468B315 6DDEFF07 BBC5D521 B560AF72 D6D5FDA7
  D9D9C99D 31E3B380 5DEB7039 A1A29EF9 46ED536E 4D768048 12D48C24 59B08973
  481AD75D E741CD9E BE06EA16 9B514AE3 91184A56 A0E51B7D 4465D730 1AB3C7DD
  62CA1AC9 DF30C39A 41316B8E 72289113 98080354 C7297AD7 89B627F8 ED40D924
  ADF48383 1B332C7F 73C58686 6279E2A4 4BF41644 3E60F131 090D3F5D 25F0C025
  43CB0203 010001A3 81FE3081 FB301D06 03551D0E 04160414 F7F4E80E F6CC4772
  5F278C44 6B85F8EE 8345AB99 3081CB06 03551D23 0481C330 81C08014 F7F4E80E
  F6CC4772 5F278C44 6B85F8EE 8345AB99 A181A4A4 81A13081 9E310B30 09060355
  04061302 55533113 30110603 55040813 0A43616C 69666F72 6E696131 11300F06
  03550407 13085361 6E204A6F 7365311C 301A0603 55040A13 13436973 636F2053
  79737465 6D732C20 496E632E 310E300C 06035504 0B13054E 53535447 31163014
  06035504 03130D4A 6F686E20 4C617574 6D616E6E 3121301F 06092A86 4886F70D
  01090116 126A6C61 75746D61 6E406369 73636F2E 636F6D82 0100300C 0603551D
  13040530 030101FF 300D0609 2A864886 F70D0101 04050003 82010100 6D12CFF8
  31078DF6 94FE5CF0 8F83639B 414F32D8 069D23E2 37E182BE 7C31EC14 E87AF216
  61A6CCD3 37656934 4BE4157A 400E182B EC390D1A DC130A56 B8F35BFB D2234556
  24152FE8 A736B670 58CC684E 750D08AE C7739907 917B7A72 3D26BEC7 9F554CF1
  5E5EF499 ABA11124 55966616 AC9C52B2 B1082DEA D962CBAF E476C575 A9DDFBFA
  C4AE63F6 1D5C9F76 7B4B9CA7 52CE65C9 E65C04FC 4B7642D6 0D1A8AF4 38194B7A
  CA307EC9 51DCB847 8B8C27FB 98ACEE60 0B80DC3F 36E4E252 BD731F5F 0E781E26
  C1CA4120 9B0B689B BA654250 97B22A76 CC126B77 C7779AAA D3F93C3F DCF46006
  2B7F7F8C 150AF889 BBEC62F1 E53B4F3B A3626CD6 05B8AB3D F8A6A361
```

```
   quit
archive
 log config
scripting tcl trustpoint name mytrust
scripting tcl secure-mode
!
!
end
```

Additional References

The following sections provide references related to the Signed Tcl Scripts feature.

Related Documents

Related Topic	Document Title
Cisco PKI Overview: Understanding and Planning a PKI Implementing and Managing a PKI	*Security Configuration Guide, Release 12.4*
PKI commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples.	*Cisco IOS Security Command Reference, Release 12.4*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Signed Tcl Scripts

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 74: Feature Information for Signed Tcl Scripts

Feature Name	Releases	Feature Information
Signed Tcl Scripts		The Signed Tcl Scripts feature allows you to create a certificate to generate a digital signature and sign a Tcl script with that digital signature. The following commands were introduced by this feature: **scripting tcl secure-mode**, **scripting tcl trustpoint name**, **scripting tcl trustpoint untrusted**,and **tclsafe.**

Glossary

CA--certification authority. Service responsible for managing certificate requests and issuing certificates to participating IPsec network devices. This service provides centralized key management for the participating devices and is explicitly trusted by the receiver to validate identities and to create digital certificates.

certificates--Electronic documents that bind a user's or device's name to its public key. Certificates are commonly used to validate a digital signature.

CRL--certificate revocation list. Electronic document that contains a list of revoked certificates. The CRL is created and digitally signed by the CA that originally issued the certificates. The CRL contains dates for when the certificate was issued and when it expires. A new CRL is issued when the current CRL expires.

IPsec--IP security

peer certificate--Certificate presented by a peer, which contains the peer's public key and is signed by the trustpoint CA.

PKI--public key infrastructure. System that manages encryption keys and identity information for components of a network that participate in secured communications.

RA--registration authority. Server that acts as a proxy for the CA so that CA functions can continue when the CA is offline. Although the RA is often part of the CA server, the RA could also be an additional application, requiring an additional device to run it.

RSA keys--Public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adleman. An RSA key pair (a public and a private key) is required before you can obtain a certificate for your device.

SHA1--Secure Hash Algorithm 1

SSH--secure shell

SSL--secure socket layer

Notices

The following notices pertain to this software license.

OpenSSL Open SSL Project

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product includes software written by Tim Hudson (tjh@cryptsoft.com).

License Issues

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”.

4. The names “OpenSSL Toolkit” and “OpenSSL Project” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:

“This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”.

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT “AS IS”" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License:

Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young’s, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

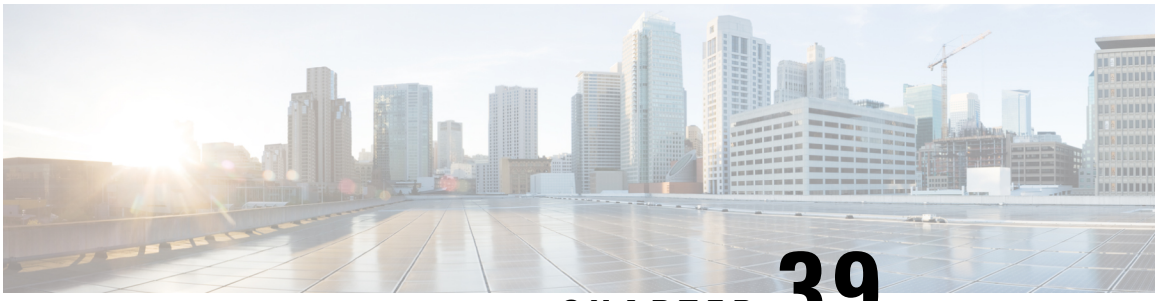
“This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)”.

The word ‘cryptographic’ can be left out if the routines from the library being used are not cryptography-related.

1. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: “This product includes software written by Tim Hudson (tjh@cryptsoft.com)”.

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The license and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License].



CHAPTER 39

EEM Action Tcl Command Extension

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

action_policy

Allows a Tcl script to run an Embedded Event Manager (EEM) policy that has been registered with the None event detector. The action of running an EEM policy can also be performed using the **event manager run** command.

Syntax

```
action_policy ?
```

Arguments

? (represents a string)	(Mandatory) The name of the EEM policy to be scheduled for execution. The policy must have been previously registered with the None event detector.

None

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err = 2) FH_ESYSERR (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID (unknown event ID)
```

This error means that the policy is unknown because it is not registered.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

action_process

Starts, restarts, or kills a Software Modularity process. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
action_process start|restart|kill [job_id ?]
[process_name ?] [instance ?]
```

Arguments

start	(Mandatory) Specifies that a process is to be started.
restart	(Mandatory) Specifies that a process is to be restarted.
kill	(Mandatory) Specifies that a process is to be stopped (killed).
job_id	(Optional) System manager assigned job ID for the process. If you specify this argument, it must be an integer between 1 and 4294967295, inclusive.
process_name	(Optional) Process name. Either job_id must be specified or process_name and instance must be specified.
instance	(Optional) Process instance ID. If you specify this argument, it must be an integer between 1 and 4294967295, inclusive.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

```
(_cerr_sub_num = 425, _cerr_sub_err = 1) SYSMGR_ERROR_INVALID_ARGS  (Invalid arguments
passed)
```

This error means that the arguments passed in were invalid.

```
(_cerr_sub_num = 425, _cerr_sub_err = 2) SYSMGR_ERROR_NO_MEMORY  (Could not allocate required
 memory)
```

This error means that an internal SYSMGR request for memory failed.

```
(_cerr_sub_num = 425, _cerr_sub_err = 5) SYSMGR_ERROR_NO_MATCH  (This process is not known
 to sysmgr)
```

This error means that the process name was not known.

```
(_cerr_sub_num = 425, _cerr_sub_err = 14) SYSMGR_ERROR_TOO_BIG  (outside the valid limit)
```

This error means that an object size exceeded its maximum.

```
(_cerr_sub_num = 425, _cerr_sub_err = 15) SYSMGR_ERROR_INVALID_OP  (Invalid operation for
this process)
```

This error means that the operation was invalid for the process.

action_program

Allows a Tcl script to run a POSIX process (program), optionally with a given argument string, environment string, Standard Input (stdin) pathname, Standard Output (stdout) pathname, or Standard Error (stderr) pathname. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
action_program path ? [argv ?] [envp ?] [stdin ?] [stdout ?] [stderr ?]
```

Arguments

path	(Mandatory) The pathname of a program to run.
argv	(Optional) The argument string of the program.
envp	(Optional) The environment string of the program.
stdin	(Optional) The pathname for stdin.
stdout	(Optional) The pathname for stdout.
stderr	(Optional) The pathname for stderr.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

```
(_cerr_sub_err = 34)    FH_EMAXLEN  (maximum length exceeded)
```

This error means that the object length or number exceeded the maximum.

action_reload

Reloads the device.

Syntax

```
action_reload
```

Arguments

None

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

action_script

Allows a Tcl script to enable or disable the execution of all Tcl scripts (enables or disables the script scheduler).

Syntax

```
action_script [status enable|disable]
```

Arguments

status	(Optional) Flag to indicate script execution status. If this argument is set to enable, script execution is enabled; if this argument is set to disable, script execution is disabled.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

```
(_cerr_sub_err = 52)    FH_ECONFIG  (configuration error)
```

This error means that a configuration error has occurred.

action_snmp_trap

Sends a Simple Network Management Protocol (SNMP) trap using the Embedded Event Manager Notification MIB.

Syntax

```
action_snmp_trap [intdata1 ?] [intdata2 ?] [strdata ?]
```

Arguments

intdata1	(Optional) Arbitrary integer sent in trap.
intdata2	(Optional) Arbitrary integer sent in trap.
strdata	(Optional) Arbitrary string data sent in trap.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

action_snmp_object_value

Sets a Simple Network Management Protocol (SNMP) object ID and value to be returned for the SNMP get request.

Syntax

```
action_snmp_object_value {int|uint|counter|gauge|ipv4|octet|counter64|string} ?
[next_oid ?]
```

Arguments

int	A 32-bit number used to specify a numbered type within the context of a managed object.
uint	A 32-bit number used to represent decimal value.
counter	A 32-bit number with a minimum value of 0.
gauge	A 32-bit number with a minimum value of 0.
ipv4	IP version 4 address.
octet	An octet string in hex notation used to represent physical addresses.
counter 64	A 64-bit number with a minimum value of 0.
string	An octet string in text notation used to represent text strings.
next_oid	The OID of the next object in the table; NULL if it is the last object in the table.

Result String

None

Set _cerrno

Yes

action_switch

Switches processing to a secondary processor in a fully redundant environment. Before using the **action_switch** Tcl command extension, you must install a backup processor in the device. If the hardware is not fully redundant, the switchover action will not be performed.

Syntax

```
action_switch
```

Arguments

None

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err = 2) FH_ESYSERR (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

This error means that the action command requested was unknown.

action_syslog

Generates a periodic syslog message using the specified facility when an EEM script is triggered.

Syntax

```
action_syslog [priority emerg|alert|crit|err|warning|notice|info|debug]
[msg ?] [facility ?]
```

Arguments

priority	(Optional) The action_syslog message facility level. If this argument is not specified, the default priority is LOG_INFO.
msg	(Optional) The message to be logged.
facility	(Optional) Syslog facility.

Result String

None

Set _cerrno

Yes

action_track_read

Reads the state of a tracked object when an Embedded Event Manager (EEM) script is triggered.

Syntax

```
action_track_read ?
```

Arguments

? (represents a number)	(Mandatory) Tracked object number in the range from 1 to 500, inclusive.

Result String

```
number {%u}
state {%s}
```

Set _cerrno

Yes

```
FH_ENOTRACK
```

This error means that the tracked object number was not found.

action_track_set

Sets the state of a tracked object when an Embedded Event Manager (EEM) script is triggered.

Syntax

```
action_track_set ? state up|down
```

Arguments

? (represents a number)	(Mandatory) Tracked object number in the range from 1 to 500, inclusive.
state	(Mandatory) Specifies that the state of the tracked object will be set. If up is specified, the state of the tracked object will be set to up. If down is specified, the state of the tracked object will be set to down.

Result String

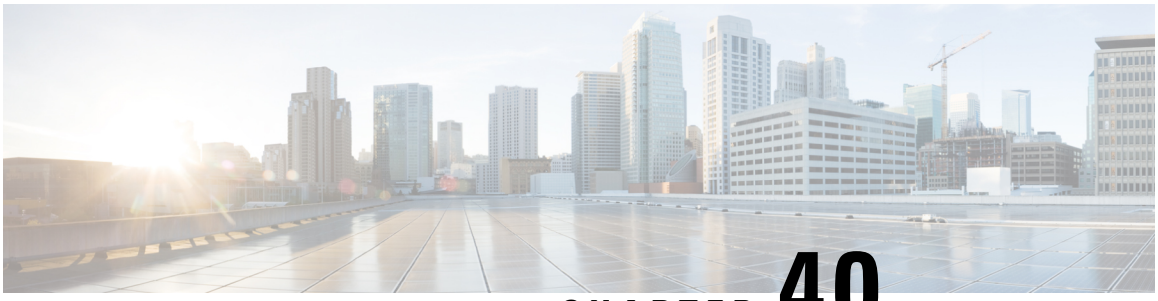
None

Set _cerrno

Yes

```
FH_ENOTRACK
```

This error means that the tracked object number was not found.



CHAPTER 40

EEM CLI Library Command Extensions

All command-line interface (CLI) library command extensions belong to the ::cisco::eem namespace.

This library provides users the ability to run CLI commands and get the output of the commands in Tcl. Users can use commands in this library to spawn an exec and open a virtual terminal channel to it, write the command to execute to the channel so that the command will be executed by exec, and read back the output of the command.

There are two types of CLI commands: interactive commands and non-interactive commands.

For interactive commands, after the command is entered, there will be a "Q&A" phase in which the device will ask for different user options, and the user is supposed to enter the answer for each question. Only after all the questions have been answered properly will the command run according to the user's options until completion.

For noninteractive commands, once the command is entered, the command will run to completion. To run different types of commands using an EEM script, different CLI library command sequences should be used, which are documented in the "Using the CLI Library to Run a Noninteractive Command" section and in the "Using the CLI Library to Run an Interactive Command" section in the cli_write Tcl command.

The vty lines are allocated from the pool of vty lines that are configured using the **line vty** CLI configuration command. EEM will use a vty line when a vty line is not being used by EEM and there are available vty lines. EEM will also use a vty line when EEM is already using a vty line and there are three or more vty lines available. Be aware that the connection will fail when fewer than three vty lines are available, preserving the remaining vty lines for Telnet use.

Your release may support XML-PI. For details about the XML-PI support, the new CLI library command extensions, and some examples of how to implement XML-PI, see EEM CLI Library XML-PI Support.

cli_close

Closes the exec process and releases the vty and the specified channel handler connected to the command-line interface (CLI).

Syntax

```
cli_close fd tty_id
```

Arguments

fd	(Mandatory) The CLI channel handler.
tty_id	(Mandatory) The TTY ID returned from the **cli_open** command extension.

Result String

None

Set _cerrno

Cannot close the channel.

cli_exec

Writes the command to the specified channel handler to execute the command. Then reads the output of the command from the channel and returns the output.

Syntax

```
cli_exec fd cmd
```

Arguments

fd	(Mandatory) The command-line interface (CLI) channel handler.
cmd	(Mandatory) The CLI command to execute.

Result String

The output of the CLI command executed.

Set _cerrno

Error reading the channel.

cli_get_ttyname

Returns the real and pseudo TTY names for a given TTY ID.

Syntax

```
cli_get_ttyname tty_id
```

Arguments

tty_id	(Mandatory) The TTY ID returned from the **cli_open** command extension.

Result String

```
pty %s tty %s
```

Set _cerrno

None

cli_open

Allocates a vty, creates an EXEC command-line interface (CLI) session, and connects the vty to a channel handler. Returns an array including the channel handler.

Note Each call to **cli_open** initiates a Cisco IOS EXEC session that allocates a Cisco IOS vty line. The vty remains in use until the **cli_close** routine is called. The vty lines are allocated from the pool of vty lines that are configured using the **line vty** CLI configuration command. EEM will use a vty line when a vty line is not being used by EEM and there are available vty lines. EEM will also use a vty line when EEM is already using a vty line and there are three or more vty lines available. Be aware that the connection will fail when fewer than three vty lines are available, preserving the remaining vty lines for Telnet use

Syntax

```
cli_open
```

Arguments

None

Result String

```
"tty_id {%s} pty {%d} tty {%d} fd {%d}"
```

Event Type	Description
tty_id	TTY ID.
pty	PTY device name.
tty	TTY device name.
fd	CLI channel handler.

Set _cerrno

- Cannot get pty for EXEC.
- Cannot create an EXEC CLI session.
- Error reading the first prompt.

cli_read

Reads the command output from the specified command-line interface (CLI) channel handler until the pattern of the device prompt occurs in the contents read. Returns all the contents read up to the match.

Syntax

```
cli_read fd
```

Arguments

fd	(Mandatory) The CLI channel handler.

Result String

All the contents read.

Set _cerrno

Cannot get device name.

Note This Tcl command extension will block waiting for the device prompt to show up in the contents read.

cli_read_drain

Reads and drains the command output of the specified command-line interface (CLI) channel handler. Returns all the contents read.

Syntax

```
cli_read_drain fd
```

Arguments

fd	(Mandatory) The CLI channel handler.

Result String

All the contents read.

Set _cerrno

None

cli_read_line

Reads one line of the command output from the specified command-line interface (CLI) channel handler. Returns the line read.

Syntax

```
cli_read_line fd
```

Arguments

fd	(Mandatory) The CLI channel handler.

Result String

The line read.

Set _cerrno

None

Note This Tcl command extension will block waiting for the end of line to show up in the contents read.

cli_read_pattern

Reads the command output from the specified command-line interface (CLI) channel handler until the pattern that is to be matched occurs in the contents read. Returns all the contents read up to the match.

Note The pattern matching logic attempts a match by looking at the command output data as it is delivered from the Cisco IOS command. The match is always done on the most recent 256 characters in the output buffer unless there are fewer characters available, in which case the match is done on fewer characters. If more than 256 characters in the output buffer are required for the match to succeed, the pattern will not match.

Syntax

```
cli_read_pattern fd ptn
```

Arguments

fd	(Mandatory) The CLI channel handler.
ptn	(Mandatory) The pattern to be matched when reading the command output from the channel.

Result String

All the contents read.

Set _cerrno

None

Note This Tcl command extension will block waiting for the specified pattern to show up in the contents read.

cli_run

Iterates over the items in the clist and assumes that each one is a command-line-interface (CLI) command to be executed in the enable mode. On success, returns the output of all executed commands and on failure, returns error from the failure.

Syntax

```
cli_run clist
```

Arguments

clist	(Mandatory) The list of commands to be executed.

Result String

Output of all the commands that are executed or an error message.

Set _cerrno

None.

Sample Usage

The following example shows how to use the **cli_run** command extension.

```
set clist [list {sh run} {sh ver} {sh event man pol reg}]
cli_run { clist }
```

cli_run_interactive

Provides a sublist to the clist which has three items. On success, returns the output of all executed commands and on failure, returns error from the failure. Also uses arrays when possible as a way of making things easier to read later by keeping expect and reply separated.

Syntax

```
cli_run_interactive clist
```

Arguments

clist	(Mandatory) List of three items: • command– Command to be executed • expect– A regular expression pattern match for the expected reply prompt • responses– A list of possible responses to the reply prompt constructed as an array of two items: • expect– A regular expression pattern match for a possible reply prompt • reply- A reply for that expected prompt

Result String

Output of all the commands that are executed or an error message. As each command is executed its output is appended to a result variable. Upon exhaustion of the input list, the CLI channel is closed and the aggregate result is returned.

Set _cerrno

None.

Sample Usage

The following example shows how to clear counters for interface fa0/0 use the cli_run_ interactive command extension.

```
set cmdarr(command) "clear counters fa0/0"
set cmdarr(responses) [list]
set resps(expect) {[confirm]}
set resps(reply) "y"
lappend cmdarr(responses) [array get resps]
set rc [catch {cli_run_interactive [list [array get cmdarr]]} result]
```

Possible errors raised include:

- cannot get pty for exec
- cannot spawn exec
- error reading the first prompt
- error reading the channel
- cannot close channel

cli_write

Writes the command that is to be executed to the specified CLI channel handler. The CLI channel handler executes the command.

Syntax

```
cli_write fd cmd
```

Arguments

fd	(Mandatory) The CLI channel handler.
cmd	(Mandatory) The CLI command to execute.

Result String

None

Set _cerrno

None

Sample Usage

As an example, use configuration CLI commands to bring up Ethernet interface 1/0:

```
if [catch {cli_open} result] {
puts stderr $result
exit 1
} else {
array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
puts stderr $result
exit 1
```

```
}
if [catch {cli_exec $cli1(fd) "config t"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "interface Ethernet1/0"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "no shut"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "end"} result] {
puts stderr $result
exit 1
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} } result] {
puts stderr $result
exit 1
```

Using the CLI Library to Run a Noninteractive Command

To run a noninteractive command, use the **cli_exec** command extension to issue the command, and then wait for the complete output and the device prompt. For example, the following shows the use of configuration CLI commands to bring up Ethernet interface 1/0:

```
if [catch {cli_open} result] {
error $result $errorInfo
} else {
set fd $result
}
if [catch {cli_exec $fd "en"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "config t"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "interface Ethernet1/0"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "no shut"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "end"} result] {
error $result $errorInfo
}
if [catch {cli_close $fd} result] {
error $result $errorInfo
}
```

Using the CLI Library to Run an Interactive Command

To run interactive commands, three phases are needed:

- Phase 1: Issue the command using the **cli_write** command extension.
- Phase 2: Q&A Phase. Use the **cli_read_pattern** command extension to read the question (the regular pattern that is specified to match the question text) and the **cli_write** command extension to write back the answers alternately.

- Phase 3: Noninteractive phase. All questions have been answered, and the command will run to completion. Use the **cli_read** command extension to wait for the complete output of the command and the device prompt.

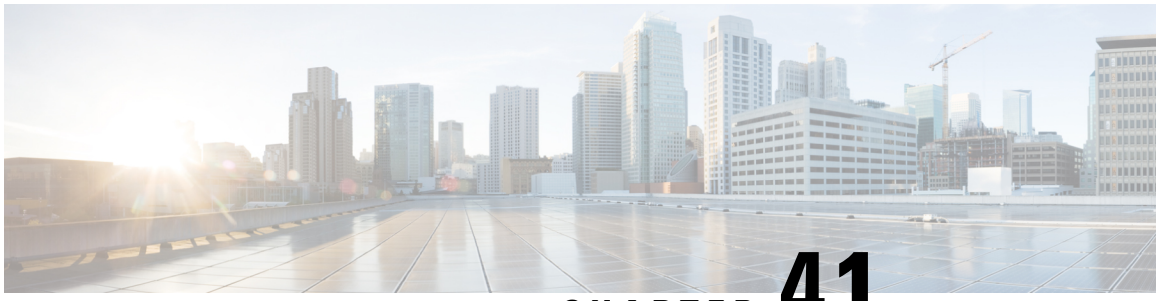
For example, use CLI commands to do squeeze bootflash: and save the output of this command in the Tcl variable cmd_output.

```
if [catch {cli_open} result] {
error $result $errorInfo
} else {
array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
error $result $errorInfo
}

# Phase 1: issue the command
if [catch {cli_write $cli1(fd) "squeeze bootflash:"} result] {
error $result $errorInfo
}

# Phase 2: Q&A phase
# wait for prompted question:
# All deleted files will be removed. Continue? [confirm]
if [catch {cli_read_pattern $cli1(fd) "All deleted"} result] {
error $result $errorInfo
}
# write a newline character
if [catch {cli_write $cli1(fd) "\n"} result] {
error $result $errorInfo
}
# wait for prompted question:
# Squeeze operation may take a while. Continue? [confirm]
if [catch {cli_read_pattern $cli1(fd) "Squeeze operation"} result] {
error $result $errorInfo
}
# write a newline character
if [catch {cli_write $cli1(fd) "\n"} result] {
error $result $errorInfo
}

# Phase 3: noninteractive phase
# wait for command to complete and the router prompt
if [catch {cli_read $cli1(fd) } result] {
error $result $errorInfo
} else {
set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
error $result $errorInfo
}
```

The following example causes a device to be reloaded using the CLI **reload** command. Note that the EEM **action_reload** command accomplishes the same result in a more efficient manner, but this example is presented to illustrate the flexibility of the CLI library for interactive command execution.

```
# 1. execute the reload command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
```

```
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
if [catch {cli_write $cli1(fd) "reload"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_read_pattern $cli1(fd) ".*(System configuration has been modified. Save\\\?
 \\\[yes/no\\\]: )"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_write $cli1(fd) "no"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_read_pattern $cli1(fd) ".*(Proceed with reload\\\? \\\[confirm\\\])"} result]
 {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_write $cli1(fd) "y"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}
```



CHAPTER 41

EEM CLI Library XML-PI Support

XML Programmatic Interface (XML-PI) was introduced in Cisco IOS Release 12.4(22)T. XML-PI provides a programmable interface which encapsulates IOS command-line interface (CLI) show commands in XML format in a consistent way across different Cisco products. Customers using XML-PI will be able to parse IOS show command output from within Tcl scripts using well-known keywords instead of having to depend on the use of regular expression support to "screen-scrape" output.

The benefit of using the XML-PI command extensions is to facilitate the extraction of specific output information that is generated using a CLI **show** command. Most show commands return many fields within the output and currently a regular expression has to be used to extract specific information that may appear in the middle of a line. XML-PI support provides a set of Tcl library functions to facilitate the parsing of output from the IOS CLI format extension in the form of:

```
show
 <
show-command
> | format
 {
spec-file
}
```

where a spec-file is a concatenation of all Spec File Entries (SFE) for each **show** command currently supported. As part of the XML-PI project a default spec-file will be included in the IOS Release 12.4(22)T images. The default spec-file will have a small set of commands and the SFE for the commands will have a subset of the possible tags. If no spec-file is provided with the format command, the default spec-file is used.

For more general details about XML-PI, see the "XML-PI" module.

xml_pi_exec

Writes the XML-PI command specified using the cmd argument to the channel whose handler is specified using the fd argument and the spec-file specified by the spec_file argument to execute the command. The raw XML output data of the command is then read from the channel and the XML output is returned.

Syntax

```
xml_pi_show fd cmd [spec_file]
```

Arguments

fd	(Mandatory) The CLI library file descriptor obtained from cli_open.
cmd	(Mandatory) IOS show command.
spec_file	(Optional) IOS CLI show command spec_file.

Result String

Result of IOS show command in XML format.

Set _cerrno

Possible error raised:

1. error reading the channel

xml_pi_parse

Processes the XML show command raw output passed into this function as xml_data and retrieve those fields that are specified by xml_tags_list. The following processing occurs:

Step 1: The XML tag list is validated as a Tcl list. An XML tag can be specified as the low order XML tag name or as a fully qualified XML tag name in case the low order name is ambiguous for a given command.

Example tags: <Interface> <ShowIpInterfaceBrief><IPInterfaces><entry><Interface>

Step 2: The xml_data is validated as valid XML and parsed into an XML parse tree.

Step 3: A walk is made through the XML parse tree and each tag is compared with entries in the XML tag list. When a match occurs it is determined if the tag name matches a Tcl procedure defined within the current Tcl scope. If so, that Tcl procedure will be called with the current result. If not, the tag name and the data associated with that tag name will be appended to the current result.

Syntax

```
xml_pi_parse fd xml_show_cmd_output xml_tags_list
```

Arguments

fd	(Mandatory) The CLI library file descriptor obtained from cli_open.
xml_show_cmd_output	(Mandatory) Output of xml_pi_show command extension in xml format.
xml_tags_list	(Mandatory) List of interesting tags.

Result String

Data in a Tcl array indexed by XML tag name.

Note The current result is reset after Tcl procedure calls.

Set _cerrno

Possible errors raised:

1. error splitting the XML tags list 2. null XML tag list specified 3. XML tag tree exceeds 20 levels 4. called Tcl procedure returned an error 5. memory allocation failure 6. XML parse failure 7. failed to create XML domain

xml_pi_read

Reads the XML-PI command output (from the specified show command) from the CLI channel whose handler is given by the file descriptor until the pattern of the router prompt occurs in the contents that are read. Returns all the contents read up to the match in XML format.

Syntax

```
xml_pi_read fd
```

Arguments

fd	(Mandatory) The CLI library file descriptor obtained from cli_open.

Result String

All the contents that are read in XML format.

Set _cerrno

Possible errors raised:

1. cannot get router name 2. command error

xml_pi_write

Writes the XML-PI command specified using the cmd argument to the channel whose handler is given by the fd argument and the spec file specified by the spec_file argument.

Syntax

```
xml_pi_write fd cmd spec_file
```

Arguments

fd	(Mandatory) The CLI library file descriptor obtained from cli_open.
cmd	(Mandatory) IOS show command.
spec_file	(Optional) IOS CLI show command spec_file.

Result String

None

Set _cerrno

None

Sample Usage of the XML-PI feature

The following EEM policy (sample.tcl) presents one example that illustrates five different implementations of the new EEM XML-PI functionality. The odm spec-file (required for Example 2) follows this policy.

```
::cisco::eem::event_register_none maxrun 60
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# open the cli_lib.tcl channel
if [catch {cli_open} result] {
error $result $errorInfo
} else {
array set cli1 $result
}
# enter "enable" privilege mode
if [catch {cli_exec $cli1(fd) "en"} result] {
error $result $errorInfo
}
# Example 1:
#
# Detect if XML-PI is present in this image
# Invoke xml_pi_exec with the default spec file for the "show inventory"
# command. After the command executes $result contains the raw XML data if
# the command is successful.
if [catch {xml_pi_exec $cli1(fd) "show inventory" ""} result] {
puts "Example 1: XML-PI support is not present in this image - exiting"
exit
} else {
puts "Example 1: XML-PI support is present in this image"
}
# Example 2:
#
# In the next example we demonstrate how to extract two data elements
# from the "show version" command using the specified XML-PI spec file.
# The raw output from this command is as follows:
#
# Device#show version | format disk2:speceemtest.odm
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowVersion>
# <Version>12.4(20071029:194217)</Version>
# <Compiled>Thu 08-Nov-07 11:28</Compiled>
# <ROM>System Bootstrap, Version 12.2(20030826:190624) [BLD-npeg1_rommon_r11 102],
DEVELOPMENT</ROM>
# <uptime>17 minutes</uptime>
```

```
# <processor>NPE-G1</processor>
# <bytesofmemory>983040K/65536K</bytesofmemory>
# <CPU>700MHz</CPU>
# <L2Cache>0.2</L2Cache>
# <GigabitEthernetinterfaces>3</GigabitEthernetinterfaces>
# <bytesofNVRAM>509K</bytesofNVRAM>
# <bytesofATAPCMCIAcard>125952K</bytesofATAPCMCIAcard>
# <Sectorsize>512 bytes</Sectorsize>
# <bytesofFlashinternalSIMM>16384K</bytesofFlashinternalSIMM>
# <Configurationregister>0x2100</Configurationregister>
# </ShowVersion>
#
# Invoke xml_pi_exec with the spec file "disk2:speceemtest.odm" for the
# "show version" command. After the command executes $result contains
# the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show version" "disk2:speceemtest.odm"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <processor> and <CPU> fields be returned.
array set xml_result [xml_pi_parse $cli1(fd) $result "<processor> <CPU>"]
puts "Example 2: Processor is $xml_result(<processor>) CPU is $xml_result(<CPU>)"
}
# Example 3:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show inventory" command using the default built-in
# XML-PI spec file. Sample raw output from this command is as follows:
#
# Device#show inventory | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowInventory>
# <SpecVersion>built-in</SpecVersion>
# <InventoryEntry>
# <ChassisName>&quot;Chassis&quot;</ChassisName>
# <Description>&quot;Cisco 7206VXR, 6-slot chassis&quot;</Description>
# <PID>CISCO7206VXR</PID>
# <VID>
# </VID>
# <SN>31413378 </SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>&quot;NPE-G1 0&quot;</ChassisName>
# <Description>&quot;Cisco 7200 Series Network Processing Engine
NPE-G1&quot;</Description>
# <PID>NPE-G1</PID>
# <VID>
# </VID>
# <SN>31493825 </SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>&quot;disk2&quot;</ChassisName>
# <Description>&quot;128MB Compact Flash Disk for NPE-G1&quot;</Description>
# <PID>MEM-NPE-G1-FLD128</PID>
# <VID>
# </VID>
# <SN>NAME: &quot;module 1&quot;</SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>&quot;module 1&quot;</ChassisName>
# <Description>&quot;Dual Port FastEthernet (RJ45)&quot;</Description>
# <PID>PA-2FE-TX</PID>
# <VID>
```

```
# </VID>
# <SN>JAE0827NGKX</SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>&quot;Power Supply 2&quot;</ChassisName>
# <Description>&quot;Cisco 7200 AC Power Supply&quot;</Description>
# <PID>PWR-7200-AC</PID>
# <VID>
# </VID>
# </InventoryEntry>
# </ShowInventory>
#
# Define a procedure to be called every time the <InventoryEntry> tag
# is processed. Since this tag precedes each new output record, the data
# that is passed into this procedure contains the fields that have been
# requested via xml_pi_parse since the previous time this procedure was
# called.
proc <InventoryEntry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
set output [format "Example 3: Item %2d %-18s %s" \
$num $xml_result(<PID>) $xml_result(<Description>)]
puts $output
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show inventory" command. After the command executes $result contains
# the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show inventory"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <PID> and <Description> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <InventoryEntry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<InventoryEntry> <PID> <Description>"]
# Display the data from the last record.
incr num
set output [format "Example 3: Item %2d %-18s %s" \
$num $xml_result(<PID>) $xml_result(<Description>)]
puts $output
}
# Example 4:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show ip interface brief" command using the default
# built-in XML-PI spec file. Sample raw output from this command is as
# follows:
#
# Device#show ip interface brief | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowIpInterfaceBrief>
# <SpecVersion>built-in</SpecVersion>
# <IPInterfaces>
# <entry>
```

```
# <Interface>GigabitEthernet0/1</Interface>
# <IP-Address>172.19.209.34</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>up</Status>
# <Protocol>up</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/2</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/3</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/0</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/1</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# </IPInterfaces>
# </ShowIpInterfaceBrief>
#
# Define a procedure to be called every time the fully qualified name
# <ShowIpInterfaceBrief><IPInterfaces><entry> tag is processed. Since
# this tag precedes each new output record, the data that is passed into
# this procedure contains the fields that have been requested via
# xml_pi_parse since the previous time this procedure was called.
proc <ShowIpInterfaceBrief><IPInterfaces><entry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
set output [format "Example 4: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
} else {
puts "Example 4: Display All Interfaces"
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show ip interface brief" command. After the command executes $result
```

```
# contains the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show ip interface brief"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <Interface> and <Status> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <entry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<ShowIpInterfaceBrief><IPInterfaces><entry> <Interface> <Status>"]
# Display the data from the last record.
incr num
set output [format "Example 4: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
# Example 5:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show ip interface brief" command using the default
# built-in XML-PI spec file. Sample raw output from this command is as
# follows:
#
# Device#show ip interface brief | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowIpInterfaceBrief>
# <SpecVersion>built-in</SpecVersion>
# <IPInterfaces>
# <entry>
# <Interface>GigabitEthernet0/1</Interface>
# <IP-Address>172.19.209.34</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>up</Status>
# <Protocol>up</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/2</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/3</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/0</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
```

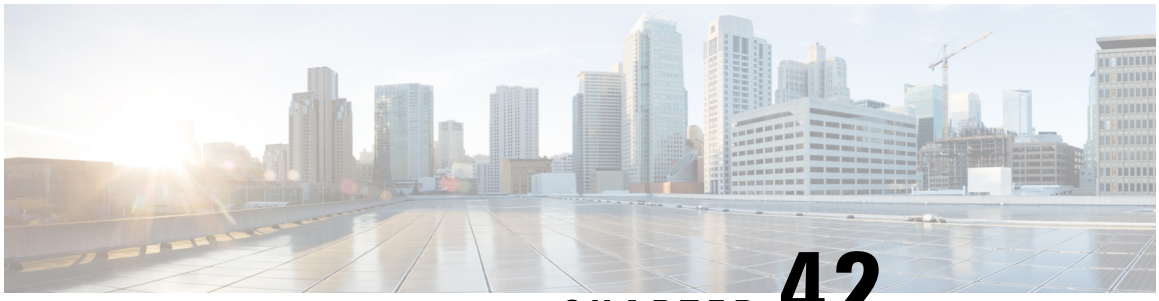
```
# <Interface>FastEthernet1/1</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# </IPInterfaces>
# </ShowIpInterfaceBrief>
#
# Note: This example is the same as Example 4 with the exception that
# the new record procedure is called by the un-qualified tag name. The
# ability to specify the un-qualified tag names is simplier but only works
# if the un-qualified name is used once per Tcl program. In this example
# the unqualified new record tag name is "<entry>" which is a very
# common name in the Cisco spec file.
# Define a procedure to be called every time the <entry> tag
# is processed. Since this tag precedes each new output record, the data
# that is passed into this procedure contains the fields that have been
# requested via xml_pi_parse since the previous time this procedure was
# called.
proc <entry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
if ([string equal $xml_result(<Status>) "up"]) {
set output [format "Example 5: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
} else {
puts "Example 5: Display All Interfaces That Are Up"
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show ip interface brief" command. After the command executes $result
# contains the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show ip interface brief"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <Interface> and <Status> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <entry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<entry> <Interface> <Status>"]
# Display the data from the last record.
incr num
if ([string equal $xml_result(<Status>) "up"]) {
set output [format "Example 5: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
}
```

Sample XML-PI spec eemtest.odm ODM File:

```
###
show version
<?xml version='1.0' encoding='utf-8'?>
<ODMSpec>
<Command>
<Name>show version</Name>
</Command>
<OS>ios</OS>
<DataModel>
<Container name="ShowVersion">
<Property name="Version" distance = "1.0" length = "1" type = "IpAddress"/>
<Property name="Technical Support" distance = "1.0" length = "1" type = "IpAddress"/>
<Property name="Compiled" distance = "1.0" length = "3" type = "String"/>
<Property name="ROM" distance = "1.0" length = "7" type = "IpAddress"/>
<Property name="uptime" distance = "2" length = "8" type = "String"/>
<Property name="image" distance = "4" length = "1" type = "IpAddress"/>
<Property name="processor" distance = "-1" length = "1" type = "String"/>
<Property name="bytes of memory" distance = "-1" length = "1" type = "Port"/>
<Property name="CPU" distance = "2" length = "1" end-delimiter = "," type = "String"/>
<Property name="L2 Cache" distance = "-2" length = "1" end-delimiter = "," type = "String"/>
<Property name="Gigabit Ethernet interfaces" distance = "-1" length = "1" type = "Integer"/>
<Property name="bytes of NVRAM" distance = "-1" length = "1" type = "String"/>
<Property name="bytes of ATA PCMCIA card" distance = "-1" length = "1" type = "String"/>
<Property name="Sector size" distance = "1.0" length = "2" end-delimiter = ")" type =
"String"/>
<Property name="bytes of Flash internal SIMM" distance = "-1" length = "1" type = "String"/>
<Property name="Configuration register" distance = "2" length = "1" type = "String"/>
</Container>
</DataModel>
</ODMSpec>
```

Example sample.tcl Run:

```
Device#config t
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)#event manager policy sample.tcl
Device(config)#end
Device#
Oct 10 20:21:26: %SYS-5-CONFIG_I: Configured from console by console
Device#event manager run sample.tcl
Example 1: XML-PI support is present in this image
Example 2: Processor is NPE-G1 CPU is 700MHz
Example 3: Item 1 CISCO7206VXR "Cisco 7206VXR, 6-slot chassis"
Example 3: Item 2 NPE-G1 "Cisco 7200 Series Network Processing Engine NPE-G1"
Example 3: Item 3 MEM-NPE-G1-FLD128 "128MB Compact Flash Disk for NPE-G1"
Example 3: Item 4 PA-2FE-TX "Dual Port FastEthernet (RJ45)"
Example 3: Item 5 PWR-7200-AC "Cisco 7200 AC Power Supply"
Example 4: Display All Interfaces
Example 4: Interface 1 GigabitEthernet0/1 up
Example 4: Interface 2 GigabitEthernet0/2 administratively down
Example 4: Interface 3 GigabitEthernet0/3 administratively down
Example 4: Interface 4 FastEthernet1/0 administratively down
Example 4: Interface 5 FastEthernet1/1 administratively down
Example 4: Interface 6 SSLVPN-VIF0 up
Example 5: Display All Interfaces That Are Up
Example 5: Interface 1 GigabitEthernet0/1 up
Example 5: Interface 6 SSLVPN-VIF0 up
```



CHAPTER 42

EEM Context Library Command Extensions

All the Tcl context library command extensions belong to the ::cisco::eem namespace.

context_retrieve

Retrieves Tcl variable(s) identified by the given context name, and possibly the scalar variable name, the array variable name, and the array index. Retrieved information is automatically deleted.

Note Once saved information is retrieved, it is automatically deleted. If that information is needed by another policy, the policy that retrieves it (using the **context_retrieve** command extension) should also save it again (using the **context_save** command extension).

Syntax

```
context_retrieve ctxt [var] [index_if_array]
```

Arguments

ctxt	(Mandatory) Context name.
var	(Optional) Scalar variable name or array variable name. Defaults to a null string if this argument is not specified.
index_if_array	(Optional) The array index.

Note The index_if_array argument will be ignored when the var argument is a scalar variable.

If var is unspecified, retrieves the whole variable table saved in the context.

If var is specified and index_if_array is not specified, or if index_if_array is specified but var is a scalar variable, retrieves the value of var.

If var is specified, and index_if_array is specified, and var is an array variable, retrieves the value of the specified array element.

Result String

Resets the Tcl global variables to the state that they were in when the save was performed.

Set _cerrno

- A string displaying _cerrno, _cerr_sub_num, _cerr_sub_err, _cerr_posix_err, _cerr_str due to appl_reqinfo error.
- Variable is not in the context.

Sample Usage

The following examples show how to use the **context_save**and **context_retrieve** command extension functionality to save and retrieve data. The examples are shown in save and retrieve pairs.

Example 1: Save

If var is unspecified or if a pattern if specified, saves multiple variables to the context.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvara 123
set testvarb 345
set testvarc 789
if {[catch {context_save TESTCTX "testvar*"} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 1: Retrieve

If var is unspecified, retrieves multiple variables from the context.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

if {[catch {foreach {var value} [context_retrieve TESTCTX] {set $var $value}} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvara]} {
      action_syslog msg "testvara exists and is $testvara"
} else {
      action_syslog msg "testvara does not exist"
}
if {[info exists testvarb]} {
      action_syslog msg "testvarb exists and is $testvarb"
```

```
} else {
      action_syslog msg "testvarb does not exist"
}
if {[info exists testvarc]} {
      action_syslog msg "testvarc exists and is $testvarc"
} else {
      action_syslog msg "testvarc does not exist"
}
```

Example 2: Save

If var is specified, saves the value of var.

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvar 123
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 2: Retrieve

If var is specified and index_if_array is not specified, or if index_if_array is specified but var is a scalar variable, retrieves the value of var.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar does not exist"
}
```

Example 3: Save

If var is specified, saves the value of var even if it is an array.

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 3: Retrieve

If var is specified, and index_if_array is not specified, and var is an array variable, retrieves the entire array.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {array set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is [array get testvar]"
} else {
      action_syslog msg "testvar does not exist"
}
```

Example 4: Save

If var is specified, saves the value of var even if it is an array.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 4: Retrieve

If var is specified, and index_if_array is specified, and var is an array variable, retrieves the specified array element value.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar testvar1]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar doesn't exist"
}
```

context_save

Saves Tcl variables that match a given pattern in current and global namespaces with the given context name as identification. Use this Tcl command extension to save information outside of a policy. Saved information can be retrieved by a different policy using the **context_retrieve** command extension.

Note Once saved information is retrieved, it is automatically deleted. If that information is needed by another policy, the policy that retrieves it (using the **context_retrieve** command extension) should also save it again (using the **context_save** command extension).

Syntax

```
context_save ctxt [pattern]
```

Arguments

ctxt	(Mandatory) Context name.
pattern	(Optional) The glob-style pattern as used by the **string match**Tcl command. If this argument is not specified, the pattern defaults to the wildcard *. There are three constructs used in glob patterns: • * = all characters • ? = 1 character • [abc] = match one of a set of characters

Result String

None

Set _cerrno

A string displaying _cerrno, _cerr_sub_num, _cerr_sub_err, _cerr_posix_err, _cerr_str due to appl_setinfo error.

Sample Usage

The following examples show how to use the **context_save**and **context_retrieve** command extension functionality to save and retrieve data. The examples are shown in save and retrieve pairs.

Example 1: Save

If var is unspecified or if a pattern if specified, saves multiple variables to the context.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvara 123
set testvarb 345
set testvarc 789
if {[catch {context_save TESTCTX "testvar*"} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 1: Retrieve

If var is unspecified, retrieves multiple variables from the context.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

if {[catch {foreach {var value} [context_retrieve TESTCTX] {set $var $value}} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvara]} {
      action_syslog msg "testvara exists and is $testvara"
} else {
      action_syslog msg "testvara does not exist"
}
if {[info exists testvarb]} {
      action_syslog msg "testvarb exists and is $testvarb"
} else {
      action_syslog msg "testvarb does not exist"
}
if {[info exists testvarc]} {
      action_syslog msg "testvarc exists and is $testvarc"
} else {
      action_syslog msg "testvarc does not exist"
}
```

Example 2: Save

If var is specified, saves the value of var.

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvar 123
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 2: Retrieve

If var is specified and index_if_array is not specified, or if index_if_array is specified but var is a scalar variable, retrieves the value of var.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
```

```
      action_syslog msg "testvar does not exist"
}
```

Example 3: Save

If var is specified, saves the value of var even if it is an array.

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 3: Retrieve

If var is specified, and index_if_array is not specified, and var is an array variable, retrieves the entire array.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {array set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is [array get testvar]"
} else {
      action_syslog msg "testvar does not exist"
}
```

Example 4: Save

If var is specified, saves the value of var even if it is an array.

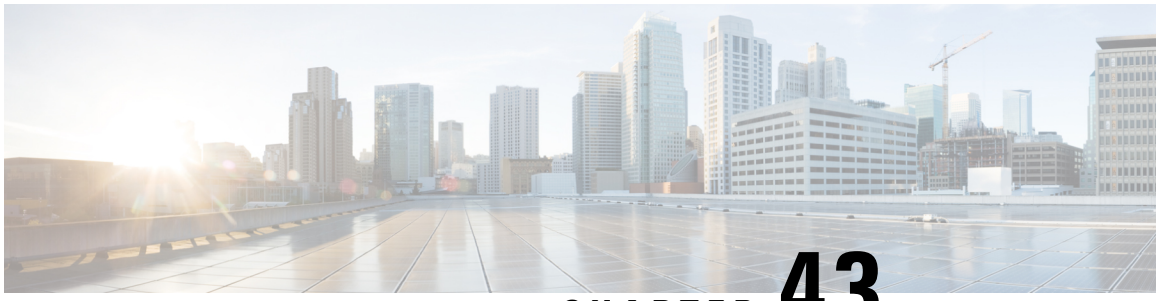
```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

Example 4: Retrieve

If var is specified, and index_if_array is specified, and var is an array variable, retrieves the specified array element value.

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar testvar1]} errmsg]} {
```

```
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar doesn't exist"
}
```



CHAPTER 43

EEM Event Registration Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

event_register_appl

Registers for an application event. Use this Tcl command extension to run a policy when an application event is triggered following another policy's execution of an **event_publish** Tcl command extension; the **event_publish** command extension publishes an application event.

In order to register for an application event, a subsystem must be specified. Either a Tcl policy or the internal Embedded Event Manager (EEM) API can publish an application event. If the event is being published by a policy, the sub_system argument that is reserved for a policy is 798.

Syntax

```
event_register_appl [tag ?] sub_system ? type ? [queue_priority low|normal|high|last] [maxrun
 ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sub_system	(Mandatory) Number assigned to the EEM policy that published the application event. The number is set to 798 because all other numbers are reserved for Cisco use. If this argument is not specified, all components are matched.
type	(Mandatory) Event subtype within the specified event. The sub_system and type arguments uniquely identify an application event. If this argument is not specified, all types are matched. If you specify this argument, you must choose an integer between 1 and 4294967295, inclusive. There must be a match of component and type between the **event_publish** command extension and the **event_register_appl**command extension in order for the publishing and registration to work.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If multiple conditions exist, the application event will be raised when all the conditions are satisfied.

Result String

None

Set _cerrno

No

Event_reqinfo

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"sub_system 0x%x type %u data1 {%s} data2 {%s} data3 {%s} data4 {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, when the event was published to the Embedded Event Manager (EEM).
sub_system	Number assigned to the EEM policy that published the application event. Number is set to 798 because all other numbers are reserved for Cisco use.
type	Event subtype within the specified component.
data1 data2 data3 data4	Argument data that is passed to the application-specific event when the event is published. The data is character text, an environment variable, or a combination of the two.

event_register_cli

Registers for a CLI event. Use this Tcl command extension to run a policy when a CLI command of a specific pattern is entered based on pattern matching performed against an expanded CLI command.

Note The user can enter an abbreviated CLI command, such as **sh mem summary**, and the parser will expand the command to **show memory summary**to perform the matching.

Note The functionality provided in the CLI event detector only allows a regular expression pattern match on a valid IOS CLI command itself. This does not include text after a pipe character when redirection is used.

Syntax

```
event_register_cli [tag ?] sync yes|no skip yes|no
[occurs ?] [period ?] pattern ? [default ?] [enter] [questionmark] [tab] [mode]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sync	(Mandatory) A "yes" means that the policy (the event publish) will run synchronously with the CLI command; a "no" means that the event publish will be performed asynchronously with the CLI command. The event detector will be notified when the policy completes running. The exit status of the policy indicates whether or not the CLI command should be executed: if the exit status is zero, which means that the policy is executed successfully, the CLI command will not be executed; otherwise, the CLI command will be executed.

skip	Mandatory if the sync argument is "no" and should not exist if the sync argument is "yes." If the skip argument is "yes," it means that the CLI command should not be executed. If the skip argument is "no," it means that the CLI command should be executed. **Caution** When the skip argument is "yes," unintended results may be produced if the pattern match is made for configuration commands because the CLI command that matches the regular expression will not be executed.
occurs	(Optional) The number of occurrences before the event is raised. If this argument is not specified, the event is raised on the first occurrence. If this argument is specified, it must be an integer between 1 and 4294967295, inclusive.
period	(Optional) Specifies a backward looking time window in which all CLI events must occur (the occurs clause must be satisfied) in order for an event to be published (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent event is used.
pattern	(Mandatory) Specifies the regular expression used to perform the CLI command pattern match.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
enter	(Optional) Specifies to perform the event match when the user presses the Enter key. When this parameter is used, the input string will not be expanded before matching.
questionmark	(Optional) Specifies to perform the event match when the user presses the ? key. When this parameter is used, the input string will not be expanded before matching.

tab	(Optional) Specifies to perform the event match when the user presses the Tab key. When this parameter is used, the input string will not be expanded before matching.
mode	(Optional) Events will only be generated when the parser is in the specified parser mode. The available modes can be listed using the **show parser dump** CLI command. The mode parameter is checked when any one of the optional parameters--enter, questionmark, or tab--is specified.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If multiple conditions are specified, the CLI event will be raised when all the conditions are matched.

Result String

None

Set _cerrno

No

Note This policy runs before the CLI command is executed. For example, suppose policy_CLI is registered to run when the **copy** command is entered. When the **copy** command is entered, the CLI event detector finds a pattern match and triggers this policy to run. When the policy execution ends, the CLI event detector determines if the **copy** command needs to be executed according to "sync", "skip" (set in the policy), and the exit status of the policy execution if needed.

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u msg {%s} msg_count %d line %u key %u tty %u error_code %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.

Event Type	Description
msg	Text entered at the CLI prompt.
msg_count	Number of times the pattern matched before the event was triggered.
line	The text the parser was able to expand up to the point where the matched key was entered.
key	The enter, questionmark, or tab key.
tty	Corresponds to the line number the user is executing the command on.
error_code	The error code in CLI. 0 --No error from parser up to point where a key was entered. 1--Command is ambiguous up to point where a key was entered. 4--Unknown command up to point where a key was entered.

event_register_counter

Registers for a counter event as both a publisher and a subscriber. Use this Tcl command extension to run a policy on the basis of a named counter crossing a threshold. This event counter, as a subscriber, identifies the name of the counter to which it wants to subscribe and depends on another policy or another process to actually manipulate the counter. For example, let policyB act as a counter policy, whereas policyA (although it does not need to be a counter policy) uses **register_counter**, **counter_modify**, or **unregister_counter** Tcl command extensions to manipulate the counter defined in policyB.

Syntax

```
event_register_counter [tag ?] name ? entry_op gt|ge|eq|ne|lt|le entry_val ?
exit_op gt|ge|eq|ne|lt|le exit_val ? [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
name	(Mandatory) Name of the counter.
entry_op	(Mandatory) Entry comparison operator used to compare the current counter value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.
entry_val	(Mandatory) Value with which the current counter value should be compared to decide if the counter event should be raised.
exit_op	(Mandatory) Exit comparison operator used to compare the current counter value with the exit value; if true, event monitoring for this event will be reenabled.

exit_val	(Mandatory) Value with which the current counter value should be compared to decide if the exit criteria are met.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"name {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.

Event Type	Description
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
name	Counter name.

event_register_gold

Registers for a Generic Online Diagnostic (GOLD) failure event. Use this Tcl command extension to run a policy on the basis of a Generic Online Diagnostic (GOLD) failure event for the specified card and subcard.

Syntax

```
event_register_gold card all|card_number
[subcard all|subcard_number]
[new_failure TRUE|FALSE]
[severity_major TRUE]
[severity_minor TRUE]
[severity_normal TRUE]
[action_notify TRUE|FALSE]
[testing_type [bootup|ondemand|schedule|monitoring]]
[test_name [testname]]
[test_id [testnumber]]
[consecutive_failure consecutive_failure_number]
[platform_action [action_flag]]
[maxrun ?]
[queue_priority low|normal|high|last]
[nice 0|1]
```

Arguments

card	(Mandatory) Specifies whether all cards or one card is to be monitored: • card all--Specifies that all cards are to be monitored. This is the default. • card-number--Specifies that the card identified by the number card-number is to be monitored. This argument must be specified to complete the **event_register_gold**Tcl command extension.
subcard	(Optional) Specifies that one or more subcards are to be monitored: • subcard all--Specifies that all subcards are to be monitored. • subcard-number--Specifies that the subcard identified by the number subcard-number is to be monitored. If this argument is not specified, all subcards are monitored by default.

new_failure	(Optional) Specifies event criteria based on the new test failure information from GOLD: • new_failure TRUE--Specifies that the event criterion for the new test failure is true from GOLD. • new_failure FALSE--Specifies that the event criterion for the new test failure is false from GOLD. If this argument is not specified, the new test failure information from GOLD is not considered in the event criteria.
severity_major	(Optional) Specifies that the event criteria for diagnostic result matches with the diagnostic major error from GOLD.
severity_minor	(Optional) Specifies that the event criteria for diagnostic result matches with diagnostic minor error from GOLD.
severity_normal	(Optional) Specifies that the event criteria for diagnostic result matches with diagnostic normal from GOLD. This is the default.
action_notify	(Optional) Specifies the event criteria based on the action notify information from GOLD: • action_notify TRUE--Specifies that the event criterion for the action notify is true from GOLD. • action_notify FALSE--Specifies that the event criterion for the action notify is false from GOLD. If this argument is not specified, the action notify information from GOLD is not considered in the event criteria.
testing_type	(Optional) Specifies the event criteria based on the testing types of the diagnostic from GOLD: • testing_type bootup--Specifies the diagnostic tests that are running on system bootup. • testing_type ondemand--Specifies the diagnostic tests that are running from CLI after the card is online. • testing_type schedule--Specifies the scheduled diagnostic tests. • testing_type monitoring--Specifies the diagnostic tests that are running periodically in the background to monitor the health of the system. If this argument is not specified, the testing type information from GOLD is not considered in the event criteria and the policy applies to all the diagnostic testing types.
test_name	(Optional) Specifies the event criteria based on the test name: • test_name test-name--Specifies the event criteria based on the test with the name test-name. If this argument is not specified, the test name information from GOLD is not considered in the event criteria.

test_id	(Optional) Specifies the event criteria based on test ID: • test_id test-id--Specifies the event criteria based on the test with the ID number test-id. The maximum value of test-id is 65535. **Note** Because the test ID can be different for the same test on different line cards, usually the test_name keyword should be used instead. If the test ID is specified and conflicts with the specified test name, the test name overwrites the test ID. If this argument is not specified, test ID information from GOLD is not considered in the event criteria.
consecutive_failure	(Optional) Specifies the event criteria based on consecutive test failure information from GOLD: • consecutive_failure consecutive-failure-number--Specifies that the event criterion is based on the occurrence of consecutive-failure-number consecutive test failures. If this argument is not specified, consecutive test failure information from GOLD is not considered in the event criteria.
platform_action	(Optional) Specifies whether callback to the platform is needed when all the event criteria are matched. When callback is needed, the platform needs to register a callback function through the provided registry. • platform_action action-flag-number--Specifies that, when callback to the platform is needed, specific information is specified by the platform-specific action-flag-number value. The maximum value of action-flag-number is 65535. **Note** It is up to the platform to determine what action needs to be taken based on the flag. If this argument is not specified, there is no callback.
maxrun	(Optional) Specifies the maximum runt time of the script. • maxrun max-run-time-number--Specifies that the maximum run time of the script is max-run-time-number seconds. The maximum value of max-run-time-number is 4294967295 seconds. If this argument is not specified, the default run time is 20 seconds.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting: • nice 0--Specifies that the policy is run at the default run-time priority level. • nice 1--Specifies that the policy is run at a run-time priority that is less than the default priority. If this argument is not specified, the default run-time priority is used.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u card %u sub_card %u"
"event_severity {%s} event_pub_sec %u event_pub_msec %u overall_result %u"
"new_failure {%s} action_notify {%s} tt %u tc %u bl %u ci %u pc %u cn {%s}"
"sn {%s} tn# {%s} ta# %s ec# {%s} rc# %u lf# {%s} tf# %u cf# %u tr# {%s}"
"tr#p# {%s} tr#d# {%s}"
```

Event Type	Description
action_notify	Action notify information in GOLD event: true or false.

Event Type	Description
bl	The boot-up diagnostic level, which can be one of the following values: • 0: complete diagnostic • 1: minimal diagnostics • 2: bypass diagnostic
card	Card information for the GOLD event.
cf *testnum*	Consecutive failure, where *testnum* is the test number. For example, **cf3** is the EEM built-in environment variable for consecutive failure of test 3.
ci	Card index.
cn	Card name.
ec *testnum*	Test error code, where *testnum* is the test number. For example, **ec3** is the EEM built-in environment variable for the error code of test 3.
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_pub_msec **event_pub_sec**	The time, in milliseconds and seconds, when the event was published to the EEM.
event_severity	GOLD event severity, which can be one of the following values: • normal • minor • major.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
lf *testnum*	Last fail time, where *testnum* is the test number. For example, **lf3** is the EEM built-in variable for the last fail time of test 3. The timestamp format is *mmm dd yyyy hh*:*mm*:*ss*. For example, Mar 11 1960 08:47:00.
new_failure	The new test failure information in a GOLD event flag: true or false.
overall_result	The overall diagnostic result, which can be one of the following values: • 0: OK • 3: minor error • 4: major error • 14: unknown result

Event Type	Description
pc	Port counts.
rc *testnum*	Test total run count, where *testnum* is the test number. For example, **rc3** is the EEM built-in variable for the total run count of test 3.
sn	Card serial number.
sub_card	The subcard on which a GOLD failure event was detected.
ta *testnum*	Test attribute, where *testnum* is the test number. For example, **ta3** is the EEM built-in variable for the test attribute of test 3.
tc	Test counts.
tf *testnum*	Total failure count, where *testnum* is the test number. For example, **tf3** is the EEM built-in variable for the total failure count of test 3.
tn *testnum*	Test name, where *testnum* is the test number. For example, **tn3** is the EEM built-in variable for the name of test 3.
tr *testnum*	Test result, where *testnum* is the test number. For example, **tr6** is the EEM built-in variable for test 6 where test 6 is not a per-port test and not a per-device test. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
tr *testnum* **d** *devnum*	Per-device test result, where *testnum* is the test number and *devnum* is the device number. For example, **tr3d20** is the EEM built-in variable for the test result for test 3, device 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
tr *testnum* **p** *portnum*	Per-port test result, where *testnum* is the test number and *portnum* is the device number. For example, **tr5p20** is the EEM built-in variable for the test result for test 3, port 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown

Event Type	Description
tt	The testing type, which can be one of the following: • 1: A boot-up diagnostic • 2: An on-demand diagnostic • 3: A schedule diagnostic • 4: A monitoring diagnostic

event_register_identity

Registers for an identity event. Use this Tcl command extension to generate an event when AAA authentication or authorization is successful or failure or after normal user traffic on the port is allowed to flow.

Syntax

```
event_register_identity [tag ?] interface ?
[aaa-attribute ?]
[authc {all | fail | success}]
[authz {all | fail | success}]
[authz-complete]
[mac-address ?]
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.
aaa-attribute	(Optional) A regular expression that can be used to filter events by specific AAA attributes.
authc	(Optional) Triggers events on successful, failed or both successful and failed authentication.
authz	(Optional) Triggers events on successful, failed or both successful and failed authorization.
authz-complete	(Optional) Triggers events once the device connected to the interface is fully authenticated, authorized and normal traffic has begun to flow on that interface.
mac-address	(Optional) A regular expression pattern that can be used to filter events by mac addresses of the remote device.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_IDENTITY

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u identity_stage %u identity_status %u interface %u identity_mac %u
identity_<attribute> {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.

Event Type	Description
identity_stage	One among authentication, autherization or autherization-complete stages.
identity_status	Success or one of these failure types: fail_authc, fail_aaa_server, fail_no_response, fail_timeout, fail_authz. For autherization-complete it is always success.
interface	The interface for the event.
identity_mac	The MAC address of the remote device for the event.
identity_<attribute>	For each AAA attribute, a set a dynamic variable to the value corresponding to that AAA attribute in the attribute or value list.

event_register_interface

Registers for an interface counter event. Use this Tcl command extension to generate an event when specified interface counters exceed specified thresholds.

Syntax

```
event_register_interface [tag ?] name ?
parameter ? entry_op gt|ge|eq|ne|lt|le
entry_val ? entry_val_is_increment TRUE|FALSE
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le]
[exit_val ?] [exit_val_is_increment TRUE|FALSE]
[exit_type value|increment|rate]
[exit_time ?] [poll_interval ?]
[average_factor ?] [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
name	(Mandatory) The name of the interface being monitored, for example, Ethernet 0/0. Abbreviations and spaces are not allowed.

parameter	(Mandatory) The name of the counter being compared as follows: • input_errors--Includes runts, giants, no buffer, CRC, frame, overrun, and ignored counts. Other input-related errors can also cause the input errors count to be increased, and some datagrams may have more than one error; therefore, this sum may not balance with the sum of enumerated input error counts. • input_errors_crc--Cyclic redundancy checksum generated by the originating LAN station or far-end device does not match the checksum calculated from the data received. • input_errors_frame--Number of packets received incorrectly having a CRC error and a noninteger number of octets. • input_errors_overrun--Number of times the receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the receiver's ability to handle the data. • input_packets_dropped--Number of packets dropped because of a full input queue. • interface_resets--Number of times that an interface has been completely reset. • output_buffer_failures--Number of failed buffers and number of buffers swapped out. • output_buffer_swappedout--Number of packets swapped to DRAM.

parameter (continued)	• output_errors--Sum of all errors that prevented the final transmission of datagrams out of the interface being examined. Note that this may not balance with the sum of the enumerated output errors, because some datagrams may have more than one error, and others may have errors that do not fall into any of the specifically tabulated categories. • output_errors_underrun--Number of times that the transmitter has been running faster than the device can handle. • output_packets_dropped--Number of packets dropped because of a full output queue. • receive_broadcasts--Number of broadcast or multicast packets received by the interface. • receive_giants--Number of packets that are discarded because they exceed the maximum packet size of the medium. • receive_rate_bps--Interface receive rate in bytes per second. • receive_rate_pps--Interface receive rate in packets per second. • receive_runts--Number of packets that are discarded because they are smaller than the minimum packet size of the medium. • receive_throttle--Number of times that the receiver on the port was disabled, possibly because of buffer or processor overload. • reliability--Reliability of the interface as a fraction of 255 (255/255 is 100 percent reliability), calculated as an exponential average over 5 minutes. • rxload--Receive rate of the interface as a fraction of 255 (255/255 is 100 percent). • transmit_rate_bps--Interface transmit rate in bytes per second. • transmit_rate_pps--Interface transmit rate in packets per second. • txload--Transmit rate of the interface as a fraction of 255 (255/255 is 100 percent).
entry_op	(Mandatory) The comparison operator used to compare the current interface value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.
entry_val	(Mandatory) The value at which the event will be triggered.
entry_val_is_increment	(Mandatory) If TRUE, the entry_val field is treated as an incremental difference and is compared with the difference between the current counter value and the value when the event was last true (the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. If FALSE, the entry_val field is compared against the current counter value. **Note** This keyword has been deprecated, and if specified, the syntax is converted into equivalent entry-type keyword syntax.

entry-type	Specifies a type of operation to be applied to the object ID specified by the entry-val argument. Value is defined as the actual value of the entry-val argument. Increment uses the entry-val field as an incremental difference and the entry-valis compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_comb	(Optional) Used to indicate the combination of exit condition tests required to rearm the event trigger; if the and operator is specified, both exit value and exit time tests must be true to cause rearm; if the or operator is specified, either exit value or exit time tests can be true to cause event monitoring to be rearmed.
exit_op	(Optional) The comparison operator used to compare the current interface value with the exit value; if true, event monitoring for this event will be reenabled.
exit_val	(Optional) The value at which the event is rearmed to be monitored again.
exit_val_is_increment	(Optional) If TRUE, the exit_val field is treated as an incremental difference and is compared with the difference between the current counter value and the value when the event was last true. A negative value checks the incremental difference for a counter that is decreasing. If FALSE, the exit_val field is compared against the current counter value. **Note** In Cisco IOS Release 12.4(20)T, this keyword is deprecated, and if specified, the syntax is converted into equivalent exit-type keyword syntax.
exit-type	(Optional) Specifies a type of operation to be applied to the object ID specified by the exit-val argument. If not specified, the value is assumed. Value is defined as the actual value of the exit-val argument. Increment uses the exit-val field as an incremental difference and the exit-val is compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_time	(Optional) The time period at which the event is rearmed to be monitored again (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).

poll_interval	(Optional) The frequency used to collect the samples (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 60 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). The poll interval value must not be less than 1 second. The default is 1 second.
average-factor	(Optional) Number in the range from 1 to 64 used to calculate the period used for rate-based calculations. The average-factor value is multiplied by the poll-interval value to derive the period in milliseconds. The minimum average factor value is 1.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} name {%s} parameter {%s} value %d"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	Interface event severity, which can be one of the following values: • normal • minor • major
name	Name of the interface.
parameter	Name of the parameter.
value	The incremental/decremental difference compared to the last event triggered or the absolute value of the parameter being monitored, depending on the specified value of entry_val_is_increment.

event_register_ioswdsysmon

Registers for an IOSWDSysMon event. Use this Tcl command extension to generate an event when a Cisco IOS task exceeds specific CPU utilization or memory thresholds. A Cisco IOS task is called a Cisco IOS process in native Cisco IOS.

Syntax

```
event_register_ioswdsysmon [tag ?] [timewin ?] [sub12op and|or] [sub1 ?] [sub2 ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
timewin	(Optional) Defines the time window within which all of the subevents must occur in order for an event to be generated (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).
sub12_op	(Optional) The combination operator for comparison between subevent 1 and subevent 2.

sub1	(Optional) The subevent 1 specification.
sub2	(Optional) The subevent 2 specification.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevent Syntax

```
cpu_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [period ?]
mem_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [is_percent TRUE|FALSE] [period ?]
```

Subevent Arguments

cpu_proc	(Mandatory) Specifies the use of a sample collection of CPU statistics.
path	(Mandatory) Software Modularity images only. The pathname of the POSIX process that contains the Cisco IOS scheduler to be monitored. For example, /sbin/cdp2.iosproc.
taskname	(Mandatory) The name of the Cisco IOS task to be monitored.
op	(Mandatory) The comparison operator used to compare the collected usage sample with the specified value; if true, an event will be raised.
val	(Mandatory) The value to be compared.

period	(Optional) The elapsed time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.
mem_proc	(Mandatory) Specifies the use of a sample collection of memory statistics.
is_percent	(Optional) Whether the specified value is a percentage.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
num_subs	Number of subevents.

Where the subevent info string is for a CPU_UTIL subevent,

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of subevent.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent.
taskname	Cisco IOS task name for this subevent.

Subevent Type	Description
taskid	Cisco IOS task ID for this subevent.
value	Actual average CPU utilization over the measured interval.
sec , **msec**	Elapsed time period for this measured interval.

Where the subevent info string is for a MEM_UTIL subevent,

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u is_percent %s value %u diff %d"
"sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of subevent.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent.
taskname	Cisco IOS task name for this subevent.
taskid	Cisco IOS task ID for this subevent.
is_percent	TRUE or FALSE depending on whether the value is a percentage value.
value	Total memory use in KB or the actual average memory utilization for this measured interval.
diff	The percentage difference between the oldest sample in the measured interval and the latest sample; a negative value represents a decrease.
sec , **msec**	Elapsed time period for this measured interval.

event_register_ipsla

Registers for an event that is triggered by the **event ipsla**command. Use this Tcl command to publish an event when an IPSLA reaction is triggered. The group ID or the operation ID is required to register the event.

Syntax

```
event_register_ipsla [tag ?] group_name ? operation_id ? [reaction_type ?]
[dest_ip_addr ?][queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
group_name	(Mandatory) Specifies the IP SLAs group name.

operation_id	(Mandatory) Specifies the IP SLA operation ID. Number must be in the range from 1 to 2147483647.
reaction_type	(Optional) Specifies the reaction to be taken for the specified IP SLAs operation. Type of IP SLAs reaction--One of the following keywords can be specified: **connectionLoss**, **icpif**, **jitterAvg**, **jitterDSAvg**, **jitterSDAvg**, **maxOfNegativeDS**, **maxOfNegativeSD**, **maxOfPositiveDS**, **maxOfPositiveSD**, **mos**, **packetLateArrival**, **packetLossDS**, **packetLossSD**, **packetMIA**, **packetOutOfSequence**, **rtt**, **timeout**or **verifyError** can be specified. Type of IP SLAs reaction. One of the following keywords can be specified: • connectionLoss • icpif • jitterAvg • jitterDSAvg • jitterSDAvg • maxOfNegativeDS • maxOfNegativeSD • maxOfPositiveDS • maxOfPositiveSD • mos • packetLateArrival • packetLossDS • packetLossSD • packetMIA • packetOutOfSequence • rtt • timeout • verifyError
dest_ip_address	(Optional) Specifies the destination IP address of the destination port for which the IP SLAs events are monitored.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

"event_ID %u event_type %u event_pub_sec %u event_pub_msec %u event_severity %u" "group_name %u operation_id %u condition %u reaction_type %u dest_ip_addr %u" "threshold_rising %u threshold_falling%u measured_threshold_value %u" "threshold_count1 %u threshold count2 %u"

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	The type of event to monitor for the create, update, and delete flow.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
group_name	The name of theIPSLA group.
operation_id	The IPSLA operation ID.
condition	The condition of IPSLA, which can be one of the following: • cleared • occurred
reaction_type	The IPSLA reaction type.
dest_ip_address	The IPSLA destination IP address.
threshold rising	The IPSLA configured rising threshold value.
threshold falling	The IPSLA configured falling threshold value.
measured_threshold_value	The measured threshold value of the IPSLA operation.
threshold_count1	Corresponds to the argument of the threshold type1.
threshold_count2	Corresponds to the argument of the threshold type2.

event_register_mat

Registers for a MAT event. Use this Tcl command extension to generate an event when a mac-address is learned in the mac-address-table.

Syntax

```
event_register_identity [tag ?] interface ?
[mac-address ?]
[type {add | delete}]
[hold-down ?]
[maxrun ?]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.
mac-address	Mandatory if the interface parameter is not specified. A regular expression pattern that can be used to filter events by mac addresses of the remote device.

type	(Optional) Filter based on a mac-address-table event type of add or delete. If not specified, the event type is not used in determining whether the event should be triggered.
hold-down	(Optional) When a mac-address-table event comes in, the hold-down timer can be set to make the event to wait between 1 and 4294967295 seconds before processing the policy. If not set then the policy is not delayed in being processed.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_MAT

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u notification %u intf_name %u mac_address {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.
notification	Notification type--add or delete.
intf_name	The interface name for the address table entry.
mac_address	The mac-address for the address table entry.

event_register_neighbor_discovery

Registers for a neighbor discover event. Use this Tcl command extension to generate an event when a Cisco Discovery Protocol (CDP) or Link Layer Discovery Protocol (LLDP) cache entry or a interface link status changes.

Syntax

```
event_register_neighbor_discovery [tag ?] interface ?
[cdp {add | update | delete | all}]
[lldp {add | update | delete | all}]
[link-event]
[line-event]
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.
cdp	Trigger an event when a matching CDP event occurs. One of the following options should be specified. • add--Trigger events only when a new CDP cache entry is created in the CDP table. • all--Trigger an event when a CDP cache entry is added or deleted from the CDP cache table and when a remote CDP device sends a keepalive to update the CDP cache entry. • delete--trigger events only when a CDP cache entry is deleted from the CDP table. • update--trigger an event when a CDP cache entry is added to the CDP table or when the remote CDP device sends a CDP keepalive to update the CDP cache entry.
lldp	Trigger an event when a matching lldp event occurs. One of the following options should be specified. • add--Trigger events only when a new cdp cache entry is created in the cdp table. • all--Trigger an event when a cdp cache entry is added or deleted from the cdp cache table and when a remote cdp device sends a keepalive to update the cdp cache entry. • delete--trigger events only when a cdp cache entry is deleted from the cdp table. • update--trigger an event when a cdp cache entry is added to the cdp table or when the remote cdp device sends a cdp keepalive to update the cdp cache entry.
line-event	Trigger an event when the interface line protocol status changes.
link-event	Trigger an event when the interface link status changes.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_NEIGHBOR_DISCOVERY

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u nd_notification {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.

Event Type	Description
event_severity	The severity of the event.
Common Event_Reqinfo	
nd_notification	The type of notification--cdp-add, cdp-update, cdp-delete, lldp-add, lldp-update, lldp-delete, link, line.
nd_intf_linkstatus	The current interface link status, up or down.
nd_intf_linestatus	The current interface line status, down, goingdown, init, testing, up, reset, admindown, deleted.
nd_local_intf_name	The local interface name for the event.
nd_short_local_intf_name	The short name of the local interface for the event.
nd_port_id	The port id as identified by either the cdp or lldp protocol. This is not set for link or line protocol events.
CDP-specific Event_reqinfo	
nd_protocol	Identifies which protocol triggered the event, for CDP it will always be set to cdp.
nd_proto_notif	Identifies which type of protocol event triggered the event, add, update or delete.
nd_proto_new_entry	If set to 1, the event was triggered because the cache entry is new, otherwise it will be set to 0.
nd_cdp_entry_name	The name of the cdp cache entry in the cdp table.
nd_cdp_hold_time	The time remaining until the cdp cache entry expires and is deleted from the cdp table. This time will be reset to some maximum by an update from the cdp neighbor. It is usually set to 0 for new entries.
nd_cdp_mgmt_domain	The CDP VTP management domain.
nd_cdp_platform	The platform name reported by the remote device.
nd_cdp_version	The version of code running on the remote device.
nd_cdp_capabilities_string	The contents of the CDP capabilities field in a string format: Router, Trans-Bridge, Source-Route-Bridge, Switch, Host, IGMP, Repeater, Phone, Remotely-Managed device, CVTA phone port, Two-port Mac Relay or any combination of these separated by commas.
nd_cdp_capabilities_bits	The CDP capabilities bits in a hexadecimal number preceded with 0x.
nd_cdp_capabilities_bit_[0-31]	A series of values that will be set to YES if that bit in the capabilities field is set or NO if it is not set.

Event Type	Description
LLDP-specific Event_reqinfo	
nd_protocol	Identifies which protocol triggered the event, for LLDP it will always be set to lldp.
nd_proto_notif	Identifies which type of protocol event triggered the event, add, update or delete.
nd_proto_new_entry	If set to 1, the event was triggered because the cache entry is new, otherwise it will be set to 0.
nd_lldp_chassis_id	The chassis id field from the LLDP cache entry.
nd_lldp_system_name	The system name from the LLDP cache entry.
nd_lldp_system_description	The system description field from the LLDP cache entry.
nd_lldp_ttl	The LLDP time to live field from the LLDP cache entry.
nd_lldp_port_description	The port description field from the LLDP cache entry.
nd_lldp_system_capabilities_string	The LLDP system capabilities field from the LLDP cache entry. Provided as a string that can contain O, P, B, W, R, T, C, S or any combination of these separated by commas.
nd_lldp_enabled_capabilities_string	The LLDP enabled system capabilities field from the LLDP cache entry. Provided as a string that can contain O, P, B, W, R, T, C, S or any combination of these separated by commas.
nd_lldp_system_capabilities_bits	The LLDP system capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_enabled_capabilities_bits	The LLDP enabled capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_capabilities_bits	The LLDP capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_capabilities_bit_[0-31]	A series of values that will be set to YES if that bit in the capabilities field is set or NO if it is not set.

event_register_nf

Registers for an event when a NetFlow event is triggered by the **event nf**command. Use this Tcl command to publish an event when an NetFlow reaction is triggered..

Syntax

```
event_register_nf [tag ?] monitor_name ? event_type create|update|delete
exit_event_type create|update|delete event1-event4 ? [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
monitor_name	(Mandatory) The name of the NetFlow monitor.
event_type	(Mandatory) The type of event to monitor for the create, update, and delete flow.
exit_event_type	(Mandatory) The event-type (create, delete, update) at which the event is rearmed to be monitored again.
event1- event4	(Mandatory) Specifies the event and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**. The subevent keywords can be used alone, together, or in any combination with each other, but each keyword can be used only once.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevent Syntax

```
field ? rate_interval ? event1 only entry_value ? entry_op eq|ge|gt|le|lt|wc
[exit_value ?] [exit_op eq|ge|gt|le|lt|wc] [exit_rate_interval ? event1 only]
```

Subevent Arguments

field	(Mandatory) Specifies the cache or field attribute to be monitored. One of the following attributes can be specified: • **counter** {**bytes** | **packets**}--Specifies the counter fields. • **datalink** {**dot1q** | **mac**}--Specifies the datalink (layer2) fields. • **flow** {**direction** | **sampler**}--Specifies the flow identifying fields. • **interface** {**input** | **output**}--Specifies the interface fields. • **ipv4** *field-type*-- Specifies the IPv4 fields. • **ipv6** *field-type*-- IPv6 fields • **routing** *routing-attrribute* -- Specifies the routing attributes. • **timestamp sysuptime** {**first | last**}--Specifies the timestamp fields. • **transport** *field-type*-- Specifies the Transport layer fields.
rate_interval	(Mandatory) Specifies the rate interval value in seconds used to calculate the rate. This field is only valid for event1.

entry_value	(Mandatory) Specifies the field or rate value.
entry_op	(Mandatory) Specifies the field operator. The comparison operator valid values are: • **eq** - Equal to • **ge** - Greater than or equal to • **gt** - Greater than • **le** - Less than or equal to • **lt** - Less than • **wc** - Wildcard
exit_value	(Optional) The value at which the event is rearmed to be monitored again.
exit_op	(Optional) The comparison operator used to compare the current event field or rate value with the exit value; if true, event monitoring for this event is reenabled. The comparison operator valid values are: • **eq** - Equal to • **ge** - Greater than or equal to • **gt** - Greater than • **le** - Less than or equal to • **lt** - Less than • **wc** - Wildcard
exit_rate_interval	(Optional) Specifies the exit rate interval value in seconds used to calculate the exit rate value. This field is only valid for event1.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_ID %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u monitor_name %u event1-event4_field %u event1-event4_value
```

Event Type	Description

event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	The type of event to monitor for the create, update, and delete flow.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the NetFlow event.
montior_name	The name of the NetFlow monitor.
event1-event4_field	Specifies the event and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**.
event1-event4_value	Specifies the event value and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**.

event_register_none

Registers for an event that is triggered by the **event manager run** command. These events are handled by the None event detector that screens for this event.

Syntax

```
event_register_none [tag ?] [sync {yes|no}] [default ?] [queue_priority low|normal|high|last]
 [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sync	(Optional) A "yes" or a "no" is required to complete this keyword. • If the yes keyword is specified, the policy will run synchronously with the CLI command. • If the no keyword is specified, the policy will run asynchronously with the CLI command.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u arg %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
argc **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14** **arg15**	The parameters that are passed from the XML SOAP command to the script.

event_register_oir

Registers for an online insertion and removal (OIR) event. Use this Tcl command extension to run a policy on the basis of an event raised when a hardware card OIR occurs. These events are handled by the OIR event detector that screens for this event.

Syntax

```
event_register_oir [tag ?] [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"slot %u event %s"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event ID.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.

Event Type	Description
slot	Slot number for the affected card.
event	Indicates a string, removed or online, that represents either an OIR removal event or an OIR insertion event.

event_register_process

Registers for a process event. Use this Tcl command extension to run a policy on the basis of an event raised when a Cisco IOS Software Modularity process starts or stops. These events are handled by the System Manager event detector that screens for this event. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
event_register_process [tag ?] abort|term|start|user_restart|user_shutdown
[sub_system ?] [version ?] [instance ?] [path ?] [node ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
abort	(Mandatory) Abnormal process termination. Process may terminate because of exiting with a nonzero exit status, receiving a kernel-generated signal, or receiving a SIGTERM or SIGKILL signal that is not sent because of user request.
term	(Mandatory) Normal process termination.
start	(Mandatory) Process start.
user_restart	(Mandatory) Process termination due to the process restart request from the CLI command.
user_shutdown	(Mandatory) Process termination due to the process kill request from the CLI command.
sub_system	(Optional) Number assigned to the EEM policy that published the process event. Number is set to 798 because all other numbers are reserved for Cisco use.
version	(Optional) Version number of the process assigned by the version manager. Must be of the form major_number.minor_number.level. If specified, each component of the version number must be an integer between 1 and 4294967295, inclusive.
instance	(Optional) Process instance ID. If specified, this argument must be an integer between 1 and 4294967295, inclusive.
path	(Optional) Process pathname (a regular expression string). If the value of the process-name argument contains embedded blanks, enclose it in double quotation marks. Use path ".*" to match all processes.

node	(Optional) The node name is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: node<slot-number>/<cpu-number> The slot-number is the hardware slot number. The cpu-number is the hardware CPU number. For example, the SP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be specified as node0/0. The RP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be addressed as node0/1. If the node argument is not specified, the default node specification is always the regular expression pattern match of * representing all applicable nodes.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If an optional argument is not specified, the event matches all possible values of the argument. If multiple arguments are specified, the process event will be raised when all the conditions are matched.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
```

```
"sub_system 0x%x instance %u process_name {%s} path {%s} exit_status 0x%x"
"respawn_count %u last_respawn_sec %ld last_respawn_msec %ld fail_count %u"
"dump_count %u node_name {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
sub_system	Number assigned to the EEM policy that published the application-specific event. Number is set to 798 because all other numbers are reserved for Cisco use.
instance	Process instance ID.
process_name	Process name.
path	Process absolute name including path.
exit_status	Process last exit status.
respawn_count	Number of times that the process was restarted.
last_respawn_sec last_respawn_msec	The calendar time when the last restart occurred.
fail_count	Number of restart attempts of the process that failed. This count will be reset to 0 when the process is successfully restarted.
dump_count	Number of core dumps taken of the process.
node_name	Name of the node that the process is on. The node name is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: **node** *slot-number* / *cpu-number* The slot-number is the hardware slot number. The cpu-number is the hardware CPU number.

event_register_resource

Registers for an Embedded Resource Manager (ERM) event. Use this Tcl command extension to run a policy on the basis of an ERM event report for a specified policy. ERM events are screened by the EEM Resource event detector, allowing an EEM policy to be run when a match occurs for the specified ERM policy.

Syntax

```
event_register_resource policy policy-name [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

policy	(Mandatory) Specifies the use of a policy.
policy-name	(Mandatory) Name of an ERM policy.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"owner_id %lld user_id %lld" time_sent %llu dampen_time %d notify_data_flags %u"
"level {%s} direction {%s} configured_threshold %u current_value %u"
"policy_violation_flag {%s} policy_id %d"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
owner_id	The Embedded Resource Manager (ERM) owner ID.
user_id	The ERM user ID.
time_sent	The ERM event time, in nanoseconds.
dampen_time	The ERM dampen time, in nanoseconds.
notify_data_flags	The ERM notify data flag.
level	The ERM event level. The four event levels are normal, minor, major, and critical.
direction	The ERM event direction. The event direction can be one of the following: up, down, or no change.
configured_threshold	The configured ERM threshold.
current_value	The current value reported by ERM.
policy_violation_flag	The ERM policy violation flag; either false or true.
policy_id	The ERM policy ID.

event_register_rf

Registers for a Redundancy Facility (RF) event. Use this Tcl command extension to run a policy when an RF progression or status event notification occurs.

Syntax

```
event_register_rf [tag ?] event ?
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
event	(Mandatory) Name of the RF progression or status event. Valid values are: • RF_PROG_ACTIVE • RF_PROG_ACTIVE_DRAIN • RF_PROG_ACTIVE_FAST = 200 • RF_PROG_ACTIVE_PRECONFIG • RF_PROG_ACTIVE_POSTCONFIG • RF_PROG_EXTRALOAD • RF_PROG_HANDBACK • RF_PROG_INITIALIZATION • RF_PROG_PLATFORM_SYNC • RF_PROG_STANDBY_BULK • RF_PROG_STANDBY_COLD • RF_PROG_STANDBY_CONFIG • RF_PROG_STANDBY_FILESYS • RF_PROG_STANDBY_HOT • RF_PROG_STANDBY_OIR_SYNC_DONE • RF_REGISTRATION_STATUS • RF_STATUS_MAINTENANCE_ENABLE • RF_STATUS_MANUAL_SWACT • RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE • RF_STATUS_PEER_COMM • RF_STATUS_PEER_PRESENCE • RF_STATUS_REDUNDANCY_MODE_CHANGE • RF_STATUS_SWACT_INHIBIT

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event	RF progression or status event notification that caused this event to be published.

event_register_routing

Registers for an event that is triggered by the **event routing** command. These events are handled by the routing event detector to publish an event when route entries change in Routing Information Base (RIB) infrastructure. Use this Tcl command extension to run a routing policy for this script. The network IP address for the route to be monitored must be specified.

Syntax

```
event_register_routing [tag ?] network ? length [ge|le|ne] [type add|remove|modify|all]
[protocol ?] [queue_priority normal|low|high|last] [maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
network	Specifies the network IP address. The network number can be any valid IP address or prefix.
length	Specifies the length of the network mask in bits. The bit mask can be a number from 0 to 32. • **ge** --(Optional) Specifies the minimum prefix length to be matched. The **ge** keyword represents greater than or equal to operator. • **le** --(Optional) Specifies the maximum prefix length to be matched. The **le** keyword represents the less than or equal to operator. • **ne** --(Optional) Specifies the prefix length not to be matched. The **ne** keyword represents not equal to operator. When **ge**, **le** and **ne** keywords are not configured, an exact match of network length is processed.
type	(Optional) Specifies the desired policy trigger. The type options are **add**, **remove**, **modify**, and **all**. The default is **all**.
protocol	(Optional) Specifies the protocol value for the network being monitored. One of the following protocols can be used: **all**, **bgp**, **connected**, **eigrp**, **isis**, **iso-igrp**, **mobile**, **odr**, **ospf**, **rip**, and **static**. The default is **all**.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u" "event_severity {%s} %u network %u mask %u protocol %u lastgateway %u distance %u" "time_sec %u time_msec %u metric %u lastinterface %u"

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
network	The network prefix in IP address format
mask	The network mask in IP address format
protocol	Type of network protocol.
type	Type of event to add, remove or modify.
lastgateway	The last known gateway.
distance	The administrative distance.
time_sec time_msec	Time of event in seconds and milliseconds, when the event was published to the EEM.
metric	Path metric.
lastinterface	The last known interface.

event_register_rpc

Registers for an event that is triggered by the EEM SSH Remote Procedure Call (RPC) command. These events are handled by the RPC event detector that screens for this event. Use this Tcl command extension to run a RPC policy for this script.

Syntax

```
event_register_rpc [queue_priority {normal | low | high | last}] [maxrun <sec.msec>] [nice
 {0 | 1}] [default <sec.msec>]
```

Arguments

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
arg %u"
```

Event Type	Description

event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
argc **arg0** **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14**	The parameters that are passed from the XML SOAP command to the script.

event_register_snmp

Registers for a Simple Network Management Protocol (SNMP) statistics event. Use this Tcl command extension to run a policy when a given counter specified by an SNMP object ID (oid) crosses a defined threshold.

Syntax

```
event_register_snmp [tag ?] oid ? get_type exact|next
entry_op gt|ge|eq|ne|lt|le entry_val ?
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le] [exit_val ?]
[exit_type value|increment|rate]
[exit_time ?] poll_interval ? [average_factor ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
oid	(Mandatory) OID number of data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
entry_op	(Mandatory) Entry comparison operator used to compare the current OID data value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.
get_type	(Mandatory) Type of SNMP get operation that needs to be applied to the OID specified. If the get_type argument is "exact," the value of the specified OID is retrieved; if the get_type argument is "next," the value of the lexicographical successor to the specified OID is retrieved.
entry_val	(Mandatory) Value with which the current oid data value should be compared to decide if the SNMP event should be raised.
entry-type	Specifies a type of operation to be applied to the object ID specified by the entry-val argument. Value is defined as the actual value of the entry-val argument. Increment uses the entry-val field as an incremental difference and the entry-valis compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_comb	(Optional) Exit combination operator used to indicate the combination of exit condition tests required to decide if the exit criteria are met so that the event monitoring can be reenabled. If it is "and," both exit value and exit time tests must be passed to meet the exit criteria. If it is "or," either exit value or exit time tests can be passed to meet the exit criteria. When exit_comb is "and," exit_op, and exit_val (exit_time) must exist. When exit_comb is "or," (exit_op and exit_val) or (exit_time) must exist.
exit_op	(Optional) Exit comparison operator used to compare the current oid data value with the exit value; if true, event monitoring for this event will be reenabled.

exit_val	(Optional) Value with which the current oid data value should be compared to decide if the exit criteria are met.
exit-type	(Optional) Specifies a type of operation to be applied to the object ID specified by the exit-val argument. If not specified, the value is assumed. Value is defined as the actual value of the exit-val argument. Increment uses the exit-val field as an incremental difference and the exit-val is compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_time	(Optional) Number of POSIX timer units after an event is raised when event monitoring will be enabled again. Specified in SSSSSSSSSS[.MMM] format where SSSSSSSSSS must be an integer number representing seconds between 0 and 4294967295, inclusive. MMM represents milliseconds and must be an integer number between 0 and 999.
poll_interval	(Mandatory) Interval between consecutive polls in POSIX timer units. Currently the interval is forced to be at least 1 second (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).
average-factor	(Optional) Number in the range from 1 to 64 used to calculate the period used for rate-based calculations. The average-factor value is multiplied by the poll-interval value to derive the period in milliseconds. The minimum average factor value is 1.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.

maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} oid {%s} val {%s} delta_val {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	SNMP event severity, which can be one of the following values: • normal • minor • major
oid	Object ID of data element, in SNMP dot notation.
val	Value of the data element.
delta_val	Delta value between the value of the policies.

event_register_snmp_notification

Registers for a Simple Network Management Protocol (SNMP) notification trap event. Use this Tcl command extension to run a policy when an SNMP trap with the specified SNMP object ID (oid) is encountered on a specific interface or address. The **snmp-server manager** CLI command must be enabled for the SNMP notifications to work using Tcl policies.

Syntax

```
event_register_snmp_notification [tag ?] oid ? oid_val ?
op {gt|ge|eq|ne|lt|le}
[maxrun ?]
[src_ip_address ?]
[dest_ip_address ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
[default ?]
[direction {incoming|outgoing}]
[msg_op {drop|send}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
oid	(Mandatory) OID number of the data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). If the specified OID ends with a dot (.), then all OIDs that start with the OID number before the dot are matched. The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
oid_val	(Mandatory) OID value with which the current OID data value should be compared to decide if the SNMP event should be raised.
op	(Mandatory) Comparison operator used to compare the current OID data value with the SNMP Protocol Data Unit (PDU) OID data value; if this is true, an event is raised.
maxrun	(Optional) Maximum run time of the script (specified in ssssssss[.mmm] format, where ssssssss must be an integer representing seconds between 0 and 31536000, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

src_ip_address	(Optional) Source IP address where the SNMP notification trap originates. The default is all; it is set to receive SNMP notification traps from all IP addresses.
dest_ip_address	(Optional) Destination IP address where the SNMP notification trap is sent. The default is all; it is set to receive SNMP traps from all destination IP addresses.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the queue_priority_last argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
default	(Optional) Specifies the time period in seconds during which the snmp notification event detector waits for the policy to exit. The time period is specified in sssssssss[.mmm] format, where sssssssss must be an integer representing seconds between 0 and 4294967295 and mmm must be an integer representing milliseconds between 0 and 999.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
direction	(Optional) The direction of the incoming or outgoing SNMP trap or inform PDU to filter. The default value is incoming.
msg_op	(Optional) The action to be taken on the SNMP PDU (drop it or send it) once the event is triggered. The default value is send.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} oid_val {%s} src_ip_addr {%s} dest_ip_addr {%s} x_x_x_x_x
(varbinds) {%s} trunc_vb_buf {%s} trap_oid {%s} enterprise_oid {%s} generic_trap %u
specific_trap %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
oid	An user specified object ID.
oid_val	An user specified object ID value.
src_ip_addr	The source IP address of the SNMP protocol data unit (PDU).
dest_ip_addr	The destination IP address of the SNMP PDU.
x_x_x_x_x (varbinds)	The SNMP PDU varbind information.
trap_oid	Indicates the trap OID value.
enterprise_oid	Indicates the enterprise OID value.
generic_trap	Indicates one of a number of generic trap types. There are seven generic trap numbers zero to six.
specific_trap	Indicates one of a number of specific trap codes.

event_register_snmp_object

Registers for a Simple Network Management Protocol (SNMP) object event. Use this Tcl command extension to replace the value when an SNMP with the specified SNMP-object ID (OID) is encountered on a specific interface or address.

Syntax

```
event_register_snmp_object oid ?
type {int|uint|counter|counter64|gauge|ipv4||oid|string}
sync {yes|no}
skip {yes|no}
[istable {yes|no}]
[default ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
```

Arguments

oid	(Mandatory) OID number of the data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). If the specified OID ends with a dot (.), then all OIDs that start with the OID number before the dot are matched. The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
type	(Mandatory) OID value type.
sync	(Mandatory) A "yes" means that the EEM policy will be notified. If the applet set_exit_status or Tcl return value is 0, then SNMP will handle the request. If the return value is 1, SNMP will use the value provided by the policy for the get request and will not process the set request. A "no" means that EEM will not be notified and SNMP will handle the request. Only one OID can be associated with a synchronous policy. However, multiple synchronous policies can be registered for the same OID.
skip	Mandatory if the sync argument is "no" and should not exist if the sync argument is "yes." If the skip argument is "yes," it means that SNMP will handle the request. If the skip argument is "no," it means that SNMP will act as if the object does not exist.
istable	(Optional) A value of "no" means the OID is scalar object, and "yes" means the OID is table object.
default	(Optional) The time period during which the SNMP Object event detector waits for the policy to exit (specified in sssssssss[.mmm] format, where sssssssss must be an integer representing seconds between 0 and 4294967295, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to process the set or get request normally by SNMP subsystem. If this argument is not specified, the default time period is set to 30 seconds.
maxrun	(Optional) Maximum run time of the script (specified in ssssssss[.mmm] format, where ssssssss must be an integer representing seconds between 0 and 31536000, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the queue_priority_last argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} request {%s} request_type {%s} value %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
oid	The ID of the SNMP object in the received get or set request.

Event Type	Description
request	The get or set request type.
request_type	The type of request (exact or next).
value	For set requests only. The value to set the object to.

event_register_syslog

Registers for a syslog event. Use this Tcl command extension to trigger a policy when a syslog message of a specific pattern is logged after a certain number of occurrences during a certain period of time.

Syntax

```
event_register_syslog [tag ?] [occurs ?] [period ?] pattern ?
[priority all|emergencies|alerts|critical|errors|warnings|notifications|
informational|debugging|0|1|2|3|4|5|6|7]
[queue_priority low|normal|high|last]
[severity_fatal] [severity_critical] [severity_major]
[severity_minor] [severity_warning] [severity_notification]
[severity_normal] [severity_debugging]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
occurs	(Optional) Number of occurrences before the event is raised; if not specified, the event is raised on the first occurrence. If specified, the value must be greater than 0.
period	(Optional) Time interval, in seconds and milliseconds, during which the one or more occurrences must take place in order to raise an event (specified in SSSSSSSSSS[.MMM] format where SSSSSSSSSS must be an integer number representing seconds between 0 and 4294967295, inclusive, and where MMM represents milliseconds and must be an integer number between 0 and 999). If this argument is not specified, no period check is applied.
pattern	(Mandatory) A regular expression used to perform syslog message pattern match. This argument is what the policy uses to identify the logged syslog message.
priority	(Optional) The message priority to be screened. If this argument is specified, only messages that are at the specified logging priority level, or lower, are screened. If this argument is not specified, the default priority is 0.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
severity_xxx	(Optional) The event severity to be screened. If this argument is specified, only messages that are at the specified severity level are screened. See the table titled "Severity Level Mapping For Syslog Events" for the severity level mapping for syslog events.

If multiple conditions are specified, the syslog event will be raised when all the conditions are matched.

Table 75: Severity Level Mapping For Syslog Events

Severity Keyword	Syslog Priority	Description
severity_fatal	LOG_EMERG (0)	System is unusable.
severity_critical	LOG_ALERT (1)	Critical conditions, immediate attention required.
severity_major	LOG_CRIT (2)	Major conditions.
severity_minor	LOG_ERR (3)	Minor conditions.
severity_warning	LOG_WARNING (4)	Warning conditions.
severity_notification	LOG_NOTICE (5)	Basic notification, informational messages.
severity_normal	LOG_INFO (6)	Normal event, indicates returning to a normal state.
severity_debugging	LOG_DEBUG (7)	Debugging messages.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"msg {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
msg	The last syslog message that matches the pattern.

event_register_timer

Creates a timer and registers for a timer event as both a publisher and a subscriber. Use this Tcl command extension when there is a need to trigger a policy that is time specific or timer based. This event timer is both an event publisher and a subscriber. The publisher part indicates the conditions under which the named timer is to go off. The subscriber part identifies the name of the timer to which the event is subscribing.

Note Both the CRON and absolute time specifications work on local time.

Syntax

```
event_register_timer [tag ?] watchdog|countdown|absolute|cron
[name ?] [cron_entry ?]
[time ?]
[queue_priority low|normal|high|last] [maxrun ?]
[nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
watchdog	(Mandatory) Watchdog timer.
countdown	(Mandatory) Countdown timer.
absolute	(Mandatory) Absolute timer.
cron	(Mandatory) CRON timer.
name	(Optional) Name of the timer.
cron_entry	(Optional) Must be specified if the CRON timer type is specified. Must not be specified if any other timer type is specified. A cron_entry is a partial UNIX crontab entry (the first five fields) as used with the UNIX CRON daemon. A cron_entry specification consists of a text string with five fields. The fields are separated by spaces. The fields represent the time and date when CRON timer events will be triggered. The fields are described in the table titled "Time and Date When CRON Events Will Be Triggered." Ranges of numbers are allowed. Ranges are two numbers separated with a hyphen. The specified range is inclusive. For example, 8-11 for an hour entry specifies execution at hours 8, 9, 10, and 11. A field may be an asterisk (*), which always stands for "first-last." Lists are allowed. A list is a set of numbers (or ranges) separated by commas. Examples: "1,2,5,9" and "0-4,8-12". Step values can be used in conjunction with ranges. Following a range with "/<number>" specifies skips of the number's value through the range. For example, "0-23/2" can be used in the hour field to specify an event that is triggered every other hour. Steps are also permitted after an asterisk, so if you want to say "every two hours", use "*/2". Names can also be used for the month and the day of week fields. Use the first three letters of the particular day or month (case does not matter). Ranges or lists of names are not allowed. The day on which a timer event is triggered can be specified by two fields: day of month and day of week. If both fields are restricted (that is, are not *), an event will be triggered when either field matches the current time. For example, "30 4 1,15 * 5" would cause an event to be triggered at 4:30 a.m. on the 1st and 15th of each month, plus every Friday. Instead of the first five fields, one of seven special strings may appear. These seven special strings are described in the table titled "Special Strings for cron_entry." Example 1: "0 0 1,15 * 1" would trigger an event at midnight on the 1st and 15th of each month, as well as on every Monday. To specify days by only one field, the other field should be set to *; "0 0 * * 1" would trigger an event at midnight only on Mondays. Example 2: "15 16 1 * *" would trigger an event at 4:15 p.m. on the first day of each month. Example 3: "0 12 * * 1-5" would trigger an event at noon on Monday through Friday of each week. Example 4: "@weekly" would trigger an event at midnight once a week on Sunday.

time	(Optional) Must be specified if a timer type other than CRON is specified. Must not be specified if the CRON timer type is specified. For watchdog and countdown timers, the number of seconds and milliseconds until the timer expires; for the absolute timer, the calendar time of the expiration time. Time is specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999. An absolute expiration date is the number of seconds and milliseconds since January 1, 1970. If the date specified has already passed, the timer expires immediately.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Table 76: Time and Date When CRON Events Will Be Triggered

Field	Allowed Values
minute	0-59
hour	0-23
day of month	1-31
month	1-12 (or names, see below)
day of week	0-7 (0 or 7 is Sun, or names; see the table titled "Special Strings for cron_entry")

Table 77: Special Strings for cron_entry

String	Meaning
@yearly	Trigger once a year, "0 0 1 1 *".
@annually	Same as @yearly.
@monthly	Trigger once a month, "0 0 1 * *".
@weekly	Trigger once a week, "0 0 * * 0".
@daily	Trigger once a day, "0 0 * * *".
@midnight	Same as @daily.
@hourly	Trigger once an hour, "0 * * * *".

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
timer_type	Type of the timer. Can be one of the following: • watchdog • countdown • absolute
timer_time_sec timer_time_msec	Time when the timer expired.

Event Type	Description
timer_remain_sec **timer_remain_msec**	The remaining time before the next expiration.

See Also

event_register_timer_subscriber

event_register_timer_subscriber

Registers for a timer event as a subscriber. Use this Tcl command extension to identify the name of the timer to which the event timer, as a subscriber, wants to subscribe. The event timer depends on another policy or another process to actually manipulate the timer. For example, let policyB act as a timer subscriber policy, but policyA (although it does not need to be a timer policy) uses register_timer, timer_arm, or timer_cancel Tcl command extensions to manipulate the timer referenced in policyB.

Syntax

```
event_register_timer_subscriber watchdog|countdown|absolute|cron
name ? [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

watchdog	(Mandatory) Watchdog timer.
countdown	(Mandatory) Countdown timer.
absolute	(Mandatory) Absolute timer.
cron	(Mandatory) CRON timer.
name	(Mandatory) Name of the timer.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Note An EEM policy that registers for a timer event or a counter event can act as both publisher and subscriber.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.

Event Type	Description
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
timer_type	Type of the timer. Can be one of the following: • watchdog • countdown • absolute
timer_time_sec timer_time_msec	Time when the timer expired.
timer_remain_sec **timer_remain_msec**	The remaining time before the next expiration.

See Also

event_register_timer

event_register_track

Registers for a report event from the Cisco IOS Object Tracking subsystem. Use this Tcl command extension to trigger a policy on the basis of a Cisco IOS Object Tracking subsystem report for a specified object number.

Syntax

```
event_register_track ? [tag ?] [state up|down|any] [queue_priority low|normal|high|last]
[maxrun ?]
[nice 0|1]
```

Arguments

? (represents a number)	(Mandatory) Tracked object number in the range from 1 to 500, inclusive.
tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
state	(Optional) Specifies that the tracked object transition will cause an event to be raised. If **up** is specified, an event will be raised when the tracked object transitions from a down state to an up state. If **down** is specified, an event will be raised when the tracked object transitions from an up state to a down state. If **any** is specified, an event will be raised when the tracked object transitions to or from any state.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If an optional argument is not specified, the event matches all possible values of the argument.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"track_number {%u} track_state {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event ID.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
track_number	Number of the tracked object that caused the event to be triggered.
track_state	State of the tracked object when the event was triggered; valid states are up or down.

event_register_wdsysmon

Registers for a Watchdog system monitor event. Use this Tcl command extension to register for a composite event which is a combination of several subevents or conditions. For example, you can use this command to register for the combination of conditions wherein the CPU usage of a certain process is over 80 percent and the memory used by the process is greater than 50 percent of its initial allocation. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
event_register_wdsysmon [tag ?] [timewin ?]
[sub12_op and|or|andnot]
[sub23_op and|or|andnot]
[sub34_op and|or|andnot]
[sub1 subevent-description]
[sub2 subevent-description]
[sub3 subevent-description]
[sub4 subevent-description] [node ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Each argument is position independent.

Note Operator definitions: and (logical and operation), or (logical or operation), andnot (logical and not operation). For example, "sub12_op and" is defined as raise an event when subevent 1 and subevent 2 are true; "sub23_op or" is defined as raise an event when the condition specified in sub12_op is true or subevent 3 is true. The logic can be diagrammed using: if (((sub1 sub12_op sub2) sub23_op sub3) sub34_op sub4) is TRUE, raise event

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
timewin	(Optional) Time window within which all of the subevents have to occur in order for an event to be generated (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).

sub12_op	(Optional) Combination operator for comparison between subevent 1 and subevent 2.
sub23_op	(Optional) Combination operator for comparison between subevent 1 and 2 and subevent 3.
sub34_op	(Optional) Combination operator for comparison between subevent 1 and 2 and subevent 3 and subevent 4.
sub1	(Optional) Indicates that subevent 1 is specified.
subevent-description	(Optional) Syntax for the subevent.
sub2	(Optional) Indicates that subevent 2 is specified.
sub3	(Optional) Indicates that subevent 3 is specified.
sub4	(Optional) Indicates that subevent 4 is specified.
node	(Optional) The node name to be monitored for deadlock conditions is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: node<slot-number>/<cpu-number> The slot-number is the hardware slot number. The cpu-number is the hardware CPU number. For example, the SP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be specified as node0/0. The RP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be addressed as node0/1. If the node argument is not specified, the default node specification is the local node on which the registration is done.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.

maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevents

The syntax of subevent descriptions can be one of seven cases.

For arguments in subevent description, the following constraints apply on the value of number arguments:

- For dispatch_mgr, val must be an integer between 0 and 4294967295, inclusive.
- For cpu_proc and cpu_tot, val must be an integer between 0 and 100, inclusive.
- For mem_proc, mem_tot_avail, and mem_tot_used, if is_percent is FALSE, val must be an integer between 0 and 4294967295, inclusive.

```
1. deadlock procname ?
```

Arguments

procname	(Mandatory) A regular expression that specifies the process name that you wish to monitor for deadlock conditions. This subevent will ignore the time window even if it is given.

```
2. dispatch_mgr [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for dispatch_manager status.
op	(Optional) Comparison operator used to compare the collected number of events with the specified value; if true, an event will be raised.
val	(Optional) The value with which the number of events that have occurred should be compared.
period	(Optional) The time period for the number of events that have occurred (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
3. cpu_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for CPU utilization conditions.
op	(Optional) Comparison operator used to compare the collected CPU usage sample percentage with the specified percentage value; if true, an event will be raised.
val	(Optional) The percentage value with which the average CPU usage during the sample period should be compared.
period	(Optional) The time period for averaging the collection of samples (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
4. cpu_tot [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected total system CPU usage sample percentage with the specified percentage value; if true, an event will be raised.
val	(Optional) The percentage value with which the average CPU usage during the sample period should be compared.
period	(Optional) The time period for averaging the collection of samples (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
5. mem_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for memory usage.
op	(Optional) Comparison operator used to compare the collected memory used with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If memory usage has increased from 150 KB to 300 KB within the time period, the percentage increase is 100. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.

period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
6. mem_tot_avail [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected available memory with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If available memory usage has decreased from 300 KB to 150 KB within the time period, the percentage decrease is 50. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.
period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
7. mem_tot_used [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected used memory with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If memory usage has increased from 150 KB to 300 KB within the time period, the percentage increase is 100. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.

period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used. **Note** This argument is mandatory if is_percent is set to TRUE; otherwise, it is optional.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
num_subs	Subevent number.

Where the subevent info string is for a deadlock subevent:

```
"{type %s num_entries %u entries {entry 1, entry 2, ...}}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
num_entries	Number of processes and threads in the deadlock.
entries	Information of processes and threads in the deadlock.

Where each entry is:

```
"{node {%s} procname {%s} pid %u tid %u state %s b_node %s b_procname %s b_pid %u
b_tid %u}"
```

Assume that the entry describes the scenario in which Process A thread m is blocked on process B thread n:

Subevent Type	Description
node	Name of the node that process A thread m is on.
procname	Name of process A.
pid	Process ID of process A.
tid	Thread ID of process A thread m.
state	Thread state of process A thread m. Can be one of the following: • STATE_CONDVAR • STATE_DEAD • STATE_INTR • STATE_JOIN • STATE_MUTEX • STATE_NANOSLEEP • STATE_READY • STATE_RECEIVE • STATE_REPLY • STATE_RUNNING • STATE_SEM • STATE_SEND • STATE_SIGSUSPEND • STATE_SIGWAITINFO • STATE_STACK • STATE_STOPPED • STATE_WAITPAGE • STATE_WAITTHREAD
b_node	Name of the node that process B thread is on.
b_procname	Name of process B.
b_pid	Process ID of process B.

Subevent Type	Description
b_tid	Thread ID of process B thread n; 0 means that process A thread m is blocked on all threads of process B.

For dispatch_mgr Subevent

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the owner process of this dispatch manager.
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the number of events processed by the dispatch manager is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the total number of events processed by this dispatch manager is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For cpu_proc Subevent

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the process whose CPU utilization is being monitored.

Subevent Type	Description
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the process CPU utilization is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged process CPU utilization is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For cpu_tot Subevent

```
"{type %s node {%s} value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node on which the total CPU utilization is being monitored.
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total CPU utilization is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total CPU utilization is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For mem_proc Subevent

```
"{type %s node {%s} procname {%s} pid %u is_percent %s value %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the process whose memory usage is being monitored.

Subevent Type	Description
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the process used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged process used memory utilization is in the given time window.
Subevent Type	Description
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first process used memory sample ever collected and the latest process used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest process used memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **value** is the process used memory in the latest sample.
- **diff** is 0.
- **sec** and **msec** are both 0.

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **value** is the averaged process used memory sample value in the specified time window.
- **diff** is 0.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **value** is 0.
- **diff** is the percentage difference between the oldest and latest process used memory samples in the specified time window.
- **sec** and **msec** are the actual time difference between the time stamps of the oldest and latest process used memory samples in this time window.

If the **is_percent** argument is TRUE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **value** is 0.
- **diff** is the percentage difference between the first process used memory sample ever collected and the latest process used memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first process used memory sample ever collected and the latest process used memory sample.

For mem_tot_avail Subevent

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node for which the total available memory is being monitored.
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).
used	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total used memory utilization is in the given time window.
avail	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **avail** is in the latest total available memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **avail** is the total available memory utilization in the specified time window.
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first total available memory sample ever collected and the latest total available memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest total available memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, they are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is the total used memory in the latest sample.
- **avail** is the total available memory in the latest sample.
- **diff** is 0.

- **sec** and **msec** are both 0.

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is 0.
- **avail** is the averaged total available memory sample value in the specified time window.
- **diff** is 0.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total available memory samples in this time window.

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the oldest and latest total available memory samples in the specified time window.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total available memory samples in this time window.

If the **is_percent** argument is TRUE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the first total available memory sample ever collected and the latest total available memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first total available memory sample ever collected and the latest total available memory sample.

For mem_tot_used Subevent

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node for which the total used memory is being monitored.
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).

Subevent Type	Description
used	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total used memory utilization is in the given time window.
avail	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **avail** is in the latest total used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **avail** is the total used memory utilization in the specified time window.
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first total used memory sample ever collected and the latest total used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest total used memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the sec and msec arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is the total used memory in the latest sample,
- **avail** is the total available memory in the latest sample,
- **diff** is 0,
- **sec** and **msec** are both 0,

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is the averaged total used memory sample value in the specified time window,
- **avail** is 0,
- **diff** is 0,
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total used memory samples in this time window,

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

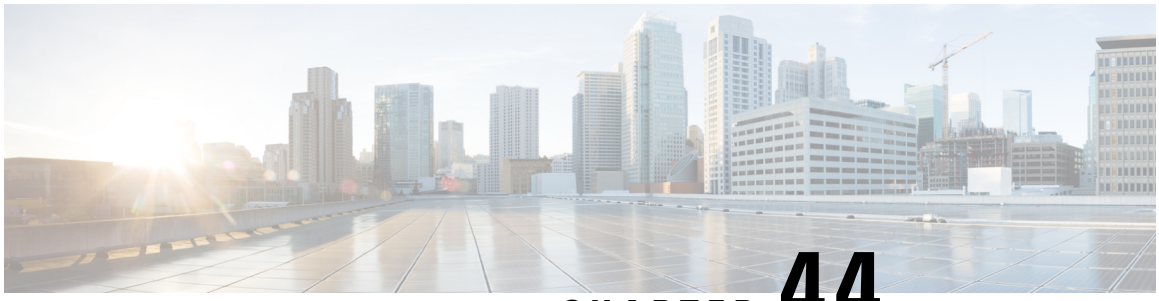
- **used** is 0.
- **avail** is 0.

- **diff** is the percentage difference between the oldest and latest total used memory samples in the specified time window.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total used memory samples in this time window.

If the **is_percent** argument is TRUE, and the sec and msec arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the first total used memory sample ever collected and the latest total used memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first total used memory sample ever collected and the latest total used memory sample.

Note Inside a subevent description, each argument is position independent.



CHAPTER 44

EEM Event Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

event_completion

Sends a notification to the EEM server that the policy is done servicing the event that triggered it. The event only takes a single argument which is the **return_code** of this event instance.

Syntax

```
event_completion status ?
```

Arguments

status	(Mandatory) Exit status (return_code) of this event instance. A value of zero indicates no error and any other integer value indicates an error.

Result String

None

Set _cerrno

No

event_completion_with_wait

The **event_completion_with_wait** command combines the two commands **event_completion** and **event_wait** into a single command for ease of use.

The **event_completion** command sends a notification to the EEM server that the policy is done servicing the event that triggered it. The event only takes a single argument which is the **return_code** of this event instance.

The **event_wait** places the Tcl policy into a sleep state. When the Tcl policy receives a new signal announcing a new event, the policy is placed into a wake state and again returns to a sleep state. This loop continues. If **event_wait** policy is invoked before **event_completed** policy, an error results and the policy exits.

Syntax

```
event_completion_with_wait status ? [refresh_vars]
```

Arguments

status	(Mandatory) exit_status (return_code) of this event instance. A value of zero indicates no error. Any other integer value indicates an error.
refresh_vars	(Optional) Indicates whether built-in and environment variables should be updated (refreshed) from the EEM Policy Director during this event instance.

Result String

None

Set _cerrno

Yes

Sample Usage

Here is a similar example as above using this single command:

```
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
```

```
set i 1
while {1 == 1} {  # Start high performance policy loop
   array set arr_einfo [event_reqinfo]
   if {$_cerrno != 0} {
       set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
       $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
       error $result
   }
   action_syslog msg "event $i serviced" priority info
   if {$i == 5} {
       action_syslog msg "Exiting after servicing 5 events" priority info
       exit 0
   }
   incr i
   array set _event_state_arr [event_completion_with_wait status 0 refresh_vars 1]
   if {$_event_state_arr(event_state) != 0} {
       action_syslog msg "Exiting: failed event_state " \
                         " $event_state_arr(event_state)" priority info
       exit 0
   }
}
```

Note The running configuration output is the same as the event_publishTcl command.

event_publish

Publishes an application-specific event.

Syntax

```
event_publish sub_system ? type ? [arg1 ?] [arg2 ?] [arg3 ?] [arg4 ?]
```

Arguments

sub_system	(Mandatory) Number assigned to the EEM policy that published the application-specific event. Number is set to 798 because all other numbers are reserved for Cisco use.
type	(Mandatory) Event subtype within the specified component. The sub_system and type arguments uniquely identify an application event. Must be an integer between 1 and 4294967295, inclusive.
[arg1 ?]-[arg4 ?]	(Optional) Four pieces of application event publisher string data.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

Sample Usage

This example demonstrates how to use the **event_publish** Tcl command extension to execute a script *n* times repeatedly to perform some function (for example, to measure the amount of CPU time taken by a given group of Tcl statements). This example uses two Tcl scripts.

Script1 publishes a type 9999 EEM event to cause Script2 to run for the first time. Script1 is registered as a none event and is run using the Cisco IOS CLI **event manager run** command. Script2 is registered as an EEM application event of type 9999, and this script checks to see if the application publish arg1 data (the iteration number) exceeds the EEM environment variable test_iterations value. If the test_iterations value is exceeded, the script writes a message and exits; otherwise the script executes the remaining statements and reschedules another run. To measure the CPU utilization for Script2, use a value of test_iterations that is a multiple of 10 to calculate the amount of average CPU time used by Script2.

To run the Tcl scripts, enter the following Cisco IOS commands:

```
configure terminal
 event manager environment test_iterations 100
 event manager policy script1.tcl
 event manager policy script2.tcl
 end
event manager run script1.tcl
```

The Tcl script Script2 will be executed 100 times. If you execute the script without the extra processing and derive the average CPU utilization, and then add the extra processing and repeat the test, you can subtract the former CPU utilization from the later CPU utilization to determine the average for the extra processing.

Script1 (script1.tcl)

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# Query the event info.
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

action_syslog priority info msg "EEM application_publish test start"
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

# Cause the first iteration to run.
event_publish sub_system 798 type 9999 arg1 0
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
```

```
    error $result
}
```

Script2 (script2.tcl)

```
::cisco::eem::event_register_appl sub_system 798 type 9999

# Check if all the required environment variables exist.
# If any required environment variable does not exist, print out an error msg and quit.
if {![info exists test_iterations]} {
    set result \
      "Policy cannot be run: variable test_iterations has not been set"
    error $result $errorInfo
}

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

# Query the event info.
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# Data1 contains the arg1 value used to publish this event.
set iter $arr_einfo(data1)

# Use the arg1 info from the previous run to determine when to end.
if {$iter >= $test_iterations} {
    # Log a message.
    action_syslog priority info msg "EEM application_publish test end"
    if {$_cerrno != 0} {
      set result [format \
            "component=%s; subsys err=%s; posix err=%s;\n%s" \
            $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
      error $result
    }
    exit 0
}
set iter [expr $iter + 1]

# Log a message.
set msg [format "EEM application_publish test iteration %s" $iter]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

# Do whatever processing that you want to measure here.

# Cause the next iteration to run. Note that the iteration is passed to the
# next operation as arg1.
event_publish sub_system 798 type 9999 arg1 $iter
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

event_wait

Places the Tcl policy into a sleep state. When the Tcl policy receives a new signal announcing a new event, the policy is placed into a wake state and again returns to a sleep state. This loop continues. If **event_wait** policy is invoked before **event_completed** policy, an error results and the policy exits.

Syntax

```
event_wait [refresh_vars]
```

Arguments

refresh_vars	(Optional) Indicates whether built-in and environment variables should be updated (refreshed) from the EEM Policy Director during this event instance.

Result String

None

Set _cerrno

No

Sample Usage

The **event_wait** event detector returns an array type value with a single element named **event_state**. Event_state is a value sent back from the EEM Server indicating whether or not an error has occurred in processing the event. An example of an error here would be if the user configured **event_wait** before configuring **event_completion** when handling the event instance.

The following sample output shows the use of both **event_completion** and **event_wait**Tcl commands:

```
::cisco::eem::event_register_syslog tag e1 occurs 1 pattern CLEAR maxrun 0
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set i 1
while {1 == 1} {  # Start high performance policy loop
   array set arr_einfo [event_reqinfo]
   if {$_cerrno != 0} {
       set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
       $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
       error $result
   }
   action_syslog msg "event $i serviced" priority info
   if {$i == 5} {
       action_syslog msg "Exiting after servicing 5 events" priority info
       exit 0
   }
   incr i
   event_completion status 0
   array set _event_state_arr [event_wait refresh_vars 0]
   if {$_event_state_arr(event_state) != 0} {
       action_syslog msg "Exiting: failed event_state " \
                         " $event_state_arr(event_state)" priority info
```

```
        exit 0
    }
}
```

Here is an example of the running configuration:

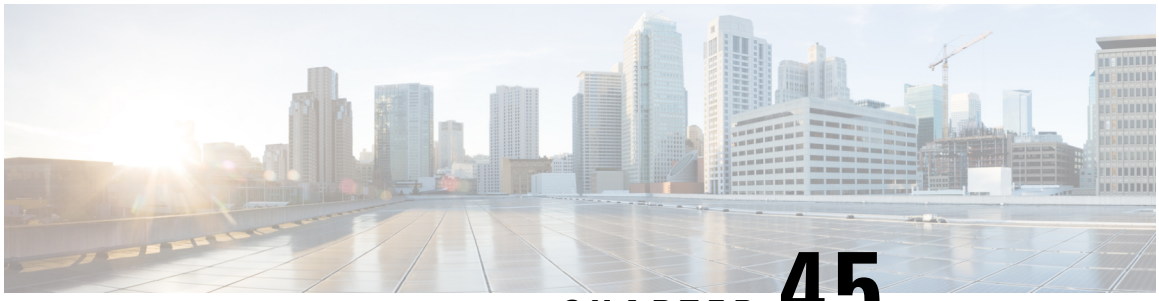
```
Device#
01:00:44: %SYS-5-CONFIG_I: Configured from console by consoleclear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:00:49: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:49: %HA_EM-6-LOG: high_perf_example.tcl: event 1 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:00:53: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:53: %HA_EM-6-LOG: high_perf_example.tcl: event 2 serviced
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:00:56: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:56: %HA_EM-6-LOG: high_perf_example.tcl: event 3 serviced
Device#
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:00:59: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
01:00:59: %HA_EM-6-LOG: high_perf_example.tcl: event 4 serviced
01:00:59: %HA_EM-6-LOG: high_perf_example.tcl: Exiting after servicing 5 events
Device#
Device#
Device#copy tftp disk1:
Address or name of remote host [dirt]?
Source filename [user/eem_scripts/high_perf_example.tcl]?
Destination filename [high_perf_example.tcl]?
%Warning:There is a file already existing with this name
Do you want to over write? [confirm]
Accessing tftp://dirt/user/eem_scripts/high_perf_example.tcl...
Loading user/eem_scripts/high_perf_example.tcl from 192.0.2.19 (via FastEthernet0/0): !
[OK - 909 bytes]
909 bytes copied in 0.360 secs (2525 bytes/sec)
Device#
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#no event manager policy high_perf_example.tcl
Device(config)#event manager po high_perf_example.tcl
Device(config)#end
Device#
Device#
Device#
Device#
01:02:19: %SYS-5-CONFIG_I: Configured from console by consoleclear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:02:23: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
Device#
01:02:23: %HA_EM-6-LOG: high_perf_example.tcl: event 1 serviced
Device#
```

```
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:26: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:26: %HA_EM-6-LOG: high_perf_example.tcl: event 2 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:29: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:29: %HA_EM-6-LOG: high_perf_example.tcl: event 3 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:33: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
01:02:33: %HA_EM-6-LOG: high_perf_example.tcl: event 4 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
Device#
01:02:36: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:36: %HA_EM-6-LOG: high_perf_example.tcl: event 5 serviced
01:02:36: %HA_EM-6-LOG: high_perf_example.tcl: Exiting after servicing 5 events
Device#
```

Also while an event has been serviced and is waiting for the next event to come in **show event manager policy active** command will display the following output:

```
Device#show event manager policy active
Key: p - Priority        :L - Low, H - High, N - Normal, Z - Last
     s - Scheduling node :A - Active, S - Standby
default class - 1 script event
 no.  job id      p s status  time of event             event type          name
 1    11          N A wait    Mon Oct20 14:15:24 2008   syslog
high_perf_example.tcl
```

In the above example the status is wait. This indicates that the policy is waiting for the next event to come in.



CHAPTER 45

EEM Library Debug Command Extensions

cli_debug

Prints a command-line interface (CLI) debug statement to syslog. This Tcl command extension is used to print a CLI debug statement to syslog if the **debug event manager tcl cli_library** Cisco IOS CLI command is in effect.

Syntax

```
cli_debug spec_string debug_string
```

Arguments

spec_string	(Mandatory) The spec_string argument is used to indicate the type of debug statement.
debug_string	(Mandatory) The debug_string argument is used to indicate the debugging text.

Result String

None

Set _cerrno

No

smtp_debug

Prints a a Simple Mail Transfer Protocol (SMTP) debug statement to syslog. This Tcl command extension prints a SMTP debug statement to syslog if the **debug event manager tcl smtp_library** Cisco IOS command-line interface (CLI) command is in effect.

Syntax

```
smtp_debug spec_string debug_string
```

Arguments

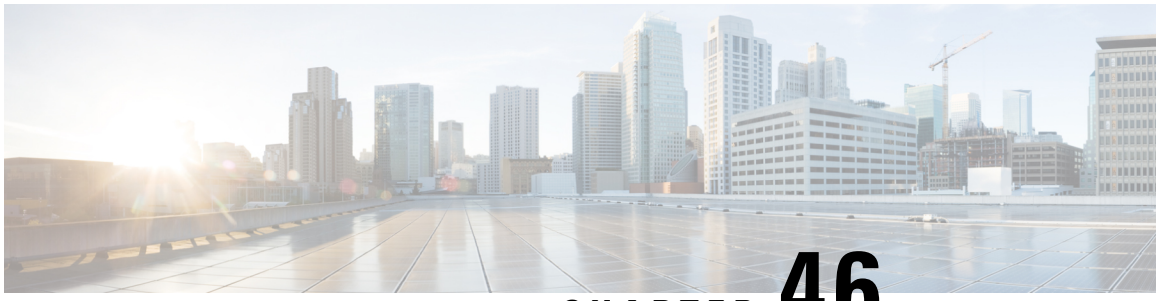
spec_string	(Mandatory) The spec_string argument is used to indicate the type of debug statement.
debug_string	(Mandatory) The debug_string argument is used to indicate the debugging text.

Result String

None

Set _cerrno

No



CHAPTER 46

EEM Multiple Event Support Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

attribute

Specifies a complex event.

Syntax

```
attribute tag ? [occurs ?]
```

Arguments

tag	Specifies a tag using the *event-tag* argument that can be used with the **attribute** command to associate an event.
occurs	(Optional) Specifies the number of occurrences before an EEM event is triggered. If not specified, an EEM event is triggered on the first occurrence. The range is from 1 to 4294967295.

Result String

None

Set _cerrno

No

correlate

Builds a single complex event and allows boolean logic to relate events and tracked objects.

Syntax

```
correlate event ? track ? [andnot | and | or] event ? track ?
```

Arguments

event	Specifies the event that can be used with the **trigger** command to support multiple event statements within an script. If the event associated with the *event-tag* argument occurs for the number of times specified by the **trigger** command, the result is true. If not, the result is false.
track	Specifies the event object number for tracking. The range is from 1 to 500. If the tracked object is set, the result of the evaluation is true. If the tracked object is not set or is undefined, the result of the evaluation is false. This result is regardless of the state of the object.
andnot	(Optional) Specifies that if event 1 occurs the action is executed, and if event 2 and event 3 occur together the action is not executed.
and	(Optional) Specifies that if event 1 occurs the action is executed, and if event 2 and event 3 occur together the action is executed. **Note** When "and" is used to group events such as traps or syslog messages, then the default trigger occurrence window is three minutes.
or	(Optional) Specifies that if event 1 occurs the action is executed, or else if event 2 and event 3 occur together the action is executed.

Result String

None

Set _cerrno

No

trigger

Specifies the multiple event configuration ability of Embedded Event Manager (EEM) events. A multiple event is one that can involve one or more event occurrences, one or more tracked object states, and a time period for the event to occur. The events are raised based on the specified parameters.

Syntax

```
trigger [occurs ?] [period ?] [period-start ?] [delay ?]
```

Arguments

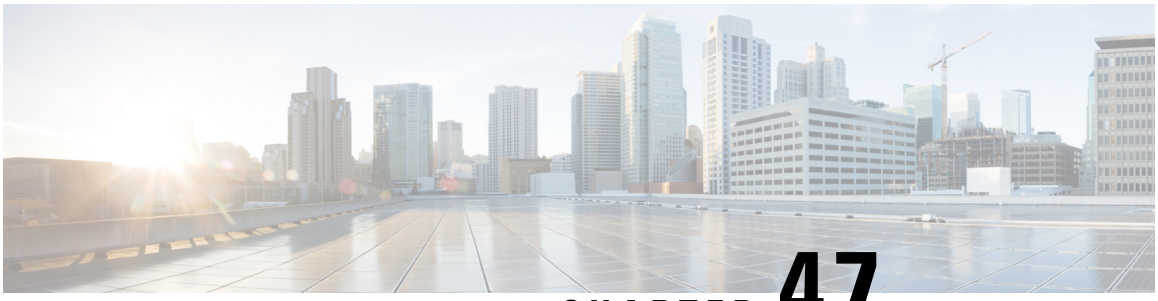
occurs	(Optional) Specifies the number of times the total correlation occurs before an EEM event is raised. When a number is not specified, an EEM event is raised on the first occurrence. The range is from 1 to 4294967295.
period	(Optional) Time interval in seconds and optional milliseconds, during which the one or more occurrences must take place. This is specified in the format sssssssssss[.mmm], where sssssssssss must be an integer number representing seconds between 0 and 4294967295, inclusive and mmm represents milliseconds and must be an integer number between 0 to 999.
period-start	(Optional) Specifies the start of an event correlation window. If not specified, event monitoring is enabled after the first CRON period occurs.
delay	(Optional) Specifies the number of seconds and optional milliseconds after which an event will be raised if all the conditions are true (specified in the format sssssssssss[.mmm], where sssssssssss must be an integer number representing seconds between 0 and 4294967295, inclusive and mmm represents milliseconds and must be an integer number between 0 to 999).

Result String

None

Set _cerrno

No



CHAPTER 47

EEM SMTP Library Command Extensions

All Simple Mail Transfer Protocol (SMTP) library command extensions belong to the ::cisco::lib namespace.

To use this library, the user needs to provide an e-mail template file. The template file can include Tcl global variables so that the e-mail service and the e-mail text can be configured through the **event manager environment Cisco IOS**command-line interface (CLI) configuration command. There are commands in this library to substitute the global variables in the e-mail template file and to send the desired e-mail context with the To address, CC address, From address, and Subject line properly configured using the configured e-mail server.

E-Mail Template

The e-mail template file has the following format:

Note Based on RFC 2554, the SMTP e-mail server name--Mailservername-- can be in any one of the following template formats: username:password@host, username@host, or host.

```
Mailservername:<space><the list of candidate SMTP server addresses>
From:<space><the e-mail address of sender>
To:<space><the list of e-mail addresses of recipients>
Cc:<space><the list of e-mail addresses that the e-mail will be copied to>
Sourceaddr:<space><the IP addresses of the recipients>
Subject:<subject line>
<a blank line>
<body>
```

Note Note that the template normally includes Tcl global variables for configuration.

In a Tcl policy, the port number can be specified by a "Port" line in the e-mail template. If port is not specified, the default port of 25 is used.

Below is a sample e-mail template file:

```
Mailservername: $_email_server
From: $_email_from
To: $_email_to
Cc: $_email_cc
```

```
Sourceaddr: $_email_ipaddr
Port: <port number>
Subject: From router $routername: Process terminated
process name: $process_name
subsystem: $sub_system
exit status: $exit_status
respawn count: $respawn_count
```


- smtp_send_email, on page 824
- smtp_subst, on page 825

smtp_send_email

Given the text of an e-mail template file with all global variables already substituted, sends the e-mail out using Simple Mail Transfer Protocol (SMTP). The e-mail template specifies the candidate mail server addresses, To addresses, CC addresses, From address, subject line, and e-mail body.

Note A list of candidate e-mail servers can be provided so that the library will try to connect the servers on the list one by one until it can successfully connect to one of them.

Syntax

```
smtp_send_email text
```

Arguments

text	(Mandatory) The text of an e-mail template file with all global variables already substituted.

Result String

None

Set _cerrno

- Wrong 1st line format--Mailservername:list of server names.
- Wrong 2nd line format--From:from-address.
- Wrong 3rd line format--To:list of to-addresses.
- Wrong 4th line format--CC:list of cc-addresses.
- Error connecting to mail server:--$sock closed by remote server (where $sock is the name of the socket opened to the mail server).
- Error connecting to mail server:--$sock reply code is $k instead of the service ready greeting (where $sock is the name of the socket opened to the mail server; $k is the reply code of $sock).
- Error connecting to mail server:--cannot connect to all the candidate mail servers.
- Error disconnecting from mail server:--$sock closed by remote server (where $sock is the name of the socket opened to the mail server).

Sample Scripts

After all needed global variables in the e-mail template are defined:

```
if [catch {smtp_subst [file join $tcl_library email_template_sm]} result] {
    puts stderr $result
    exit 1
}
if [catch {smtp_send_email $result} result] {
    puts stderr $result
    exit 1
}
```

smtp_subst

Given an e-mail template file e-mail_template, substitutes each global variable in the file by its user-defined value. Returns the text of the file after substitution.

Syntax

```
smtp_subst e-mail_template
```

Arguments

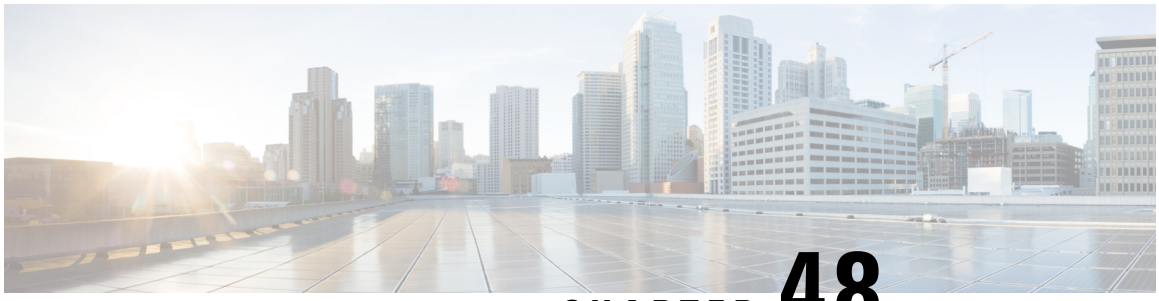
e-mail_template	(Mandatory) Name of an e-mail template file in which global variables need to be substituted by a user-defined value. An example filename could be /disk0://example.template which represents a file named example.template in a top-level directory on an ATA flash disk in slot 0.

Result String

The text of the e-mail template file with all the global variables substituted.

Set _cerrno

- cannot open e-mail template file
- cannot close e-mail template file



CHAPTER 48

EEM System Information Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note All EEM system information commands--**sys_reqinfo_xxx**--have the Set _cerrno section set to yes.

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

• sys_reqinfo_syslog_history, on page 837

sys_reqinfo_cli_freq

Queries the frequency information of all command-line interface (CLI) events.

Syntax

```
sys_reqinfo_cli_freq
```

Arguments

None

Result String

```
rec_list {{CLI frequency string 0},{CLI frequency str 1}, ...}
```

Where each CLI frequency string is:

time_sec %ld time_msec %ld match_count %u raise_count %u occurs %u period_sec %ld period_msec %ld pattern {%s}

rec_list	Marks the start of the CLI event frequency list.
time_sec time_msec	Last time when this CLI event was raised.
match count	Number of times that a CLI command matches the pattern specified by this CLI event specification.
raise_count	Number of times that this CLI event was raised. The following fields are information about the CLI event specification: • sync--A "yes" means that event publish should be performed sychronously. The event detector will be notified when the Event Manager Server has completed publishing the event. The Event Manager Server will return a code that indicates whether or not the CLI command should be executed. • skip--A "yes" means that the CLI command should not be executed if the sync flag is not set.
occurs	Number of occurrences before an event is raised; if this argument is not specified, an event is raised on the first occurrence.
period_sec period_msec	Number of occurrences must occur within this number of POSIX timer units in order to raise event; if this argument is not specified, it does not apply.
pattern	Regular expression used to perform CLI command pattern matching.

Set _cerrno

Yes

sys_reqinfo_cli_history

Queries the history of command-line interface (CLI) commands.

Syntax

```
sys_reqinfo_cli_history
```

Arguments

None

Result String

```
rec_list {{CLI history string 0}, {CLI history str 1},...}
```

Where each CLI history string is:

```
time_sec %ld time_msec %ld cmd {%s}
```

rec_list	Marks the start of the CLI command history list.
time_sec time_msec	Time when the CLI command was run.
cmd	Text of the CLI command.

Set _cerrno

Yes

sys_reqinfo_cpu_all

Queries the CPU utilization of the top processes (both POSIX processes and IOS processes) during a specified time period and in a specified order. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
sys_reqinfo_cpu_all order cpu_used [sec ?] [msec ?] [num ?]
```

Arguments

order	(Mandatory) Order used for sorting the CPU utilization of processes.
cpu_used	(Mandatory) Specifies that the average CPU utilization, for the specified time window, will be sorted in descending order.

sec msec	(Optional) The time period, in seconds and milliseconds, during which the average CPU utilization is calculated. Must be integers in the range from 0 to 4294967295. If not specified, or if both sec and msec are specified as 0, the most recent CPU sample is used.
num	(Optional) Number of entries from the top of the sorted list of processes to be displayed. Must be an integer in the range from 1 to 4294967295. Default value is 5.

Result String

```
rec_list {{process CPU info string 0},{process CPU info string 1}, ...}
```

Where each process CPU info string is:

```
pid %u name {%s} cpu_used %u
```

rec_list	Marks the start of the process CPU information list.
pid	Process ID.
name	Process name.
cpu_used	Specifies that if sec and msec are specified with a number greater than zero, the average percentage is calculated from the process CPU utilization during the specified time period. If sec and msec are both zero or not specified, the average percentage is calculated from the process CPU utilization in the latest sample.

Set _cerrno

Yes

sys_reqinfo_crash_history

Queries the crash information of all processes that have ever crashed. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
sys_reqinfo_crash_history
```

Arguments

None

Result String

```
rec_list {{crash info string 0},{crash info string 1}, ...}
Where each crash info string is:
job_id %u name {%s} respawn_count %u fail_count %u dump_count %u
inst_id %d exit_status 0x%x exit_type %d proc_state {%s} component_id 0x%x
crash_time_sec %ld crash_time_msec %ld
```

job_id	System manager assigned job ID for the process. An integer between 1 and 4294967295, inclusive.
name	Process name.
respawn_count	Total number of restarts for the process.
fail_count	Number of restart attempts of the process. This count is reset to zero when the process is successfully restarted.
dump_count	Number of core dumps performed.
inst_id	Process instance ID.
exit_status	Last exit status of the process.
exit_type	Last exit type.
proc_state	Sysmgr process states. One of the following: error, forced_stop, hold, init, ready_to_run, run, run_rnode, stop, waitEOltimer, wait_rnode, wait_spawntimer, wait_tpl.
component_id	Version manager assigned component ID for the component to which the process belongs.
crash_time_sec crash_time_msec	Seconds and milliseconds since January 1, 1970, which represent the last time the process crashed.

Set _cerrno

Yes

sys_reqinfo_mem_all

Queries the memory usage of the top processes (both POSIX and IOS) during a specified time period and in a specified order. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
sys_reqinfo_mem_all order allocates|increase|used [sec ?] [msec ?] [num ?]
```

Arguments

order	(Mandatory) Order used for sorting the memory usage of processes.
allocates	(Mandatory) Specifies that the memory usage is sorted by the number of process allocations during the specified time window, and in descending order.
increase	(Mandatory) Specifies that the memory usage is sorted by the percentage of process memory increase during the specified time window, and in descending order.
used	(Mandatory) Specifies that the memory usage is sorted by the current memory used by the process.

sec msec	(Optional) The time period, in seconds and milliseconds, during which the process memory usage is calculated. Must be integers in the range from 0 to 4294967295. If both sec and msec are specified and are nonzero, the number of allocations is the difference between the number of allocations in the oldest and latest samples collected in the time period. The percentage is calculated as the the percentage difference between the memory used in the oldest and latest samples collected in the time period. If not specified, or if both sec and msec are specified as 0, the first sample ever collected is used as the oldest sample; that is, the time period is set to be the time from startup until the current moment.
num	(Optional) Number of entries from the top of the sorted list of processes to be displayed. Must be an integer in the range from 1 to 4294967295. Default value is 5.

Result String

```
rec_list {{process mem info string 0},{process mem info string 1}, ...}
```

Where each process mem info string is:

```
pid %u name {%s} delta_allocs %d initial_alloc %u current_alloc %u percent_increase %d
```

rec_list	Marks the start of the process memory usage information list.
pid	Process ID.
name	Process name.
delta_allocs	Specifies the difference between the number of allocations in the oldest and latest samples collected in the time period.
initial_alloc	Specifies the amount of memory, in kilobytes, used by the process at the start of the time period.
current_alloc	Specifies the amount of memory, in kilobytes, currently used by the process.
percent_increase	Specifies the percentage difference between the memory used in the oldest and latest samples collected in the time period. The percentage difference can be expressed as current_alloc minus initial_alloc times 100 and divided by initial_alloc.

Set _cerrno

Yes

sys_reqinfo_proc

Queries the information about a single POSIX process. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
sys_reqinfo_proc job_id ?
```

Arguments

job_id	(Mandatory) System manager assigned job ID for the process. Must be an integer between 1 and 4294967295, inclusive.

Result String

```
job_id %u component_id 0x%x name {%s} helper_name {%s} helper_path {%s} path {%s}
node_name {%s} is_respawn %u is_mandatory %u is_hold %u dump_option %d
max_dump_count %u respawn_count %u fail_count %u dump_count %u
last_respawn_sec %ld last_respawn_msec %ld inst_id %u proc_state %s
level %d exit_status 0x%x exit_type %d
```

job_id	System manager assigned job ID for the process. An integer between 1 and 4294967295, inclusive.
component_id	Version manager assigned component ID for the component to which the process belongs.
name	Process name.
helper_name	Helper process name.
helper_path	Executable path of the helper process.
path	Executable path of the process.
node_name	System manager assigned node name for the node to which the process belongs.
is_respawn	Flag that specifies that the process can be respawned.
is_mandatory	Flag that specifies that the process must be alive.
is_hold	Flag that specifies that the process is spawned until called by the API.
dump_option	Core dumping options.
max_dump_count	Maximum number of core dumping permitted.
respawn_count	Total number of restarts for the process.
fail_count	Number of restart attempts of the process. This count is reset to zero when the process is successfully restarted.
dump_count	Number of core dumps performed.
last_respawn_sec last_respawn_msec	Seconds and milliseconds in POSIX timer units since January 1, 1970, which represent the last time the process was started.
inst_id	Process instance ID.
proc_state	Sysmgr process states. One of the following: error, forced_stop, hold, init, ready_to_run, run, run_rnode, stop, waitEOltimer, wait_rnode, wait_spawntimer, wait_tpl.

level	Process run level.
exit_status	Last exit status of the process.
exit_type	Last exit type.

Set _cerrno

Yes

sys_reqinfo_proc_all

Queries the information of all POSIX processes. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
sys_reqinfo_proc_all
```

Arguments

None

Result String

```
rec_list {{process info string 0}, {process info string 1},...}
```

Where each process info string is the same as the result string of the **sysreq_info_proc** Tcl command extension.

Set _cerrno

Yes

sys_reqinfo_routername

Queries the device name.

Syntax

```
sys_reqinfo_routername
```

Arguments

None

Result String

```
routername %s
```

Where routername is the name of the device.

Set _cerrno

Yes

sys_reqinfo_snmp

Queries the value of the entity specified by a Simple Network Management Protocol (SNMP) object ID.

Syntax

```
sys_reqinfo_snmp oid ? get_type exact|next
```

Arguments

oid	(Mandatory) SNMP OID in dot notation (for example, 1.3.6.1.2.1.2.1.0).
get_type	(Mandatory) Type of SNMP get operation that needs to be applied to the specified oid. If the get_type is "exact," the value of the specified oid is retrieved; if the get_type is "next," the value of the lexicographical successor to the specified oid is retrieved.

Result String

```
oid {%s} value {%s}
```

oid	SNMP OID.
value	Value string of the associated SNMP data element.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 37)    FH_ENOSNMPDATA  (can't retrieve data from SNMP)
```

This error means that there was no data for the SNMP object type.

```
(_cerr_sub_err = 51)    FH_ESTATSTYP  (invalid statistics data type)
```

This error means that the SNMP statistics data type was invalid.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

sys_reqinfo_syslog_freq

Queries the frequency information of all syslog events.

Syntax

```
sys_reqinfo_syslog_freq
```

Arguments

None

Result String

```
rec_list {{event frequency string 0}, {log freq str 1}, ...}
```

Where each event frequency string is:

```
time_sec %ld time_msec %ld match_count %u raise_count %u occurs %u
period_sec %ld period_msec %ld pattern {%s}
```

time_sec time_msec	Seconds and milliseconds in POSIX timer units since January 1, 1970, which represent the time the last event was raised.
match_count	Number of times that a syslog message matches the pattern specified by this syslog event specification since event registration.
raise_count	Number of times that this syslog event was raised.
occurs	Number of occurrences needed in order to raise the event; if not specified, the event is raised on the first occurrence.
period_sec period_msec	Number of occurrences must occur within this number of POSIX timer units in order to raise the event; if not specified, the period check does not apply.
pattern	Regular expression used to perform syslog message pattern matching.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 45)    FH_ESEQNUM  (sequence or workset number out of sync)
```

This error means that the event detector sequence or workset number was invalid.

```
(_cerr_sub_err = 46)    FH_EREGEMPTY  (registration list is empty)
```

This error means that the event detector registration list was empty.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

sys_reqinfo_syslog_history

Queries the history of the specified syslog message.

Syntax

```
sys_reqinfo_syslog_history
```

Arguments

None

Result String

```
rec_list {{log hist string 0}, {log hist str 1}, ...}
```

Where each log hist string is:

```
time_sec %ld time_msec %ld msg {%s}
```

time_sec time_msec	Seconds and milliseconds since January 1, 1970, which represent the time the message was logged.
msg	Syslog message.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 44)    FH_EHISTEMPTY  (history list is empty)
```

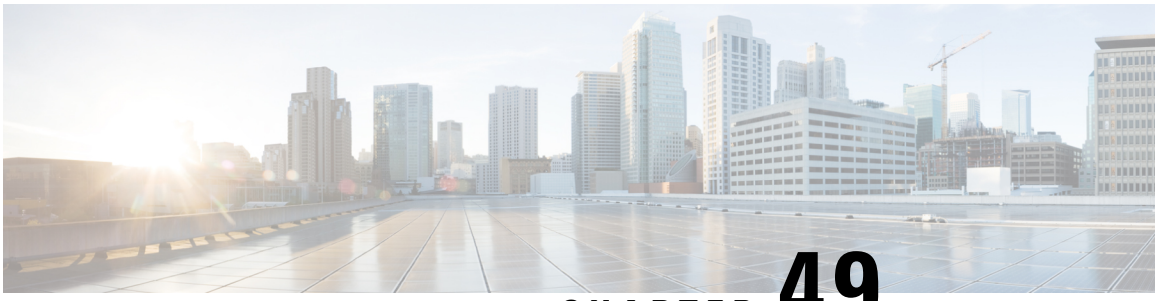
This error means that the history list was empty.

```
(_cerr_sub_err = 45)    FH_ESEQNUM  (sequence or workset number out of sync)
```

This error means that the event detector sequence or workset number was invalid.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.



CHAPTER 49

EEM Utility Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

appl_read

Reads Embedded Event Manager (EEM) application volatile data. This Tcl command extension provides support for reading EEM application volatile data. EEM application volatile data can be published by a Cisco software process that uses the EEM application publish API. EEM application volatile data cannot be published by an EEM policy.

Note Currently there are no Cisco software processes that publish application volatile data.

Syntax

```
appl_read name ? length ?
```

Arguments

name	(Mandatory) Name of the application published string data.
length	(Mandatory) Length of the string data to read. Must be an integer number between 1 and 4294967295, inclusive.

Result String

```
data %s
```

Where data is the application published string data to be read.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

This error means that the application event detector info key or other ID was not found.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

appl_reqinfo

Retrieves previously saved information from the Embedded Event Manager (EEM). This Tcl command extension provides support for retrieving information from EEM that has been previously saved with a unique

key, which must be specified in order to retrieve the information. Note that retrieving the information deletes it from EEM. It must be resaved if it is to be retrieved again.

Syntax

```
appl_reqinfo key ?
```

Arguments

key	(Mandatory) The string key of the data.

Result String

```
data %s
```

Where data is the application string data to be retrieved.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

This error means that the application event detector info key or other ID was not found.

appl_setinfo

Saves information in the Embedded Event Manager (EEM). This Tcl command extension provides support for saving information in the Embedded Event Manager that can be retrieved later by the same policy or by another policy. A unique key must be specified. This key allows the information to be retrieved later.

Syntax

```
appl_setinfo key ? data ?
```

Arguments

key	(Mandatory) The string key of the data.
data	(Mandatory) The application string data to save.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  8)    FH_EDUPLICATEKEY  (duplicate appl info key)
```

This error means that the application event detector info key or other ID was a duplicate.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 34)    FH_EMAXLEN  (maximum length exceeded)
```

This error means that the object length or number exceeded the maximum.

```
(_cerr_sub_err = 43)    FH_EBADLENGTH  (bad API length)
```

This error means that the API message length was invalid.

counter_modify

Modifies a counter value.

Syntax

```
counter_modify event_id ? val ? op nop|set|inc|dec
```

Arguments

event_id	(Mandatory) The counter event ID returned by the **register_counter** Tcl command extension. Must be an integer between 0 and 4294967295, inclusive.
val	(Mandatory) **Note** Mandatory except when the op nop argument value combination is specified. • If op is set, this argument represents the counter value that is to be set. • If op is inc, this argument is the value by which to increment the counter. • If op is dec, this argument is the value by which to decrement the counter.

op	(Mandatory) • nop--Retrieves the current counter value. • set--Sets the counter value to the given value. • inc--Increments the counter value by the given value. • dec--Decrements the counter value by the given value.

Result String

```
val_remain %d
```

Where val_remain is the current value of the counter.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 30)    FH_ECTBADOPER  (bad counter threshold operator)
```

This error means that the counter event detector set or modify operator was invalid.

description

Provides a brief description of the registered policy.

Syntax

```
description ?
```

Arguments

line	(Optional) Brief description of the policy consisting of 1 to 240 characters.

Result String

None

Set _cerrno

Yes

Sample Usage

The description statement is entered by the author of the policy. It can appear before or after any event registration statement in Tcl. The policy can have only one description.

Note Registration of a policy with more than one description statement will fail.

The following example shows how a brief description is provided for the **event_register_syslog** policy:

```
::cisco::eem::description "This Tcl command looks for the word count in syslog messages."
::cisco::eem::event_register_syslog tag 1 ...
::cisco::eem::event_register_snmp_object tag 2 ...
::cisco::eem::trigger {
       ::cisco::eem::correlate event 1 and event 2
       ::cisco::eem::attribute tag 1 occurs 1
       ::cisco::eem::attribute tag 2 occurs 1
}
```

fts_get_stamp

Returns the time period elapsed since the last software boot. Use this Tcl command extension to return the number of nanoseconds since boot in an array "nsec nnnn" where nnnn is the number of nanoseconds.

Syntax

```
fts_get_stamp
```

Arguments

None

Result String

```
nsec %d
```

Where nsec is the number of nanoseconds since boot.

Set _cerrno

No

register_counter

Registers a counter and returns a counter event ID. This Tcl command extension is used by a counter publisher to perform this registration before using the event ID to manipulate the counter.

Syntax

```
register_counter name ?
```

Arguments

name	(Mandatory) The name of the counter to be manipulated.

Result String

```
event_id %d
event_spec_id %d
```

Where event_id is the counter event ID for the specified counter; it can be used to manipulate the counter by the **unregister_counter** or **counter_modify** Tcl command extensions. The event_spec_id argument is the event specification ID for the specified counter.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  4)    FH_EINITONCE  (Init() is not yet done, or done twice.)
```

This error means that the request to register the specific event was made before the EEM event detector had completed its initialization.

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

This error means that the event type specified in the internal event specification was invalid.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 10)    FH_ECORRUPT  (internal EEM API context is corrupt)
```

This error means that the internal EEM API context structure is corrupt.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

This error means that the event ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 16)    FH_EBADFMPPTR  (bad ptr to fh_p data structure)
```

This error means that the context pointer that is used with each EEM API call is incorrect.

```
(_cerr_sub_err = 17)    FH_EBADADDRESS  (bad API control block address)
```

This error means that a control block address that was passed in the EEM API was incorrect.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 25)    FH_ESUBSEXCEED  (number of subscribers exceeded)
```

This error means that the number of timer or counter subscribers exceeded the maximum.

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

This error means that the subscriber index was invalid.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

This error means that the EEM event detector that handles this request is not available.

register_timer

Registers a timer and returns a timer event ID. This Tcl command extension is used by a timer publisher to perform this registration before using the event ID to manipulate the timer if it does not use the **event_register_timer** command extension to register as a publisher and subscriber.

Syntax

```
register_timer watchdog|countdown|absolute|cron name ?
```

Arguments

name	(Mandatory) The name of the timer to be manipulated.

Result String

```
event_id %u
```

Where event_id is the timer event ID for the specified timer (can be used to manipulate the timer by the **timer_arm** or **timer_cancel** command extensions).

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  4)    FH_EINITONCE  (Init() is not yet done, or done twice.)
```

This error means that the request to register the specific event was made before the EEM event detector had completed its initialization.

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

This error means that the event type specified in the internal event specification was invalid.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 10)    FH_ECORRUPT  (internal EEM API context is corrupt)
```

This error means that the internal EEM API context structure is corrupt.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 16)    FH_EBADFMPPTR  (bad ptr to fh_p data structure)
```

This error means that the context pointer that is used with each EEM API call is incorrect.

```
(_cerr_sub_err = 17)    FH_EBADADDRESS  (bad API control block address)
```

This error means that a control block address that was passed in the EEM API was incorrect.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 25)    FH_ESUBSEXCEED  (number of subscribers exceeded)
```

This error means that the number of timer or counter subscribers exceeded the maximum.

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

This error means that the subscriber index was invalid.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

This error means that the EEM event detector that handles this request is not available.

timer_arm

Arms a timer. The type could be CRON, watchdog, countdown, or absolute.

Syntax

```
timer_arm event_id ? cron_entry ?|time ?
```

Arguments

event_id	(Mandatory) The timer event ID returned by the **register_timer** command extension. Must be an integer between 0 and 4294967295, inclusive.
cron_entry	(Mandatory) Must exist if the timer type is CRON. Must not exist for other types of timer. CRON timer specification uses the format of the CRON table entry.
time	(Mandatory) Must exist if the timer type is not CRON. Must not exist if the timer type is CRON. For watchdog and countdown timers, the number of seconds and milliseconds until the timer expires; for an absolute timer, the calendar time of the expiration time (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). An absolute expiration date is the number of seconds and milliseconds since January 1, 1970. If the date specified has already passed, the timer expires immediately.

Result String

```
sec_remain %ld msec_remain %ld
```

Where sec_remain and msec_remain are the remaining time before the next expiration of the timer.

Note A value of 0 will be returned for the sec_remain and msec_remain arguments if the timer type is CRON.

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

This error means that the event type specified in the internal event specification was invalid.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

This error means that the event ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 27)    FH_ETMDELAYZR  (zero delay time)
```

This error means that the time specified to arm a timer was zero.

```
(_cerr_sub_err = 42)    FH_ENOTREGISTERED  (request for event spec that is unregistered)
```

This error means that the event was not registered.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

This error means that the EEM event detector that handles this request is not available.

timer_cancel

Cancels a timer.

Syntax

```
timer_cancel event_id ?
```

Arguments

event_id	(Mandatory) The timer event ID returned by the **register_timer** command extension. Must be an integer between 0 and 4294967295, inclusive.

Result String

```
sec_remain %ld msec_remain %ld
```

Where sec_remain and msec_remain are the remaining time before the next expiration of the timer.

Note A value of 0 will be returned for sec_remain and msec_remain if the timer type is CRON .

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

This error means that the event type specified in the internal event specification was invalid.

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

This error means that the application event detector info key or other ID was not found.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

This error means that the event ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

This error means that the event detector was unavailable.

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

This error means that the EEM event detector that handles this request is not available.

unregister_counter

Unregisters a counter. This Tcl command extension is used by a counter publisher to unregister a counter that was previously registered with the **register_counter** Tcl command extension.

Syntax

```
unregister_counter event_id ? event_spec_id ?
```

Arguments

event_id	(Mandatory) Counter event ID returned by the **register_counter** command extension. Must be an integer between 0 and 4294967295, inclusive.
event_spec_id	(Mandatory) Counter event specification ID for the specified counter returned by the **register_counter**command extension. Must be an integer between 0 and 4294967295, inclusive.

Result String

None

Set _cerrno

Yes

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

This error means that the operating system reported an error. The POSIX errno value that is reported with the error should be used to determine the cause of the operating system error.

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

This error means that an internal EEM request for memory failed.

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

This error means that the event specification ID could not be matched when the event was being registered or that an event detector internal event structure is corrupt.

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

This error means that an internal EEM event detector pointer was null when it should have contained a value.

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

This error means that the subscriber index was invalid.

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

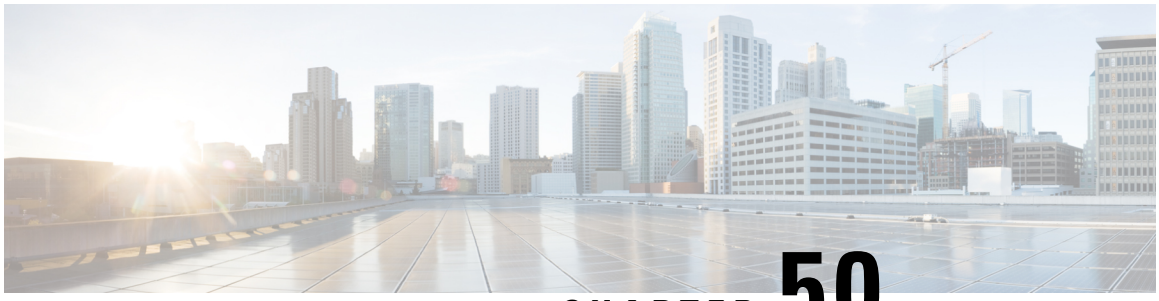
This error means that the event detector was unavailable.

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

This error means that the EEM event detector that handles this request is not available.

PART IX

Embedded Syslog Manager



CHAPTER 50

Embedded Syslog Manager (ESM)

The Embedded Syslog Manager (ESM) feature provides a programmable framework that allows you to filter, escalate, correlate, route, and customize system logging messages prior to delivery by the system message logger.

Restrictions for Embedded Syslog Manager

Embedded Syslog Manager (ESM) depends on the Tcl 8.3.4 Cisco IOS XE subsystem, because ESM filters are written in Tool Command Language (Tcl). ESM is available only in images that support Tcl version 8.3.4 or later versions. Support for Tcl 8.3.4 is added depending on your release.

ESM filters are written in Tcl.

ESM filtering cannot be applied to SNMP “history﷯? logging. Therefore, ESM filtering will not be applied to messages logged using the **logging history** and **snmp-server enable traps syslog** commands.

Information About the Embedded Syslog Manager

System Message Logging

With the introduction of the Embedded Syslog Manager, system messages can be logged independently as standard messages, XML-formatted messages, or ESM filtered messages. These outputs can be sent to any of the traditional syslog targets. For example, you could enable standard logging to the console connection, XML-formatted message logging to the buffer, and ESM filtered message logging to the monitor. Similarly, each type of output could be sent to different remote hosts. A benefit of separate logging processes is that standard logging will not be affected if, for example, there is some problem with the ESM filter modules.

System Logging Message Formatting

System logging messages are displayed in the following format:

%<facility>-<severity>-<mnemonic>: <message-text>

The following is an example of a system logging message:

```
%LINK-5-CHANGED: Interface Serial3/3, changed state to administratively down
```

Usually, these messages are preceded by additional text, such as the error sequence number and time stamp:

<sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

The following is an example of a system logging message preceded by an error sequence number and time stamp:

```
000013: Mar 18 14:52:10.039:%LINK-5-CHANGED: Interface Serial3/3, changed state
to administratively down
```

Note The time stamp format used in system logging messages is determined by the **service timestamps** global configuration mode command. The **service sequence-numbers** global configuration command enables or disables the leading sequence number. An asterisk (*) before the time indicates that the time may be incorrect because the system clock has not synchronized to a reliable time source.

Benefits of Embedded Syslog Manager

The Embedded Syslog Manager (ESM) is a feature integrated in Cisco software that allows complete control over system message logging at the source. ESM provides a programmatic interface to allow you to write custom filters that meet your specific needs relating to system logging. Benefits of this feature are:

- Customization--Fully customizable processing of system logging messages, with support for multiple, interfacing syslog collectors.
- Severity escalation for key messages--The ability to configure your own severity levels for syslog messages instead of using the system-defined severity levels.
- Specific message targeting--The ability to route specific messages or message types, based on type of facility or type of severity, to different syslog collectors.
- SMTP-base e-mail alerts--Capability for notifications using TCP to external servers, such as TCP-based syslog collectors or Simple Mail Transfer Protocol (SMTP) servers.
- Message limiting--The ability to limit and manage syslog “message storms” by correlating device-level events.

The ESM is not a replacement for the UDP-based syslog mechanism; instead, it is an optional subsystem that can operate in parallel with the current system logging process. For example, you can continue to have the original syslog message stream collected by server A, while the filtered, correlated, or otherwise customized ESM logging stream is sent to server B. All of the current targets for syslog messages (console, monitor,

buffer, and syslog host list) can be configured to receive either the original syslog stream or the ESM stream. The ESM stream can be further divided into user-defined streams and routed to collectors accordingly.

Syslog Filter Modules

Embedded Syslog Manager (ESM) uses syslog filter modules to process system logging messages. Syslog filter modules are scripts written in the Tool Command Language (Tcl) stored in local system memory or on a remote file server. The ESM is customizable because you can write and reference your own scripts.

Syslog filter modules can be written and stored as plain-text files or as precompiled files. Tcl script pre-compiling can be done with tools such as TclPro. Precompiled scripts allow a measure of security and managed consistency because they cannot be edited.

Note Because Tcl script modules contain executable commands, you should manage the security of these files using the same processes you use to manage configuration files.

How to Use the Embedded Syslog Manager

Writing ESM Syslog Filter Modules

Before referencing syslog filter modules in the Embedded Syslog Manager (ESM) configuration, you must write or obtain the modules you want to apply to system logging messages. Syslog filter modules can be stored in local system memory, or on a remote file server. Before you write syslog filter modules, you should understand the following concepts:

ESM Filter Process

When ESM is enabled, all system logging messages are processed through the referenced syslog filter modules. Syslog filter modules are processed in their order in the filter chain. The position of a syslog filter module in the filter chain is determined by the position tag applied in the **logging filter** global configuration mode command. If a position is not specified, the modules are processed in the order in which they were added to the configuration.

The output of each filter module is used as the input for the next filter module in the chain. Therefore, the Tcl global variable containing the original syslog message (::orig_msg) is set to the return value of each filter before invoking the next filter in the chain. Thus, if a filter returns NULL, no message will be sent out to the ESM stream. Once all filters have processed the message, the message is queued for distribution by the logger.

The console, buffer, monitor, and syslog hosts can be configured to receive a particular message stream (normal, XML, or ESM). The syslog hosts can be further restricted to receive user-defined numbered streams. Each target examines each message and accepts or rejects the message based on its stream tag. ESM filters can change the destination stream by altering the messages' stream tag by changing the Tcl global variable "::stream."

Syslog Filter Module Input

When Embedded Syslog Manager (ESM) is enabled, system logging messages are sent to the logging process. Each data element in the system logging message, and in the formatted syslog message as a whole, is recorded as a Tcl global variable. The data elements format for the syslog message are as follows:

<sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

The message-text will often contain message arguments.

When messages are received on a syslog host a “syslog-count” number is also added:

<syslog-count>: <sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

The following examples shows the syslog-count number included in the beginning of the sequence:

The table below lists the Tcl script input variables used in syslog filter modules. The syslog message data that the filter must operate on is passed as Tcl global namespace variables. Therefore, variables should be prefixed by a double-colon within the script module.

Standard ESM Filter Processing

Each time a system logging message is generated, the syslog filter modules are called in a series. This series is determined by the ::module_position variable, which in turn is typically the order in which the modules are referenced in the system configuration (the order in which they are configured).

The output of one filter module becomes the input to the next. Because the input to the filters is the Tcl global namespace variables, each filter can change any or all of these variables depending upon the purpose of the filter.

The only Tcl global variables that are automatically updated by the Embedded Syslog Manager (ESM) framework between subsequent filter executions are the ::orig_msg and ::cli_args variables. The framework automatically sets the value of ::orig_msg to the return value of the filter module. Thus a filter that is designed to alter or filter the original message must not manually set the value for the ::orig_msg variable; the filter only needs to return the desired value. For example, the following one-line ESM filter

```
return “This is my new syslog message.”
```

would ignore any message passed to it, and always change the output to the constant string “This is my new syslog message.” If the module was the last filter in the chain, all ESM targets would receive this string as the final syslog message.

The one-line ESM filter

```
return “”
```

would block all syslog messages to the ESM stream. For example, the line

```
return $::orig_msg
```

would do nothing but pass the message along to the next filter in the chain. Thus, an ESM filter designed to suppress unwanted messages would look something like this:

```
if { [my_procedure_to_check_this_message] == 1 } {
     return $::orig_msg
```

```
} else {
     return ""
}
```

Depending upon their design, some filters may not use the ::orig_msg variable at all, but rather reconstruct a syslog message from its data elements (using ::format_string, ::msg_args, ::timestamp, and so on). For example, an XML tagging filter will tag the individual data elements, and disregard the original formatted message. It is important for such modules to check the ::orig_msg variable at the beginning of the Tcl script, so that if a previous filter indicated that the message should not be sent out (::orig_msg is NULL), the message would not be processed, but return NULL also.

Commands can also be added to syslog filter modules using the **exec** and **config** Tcl commands. For example, if you wanted to add the source IP address to the syslog messages, and syslog messages were configured to be sent from the Ethernet 2/0 interface (using the **logging source-interface** command) you could issue the **show interface Ethernet 2/0** command during the module initialization by using the **exec** Tcl command within the script:

```
set source_ip_string [exec show ip int E2/0 | inc Internet]
puts $source_ip_string
"  Internet address is 10.4.2.63/24"
```

Background ESM Filter Processing

In Tcl, commands can be queued for future processing by using the **after** Tcl command. The most common use of this command is to correlate (gather and summarize) events over a fixed interval of time, called the "correlation window." Once the window of interest expires, the filter will need to "wake up," and calculate or summarize the events that occurred during the window, and often send out a new syslog message to report the events. This background process is handled by the ESM Event Loop process, which allows the Tcl interpreter to execute queued commands after a certain amount of time has passed.

If your syslog filter module needs to take advantage of correlation windows, it must use the **after** Tcl command to call a summary procedure once the correlation window expires (see examples in the "Configuration Examples for the Embedded Syslog Manager" section). Because there is no normal filter chain processing when background processes are run, in order to produce output these filters must use one of two ESM Tcl extensions: **errmsg** or **esm_errmsg**.

During background processing, the commands that have been queued by the **after** command are not run in the context of the filter chain (as in normal processing), but rather are autonomous procedures that are executed in series by the Tcl interpreter. Thus, these background procedures should not operate on the normal Tcl global namespace variables (except for setting the global namespace variables for the next filter when using **esm_errmsg**), but should operate on variables stored in their own namespace. If these variables are declared outside of a procedure definition, they will be persistent from call to call.

The purpose of the **errmsg** Tcl command is to create a new message and send it out for distribution, bypassing any other syslog filter modules. The syntax of the **errmsg** command is:

```
errmsg <severity> <stream> <message_string>
```

The purpose of the **esm_errmsg** Tcl command is to create a new message, process it with any syslog filter modules below it in the filter chain, and then send it out for distribution. The syntax of the **esm_errmsg** command is:

```
esm_errmsg <module_position>
```

The key difference between the errmsg() Tcl function and the esm_errmsg() Tcl function is that **errmsg** ignores the filters and directly queues a message for distribution, while **esm_errmsg** will send a syslog message down the chain of filters.

In the following example, a new syslog message is created and sent out tagged as Alert severity 1 to the configured ESM logging targets (stream 2). The purpose of this filter is to suppress the individual SYS-5-CONFIG messages over a thirty minute correlation window, and send out a summary message at the end of the window.

```
errmsg 1 2 "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
```

In order to use **esm_errmsg**, because the remaining filters following this one will be called, this background process must populate the needed Tool Command Language (Tcl) global namespace variables prior to calling **esm_errmsg**. Passing the ::module_position tells the ESM framework which filter to start with. Thus, filters using the **esm_errmsg** command should store their ::module_position (passed in the global namespace variables during normal processing) in their own namespace variable for use in background processing. Here is an example:

```
proc ::my_filter_namespace::my_summary_procedure{}
{
 set ::orig_msg "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
 set ::timestamp "*Jan 24 09:34:02.539"
 set ::severity 1
 set ::stream 2
 set ::traceback ""
 set ::pid ""
set ::process ""
set ::format_string "There have been %d configuration changes to the router
between %s and %s"
set ::msg_args {12 "Jan 24 09:04:01.539" "Jan 24 09:34:01.324"}
esm_errmsg $::my_filter_namespace::my_module_position
}
```

The benefit of setting all the global namespace variables for the **esm_errmsg** command is that your filters will be modular, and the order they are used in the ESM framework will not matter. For example, if you want all of the messages destined for the ESM targets to be suffixed with the message originator's hostname, you could write a one-line "hostname" filter and place it at the bottom of the filter chain:

```
return "$::orig_msg -- $::hostname"
```

In this example, if any of your filters generate new messages during background processing and they use **esm_errmsg** instead of **errmsg**, these messages will be clearly suffixed with the hostname.

What to Do Next

After creating your syslog filter module, you should store the file in a location accessible to the device. You can copy the file to local system memory, or store it on a network file server.

Configuring the Embedded Syslog Manager

To configure the Embedded Syslog Manager (ESM), specify one or more filters to be applied to generated syslog messages, and specify the syslog message target.

Before you begin

One or more syslog filter modules must be available to the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging filter** *filter-url* [*position*] [**args** *filter-arguments*]
4. Repeat Step 3 for each syslog filter module that should be applied to system logging output.
5. Enter one of the following:
   - **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*]
   - or
   - **logging host** {*ip-address* | *hostname*} **filtered** [**stream** *stream-id*]
6. Repeat Step 5 for each desired system logging destination.
7. **logging source-interface** *type number*
8. **logging origin-id** {**hostname** | **ip** | **ipv6** |**string** *user-defined-id*}
9. **end**
10. **show logging**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging filter** *filter-url* [*position*] [**args** *filter-arguments*] **Example:** `Device(config)# logging filter slot0:/escalate.tcl 1 args CONFIG_I 1`	Specifies one or more syslog filter modules to be applied to generated system logging messages. • Repeat this command for each syslog filter module that should be used. • The *filter-url* argument is the Cisco IOS File System location of the syslog filter module (script). The location can be in local memory, or a remote server using **tftp:**, **ftp:**, or **rcp:**.

	Command or Action	Purpose
		• The optional *position* argument specifies the order in which the syslog filter modules should be executed. If this argument is omitted, the specified module will be positioned as the last module in the chain. • Filters can be reordered quickly by again entering the **logging filter** command and specifying a different position. • The optional **args** *filter-arguments* syntax can be added to pass arguments to the specified filter. Multiple arguments can be specified. The number and type of arguments should be defined in the syslog filter module. For example, if the syslog filter module is designed to accept a specific e-mail address as an argument, you could pass the e-mail address using the **args user@host.com** syntax. Multiple arguments are typically delimited by spaces. • To remove a module from the list of modules to be executed, use the **no** form of this command.
Step 4	Repeat Step 3 for each syslog filter module that should be applied to system logging output.	--
Step 5	Enter one of the following: • **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*] • or • **logging host** {*ip-address* | *hostname*} **filtered** [**stream** *stream-id*] **Example:** `Device(config)# logging console filtered informational` **Example:** `Device(config)# logging host 209.165.200.225 filtered stream 20`	Specifies the target for ESM filtered syslog output. • ESM filtered syslog messages can be sent to the console, a monitor (TTY and Telnet connections), the system buffer, or remote hosts. • The optional *level* argument limits the sending of messages to those at or numerically lower than the specified value. For example, if level **1** is specified, only messages at level 1 (alerts) or level 0 (emergencies) will be sent to the specified target. The level can be specified as a keyword or number. • When you log to the console, monitor connection, or system buffer, the severity threshold specified by the *level* argument takes precedence over the ESM filtering. Even if the ESM filters return a message to be delivered to ESM targets, if the severity does not meet the configured threshold (is numerically higher than the level value), the message will not be delivered. • When you log to remote hosts, the stream tag allows you to specify a destination based on the type of message. The **stream** *stream-id* syntax allows you to configure the ESM to send only messages that have a specified stream value to a certain host.

	Command or Action	Purpose
		• The stream value is applied to messages by the configured syslog filter modules. For example, all Severity 5 messages could have a stream tag of "20" applied. You can then specify that all messages with a stream tag of "20" be sent to the host at 209.165.200.225.:
Step 6	Repeat Step 5 for each desired system logging destination.	• By issuing the logging host command multiple times, you can specify different targets for different system logging streams. • You can configure messages at different severity levels to be sent to the console, monitor connection, or system buffer. For example, you may want to display only important messages to the screen (using a monitor or console connection) at your network operations center (NOC).
Step 7	**logging source-interface** *type number* **Example:** `Device(config)# logging source-interface Gigabitethernet 0/0`	(Optional) Specifies the source interface for syslog messages sent to remote syslog hosts. • Normally, a syslog messages sent to remote hosts will use whatever interface is available at the time of the message generation. This command forces the device to send syslog messages to remote hosts only from the specified interface.
Step 8	**logging origin-id** {**hostname** | **ip** | **ipv6** |**string** *user-defined-id*} **Example:** `Device(config)# logging origin-id string "Domain 2, Router 5"`	(Optional) Allows you to add an origin identifier to syslog messages sent to remote hosts. • The origin identifier is added to the beginning of all syslog messages sent to remote hosts. The identifier can be the hostname, the IP address, or any text that you specify. • The origin identifier is useful for identifying the source of system logging messages in cases where you send syslog output from multiple devices to a single syslog host.
Step 9	**end** **Example:** `Device(config)# end`	Ends your current configuration session and returns the CLI to privileged EXEC mode.
Step 10	**show logging** **Example:** `Device# show logging`	(Optional) Displays the status of system logging, including the status of ESM filtered logging. • If filtered logging to the buffer is enabled, this command also shows the data stored in the buffer.

	Command or Action	Purpose
		• The order in which syslog filter modules are listed in the output of this command is the order in which the filter modules are executed.

Configuration Examples for the Embedded Syslog Manager

Example: Configuring the Embedded Syslog Manager Example

In the following example, the Embedded Syslog Manager (ESM) filter logging is enabled for the console connection, standard logging is enabled for the monitor connection and for the buffer, and XML-formatted logging is enabled for the host at 209.165.200.225:

```
Device(config)# logging filter tftp://209.165.200.225/ESM/escalate.tcl
Device(config)# logging filter slot0:/email.tcl user@example.com
Device(config)# logging filter slot0:/email_guts.tcl
Device(config)# logging console filtered
Device(config)# logging monitor 4
Device(config)# logging buffered debugging
Device(config)# logging host 209.165.200.225 xml
Device(config)# end

Device# show logging
Syslog logging: enabled (0 messages dropped, 8 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering enabled)
    Console logging: level debugging, 21 messages logged, xml disabled,
                     filtering enabled
    Monitor logging: level warnings , 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 30 messages logged, xml disabled,
                     filtering disabled
    Logging Exception size (8192 bytes)
    Count and timestamp logging messages: disabled

Filter modules:
    tftp://209.165.200.225/ESM/escalate.tcl
    slot0:/email.tcl user@example.com

    Trap logging: level informational, 0 message lines logged
        Logging to 209.165.200.225, 0 message lines logged, xml enabled,
              filtering disabled

Log Buffer (8192 bytes):

*Jan 24 09:34:28.431: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:34:51.555: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:49:44.295: %SYS-5-CONFIG_I: Configured from console by console
Device#
```

Example: Syslog Filter Module

Syslog Script Modules are Tcl scripts. The following examples are provided to assist you in developing your own Syslog Script Modules.

Note These script modules are provided as examples only, and are not supported by Cisco. No guarantees, expressed or implied, are provided for the functionality or impact of these scripts.

Example: Severity Escalation

This ESM syslog filter module example watches for a single mnemonic (supplied via the first CLI argument) and escalates the severity of the message to that specified by the second CLI argument.

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Severity Escalation Filter                          ||||      ||||
#                                                  ..:||||||:..:||||||:..
#                                                 ------------------------
#                                                 C i s c o  S y s t e m s
#  ====================================================================
#
# Usage: Set CLI Args to "mnemonic new_severity"
#
# Namespace: global
# Check for null message
if { [string length $::orig_msg] == 0} {
   return ""
}

if { [info exists ::cli_args] } {
    set args [split $::cli_args]
    if { [ string compare -nocase [lindex $args 0] $::mnemonic ] == 0 } {
        set ::severity [lindex $args 1]
        set sev_index [ string first [lindex $args 0] $::orig_msg ]
        if {  $sev_index >= 2 } {
           incr sev_index -2
           return [string replace $::orig_msg $sev_index $sev_index \
              [lindex $args 1]]
        }
    }
}
return $::orig_msg
```

Example: Message Counting

This ESM syslog filter module example is divided into two files for readability. The first file allows the user to configure those messages that they want to count and how often to summarize (correlation window) by populating the msg_to_watch array. The actual procedures are in the counting_guts.tcl file. Note the use of the separate namespace “counting” to avoid conflict with other ESM filters that may also perform background processing.

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
```

```
#  Message Counting Filter                          ||||      ||||
#                                                ..:||||||:..:||||||:..
#                                               -----------------------
#                                               C i s c o  S y s t e m s
# ============================================================================


#
# Usage:
# 1) Define the location for the counting_guts.tcl script
#
# 2) Define message categories to count and how often to dump them (sec)
#    by populating the "msg_to_watch" array below.
#    Here we define category as facility-severity-mnemonic
#    Change dump time to 0 to disable counting for that category
#
# Namespace: counting
namespace eval ::counting {
    set sub_script_url tftp://172.16.0.0/12/ESM/counting_guts.tcl
    array set msg_to_watch {
        SYS-5-CONFIG_I            5
    }
#  ====================== End User Setup ==============================
# Initialize processes for counting
    if { [info exists init] == 0 } {
       source $sub_script_url
       set position $module_position
    }
# Process the message
process_category
} ;# end namespace counting
```

Message Counting Support Module (counting_guts.tcl)

```
#  ====================================================================
#  Embedded Syslog Manager                            ||        ||
#                                                     ||        ||
#  Message Counting Support Module                   ||||      ||||
#                                                 ..:||||||:..:||||||:..
#  (No User Modification)                        -----------------------
#                                                C i s c o  S y s t e m s
# ============================================================================



namespace eval ::counting {

# namespace variables

array set cat_msg_sev {}
array set cat_msg_traceback {}
array set cat_msg_pid {}
array set cat_msg_proc {}
array set cat_msg_ts {}
array set cat_msg_buginfseq {}
array set cat_msg_name {}
array set cat_msg_fac {}
array set cat_msg_format {}
array set cat_msg_args {}
array set cat_msg_count {}
array set cat_msg_dump_ts {}
```

```
# Should I count this message ?
   proc query_category {cat} {
        variable msg_to_watch
        if { [info exists msg_to_watch($cat)] } {
            return $msg_to_watch($cat)
        } else {
            return 0
        }
    }
   proc clear_category {index} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        unset cat_msg_sev($index) cat_msg_traceback($index) cat_msg_pid($index)\
              cat_msg_proc($index) cat_msg_ts($index) \
              cat_msg_buginfseq($index)  cat_msg_name($index) \
              cat_msg_fac($index) cat_msg_format($index) cat_msg_args($index)\
              cat_msg_count($index) cat_msg_dump_ts($index)
    }
# send out the counted messages
   proc dump_category {category} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        variable poll_interval
        set dump_timestamp [cisco_service_timestamp]
foreach index [array names cat_msg_count $category] {
      set fsm "$cat_msg_fac($index)-$cat_msg_sev($index)-$cat_msg_name($index)"
      set ::orig_msg \
        [format "%s%s: %%%s: %s %s %s %s - (%d occurence(s) between %s and %s)"\
        $cat_msg_buginfseq($index)\
          $dump_timestamp\
          $fsm \
          [uplevel 1 [linsert $cat_msg_args($index) 0 ::format
$cat_msg_format($index) ]] \
          $cat_msg_pid($index) \
          $cat_msg_proc($index) \
          $cat_msg_traceback($index) \
          $cat_msg_count($index) \
          $cat_msg_ts($index) \
          $dump_timestamp]
# Prepare for remaining ESM filters
               set ::severity $cat_msg_sev($index)
               set ::traceback $cat_msg_traceback($index)
               set ::pid $cat_msg_pid($index)
```

```
                set ::process $cat_msg_proc($index)
                set ::timestamp $cat_msg_ts($index)
                set ::buginfseq $cat_msg_buginfseq($index)
                set ::mnemonic $cat_msg_name($index)
                set ::facility $cat_msg_fac($index)
                set ::format_string $cat_msg_format($index)
                set ::msg_args [split $cat_msg_args($index)]
                esm_errmsg $counting::position
                clear_category $index
        }
  }
# See if this message already has come through since the last dump.
# If so, increment the count, otherwise store it.
   proc process_category {} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        if { [string length $::orig_msg] == 0} {
           return ""
        }
        set category "$::facility-$::severity-$::mnemonic"
        set correlation_window [expr [ query_category $category ] * 1000]
        if { $correlation_window == 0 } {
            return $::orig_msg
        }
        set message_args [join $::msg_args]
        set index "$category,[lindex $::msg_args 0]"
        if { [info exists cat_msg_count($index)] } {
                 incr cat_msg_count($index)
         } else {
              set cat_msg_sev($index) $::severity
              set cat_msg_traceback($index) $::traceback
              set cat_msg_pid($index) $::pid
              set cat_msg_proc($index) $::process
              set cat_msg_ts($index) $::timestamp
              set cat_msg_buginfseq($index) $::buginfseq
              set cat_msg_name($index) $::mnemonic
              set cat_msg_fac($index) $::facility
              set cat_msg_format($index) $::format_string
              set cat_msg_args($index) $message_args
              set cat_msg_count($index) 1
              set cat_msg_dump_ts($index) [clock seconds]
              catch [after $correlation_window counting::dump_category $index]
         }
         return ""
   }
# Initialized
set init 1
} ;#end namespace counting
```

Example: XML Tagging

This ESM syslog filter module applies user-defined XML tags to syslog messages:

```
#  ========================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  XML Tagging Filter                                  ||||      ||||
#                                                    ..:||||||:..:||||||:..
#                                                   -----------------------
#                                                   C i s c o  S y s t e m s
#  ========================================================================
#
# Usage: Define desired tags below.
#
# Namespace: xml
# Check for null message
    if { [string length $::orig_msg] == 0} {
       return ""
    }
namespace eval xml {
####  define tags ####
set MSG_OPEN "<ios-log-msg>"
set MSG_CLOSE "</ios-log-msg>"
set FAC_OPEN   "<facility>"
set FAC_CLOSE  "</facility>"
set SEV_OPEN   "<severity>"
set SEV_CLOSE  "</severity>"
set MNE_OPEN   "<msg-id>"
set MNE_CLOSE  "</msg-id>"
set SEQ_OPEN   "<seq>"
set SEQ_CLOSE  "</seq>"
set TIME_OPEN  "<time>"
set TIME_CLOSE "</time>"
set ARGS_OPEN  "<args>"
set ARGS_CLOSE "</args>"
set ARG_ID_OPEN "<arg id="
set ARG_ID_CLOSE "</arg>"
set PROC_OPEN "<proc>"
set PROC_CLOSE "</proc>"
set PID_OPEN "<pid>"
set PID_CLOSE "</pid>"
set TRACE_OPEN "<trace>"
set TRACE_CLOSE "</trace>"
#  ======================= End User Setup ==============================
#### clear result ####
set result ""
#### message opening, facility, severity, and name ####
append result $MSG_OPEN $FAC_OPEN $::facility $FAC_CLOSE $SEV_OPEN $::severity
$SEV_CLOSE $MNE_OPEN $::mnemonic $MNE_CLOSE
#### buginf sequence numbers ####
if { [string length $::buginfseq ] > 0 } {
    append result $SEQ_OPEN $::buginfseq $SEQ_CLOSE
}
#### timestamps ####
if { [string length $::timestamp ] > 0 } {
    append result $TIME_OPEN $::timestamp $TIME_CLOSE
}
#### message args ####
if { [info exists ::msg_args] } {
    if { [llength ::msg_args] > 0 } {
        set i 0
        append result $ARGS_OPEN
        foreach arg $::msg_args {
            append result $ARG_ID_OPEN $i ">" $arg $ARG_ID_CLOSE
                incr i
        }
```

```
        append result $ARGS_CLOSE
    }
}
#### traceback ####
if { [string length $::traceback ] > 0 } {
    append result $TRACE_OPEN $::traceback $TRACE_CLOSE
}
#### process ####
if { [string length $::process ] > 0 } {
    append result $PROC_OPEN $::process $PROC_CLOSE
}
#### pid ####
if { [string length $::pid ] > 0 } {
    append result $PID_OPEN $::pid $PID_CLOSE
}
#### message close ####
append result $MSG_CLOSE
return "$result"
} ;# end namespace xml
```

Example: SMTP-Based E-Mail Alert

This ESM syslog filter module example watches for configuration messages and sends them to the e-mail address supplied as a CLI argument. This filter is divided into two files. The first file implements the filter, and the second file implements the Simple Mail Transfer Protocol (SMTP) client.

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Email Filter                                        ||||      ||||
#  (Configuration Change Warning)                 ..:||||||:..:||||||:..
#                                                ------------------------
#                                                 C i s c o  S y s t e m s
#   ====================================================================
# Usage:  Provide email address as CLI argument.  Set email server IP in
#         email_guts.tcl
#
# Namespace: email
if { [info exists email::init] == 0 } {
   source tftp://123.123.123.123/ESM/email_guts.tcl
}
# Check for null message
if { [string length $::orig_msg] == 0} {
      return ""
   }
if { [info exists ::msg_args] } {
    if { [string compare -nocase CONFIG_I $::mnemonic ] == 0 } {
                email::sendmessage $::cli_args $::mnemonic \
                [string trim $::orig_msg]
    }
}
return $::orig_msg
```

E-Mail Support Module (email_guts.tcl)

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Email Support Module                                ||||      ||||
#                                                   ..:||||||:..:||||||:..
#                                                  ------------------------
```

```
#                                                 C i s c o  S y s t e m s
#   ================================================================
#
# Usage: Set email host IP, from, and friendly strings below.
#
namespace eval email {
    set sendmail(smtphost)172.16.0.1
    set sendmail(from) $::hostname
    set sendmail(friendly) $::hostname
    proc sendmessage {toList subject body} {
        variable sendmail
        set smtphost $sendmail(smtphost)
        set from $sendmail(from)
        set friendly $sendmail(friendly)
        set sockid [socket $smtphost 25]
## DEBUG
set status [catch {
        puts $sockid "HELO $smtphost"
        flush $sockid
        set result [gets $sockid]
        puts $sockid "MAIL From:<$from>"
        flush $sockid
        set result [gets $sockid]
        foreach to $toList {
            puts $sockid "RCPT To:<$to>"
            flush $sockid
        }
        set result [gets $sockid]
        puts  $sockid "DATA "
        flush $sockid
        set result [gets  $sockid]
        puts  $sockid "From: $friendly <$from>"
        foreach to $toList {
            puts $sockid "To:<$to>"
        }
        puts  $sockid "Subject: $subject"
        puts  $sockid "\n"
        foreach line [split $body "\n"] {
            puts  $sockid " $line"
        }
        puts  $sockid "."
        puts  $sockid "QUIT"
        flush $sockid
        set result [gets  $sockid]
} result]
        catch {close $sockid }
    if {$status} then {
        return -code error $result
    }
}
} ;# end namespace email
set email::init 1
```

Example: Stream

This ESM syslog filter module example watches for a given facility (first CLI argument) and routes these messages to a given stream (second CLI argument:.

```
#  ===================================================================
#  Embedded Syslog Manager                              ||         ||
#                                                       ||         ||
#  Stream Filter (Facility)                            ||||       ||||
#                                                  ..:||||||:..:||||||:..
```

```
#                                                 -----------------------
#                                                  C i s c o  S y s t e m s
#   ================================================================
# Usage:  Provide facility and stream as CLI arguments.
#
# Namespace: global
# Check for null message
#  ======================= End User Setup ==============================
set args [split $::cli_args]
if { [info exists ::msg_args] } {
    if { $::facility == [lindex $args 0] } {
                set ::stream [lindex $args 1]
    }
}
return $::orig_msg}
```

Example: Source IP Tagging

The **logging source-interface** CLI command can be used to specify a source IP address in all syslog packets sent from the device. The following syslog filter module example demonstrates the use of **show** CLI commands (**show running-config** and **show ip interface** in this case) within a filter module to add the source IP address to syslog messages. The script looks for the local namespace variable “source_ip::init” first. If the variable is not defined in the first syslog message processed, the filter will run the **show** commands and use regular expressions to get the source interface and then its IP address.

Note that in this script, the **show** commands are run only once. If the source interface or its IP address were to be changed, the filter would have to be reinitialized to pick up the new information. (You could have the show commands run on every syslog message, but this would not scale well.)

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Source IP Module                                    ||||      ||||
#                                                   ..:||||||:..:||||||:..
#                                                  -----------------------
#                                                   C i s c o  S y s t e m s
#   ================================================================
# Usage: Adds Logging Source Interface IP address to all messages.
#
# Namespace:source_ip
#
#  ======================= End User Setup ==============================
namespace eval ::source_ip {
    if { [info exists init] == 0 } {
       if { [catch {regexp {^logging source-interface (.*$)} [exec show
run | inc logging source-interface] match source_int}]} {
            set suffix "No source interface specified"
       } elseif { [catch {regexp {Internet address is (.*)/.*$} [exec
show ip int $source_int | inc Internet] match ip_addr}]} {
            set suffix "No IP address configured for source interface"
       } else {
            set suffix $ip_addr
       }
       set init 1
    }

    if { [string length $::orig_msg] == 0} {
      return ""
    }
    return "$::orig_msg - $suffix"
} ;# end namespace source_ip
```

Additional References for the Embedded Syslog Manager

Related Documents

Related Topic	Document Title
Cisco IOS XE Commands	Command Lookup Tool
System Message Logging	Troubleshooting and Fault Management module
XML Formatted System Message Logging	XML Interface to Syslog Messages module
Tcl 8.3.4 Support in Cisco Software	*Cisco IOS Scripting with Tcl* module
Network Management commands (including logging commands): complete command syntax, defaults, command mode, command history, usage guidelines, and examples	*Cisco IOS Network Management Command Reference*

Standards and RFCs

Standard/RFC	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--
RFC-3164	*The BSD Syslog Protocol* This RFC is informational only. The Cisco implementation of syslog does not claim full compliance with the protocol guidelines mentioned in this RFC. Not all supported RFCs are listed.

MIBs

MIB	MIBs Link
No new or modified standards are supported, and support for existing standards has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for the Embedded Syslog Manager

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 78: Feature Information for the Embedded Syslog Manager

Feature Name	Releases	Feature Information
Embedded Syslog Manager		The Embedded Syslog Manager (ESM) feature provides a programmable framework that allows you to filter, escalate, correlate, route, and customize system logging messages prior to delivery by the Cisco IOS system message logger.

Glossary

Note Refer to the "Internetworking Terms and Acronyms" section for terms not included in this glossary.

console--Specifies the connection (CTY or console line) to the console port of the device. Typically, this is a terminal attached directly to the console port, or a PC with a terminal emulation program. Corresponds to the **show terminal** command.

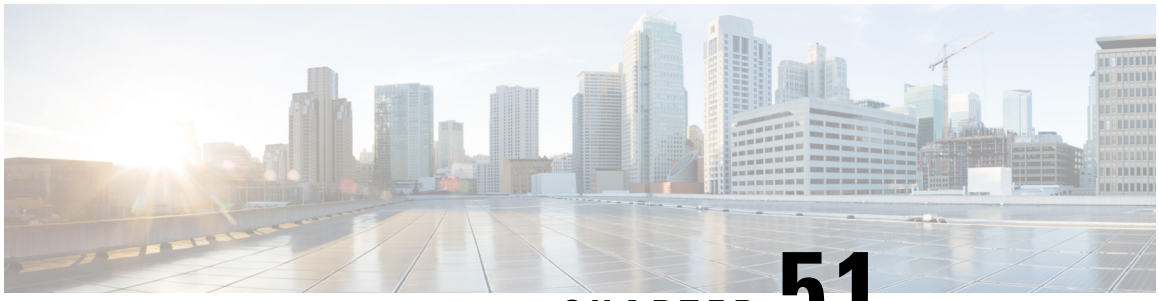
monitor--Specifies the TTY (TeleTYpe terminal) line connection at a line port. In other words, the “monitor” keyword corresponds to a terminal line connection or a Telnet (terminal emulation) connection. TTY lines (also called ports) communicate with peripheral devices such as terminals, modems, and serial printers. An example of a TTY connection is a PC with a terminal emulation program connected to the device using a dialup modem.

SEMs--Abbreviation for system error messages. “System error messages” is the term formerly used for messages generated by the system logging (syslog) process. Syslog messages use a standardized format, and come in eight severity levels, from “emergencies” (level 0) to “debugging” (level 7). The term “system error

message" is actually misleading, because these messages can include notifications of device activity beyond "errors" (such as informational notices).

syslog--Abbreviation for the system message logging process in Cisco software. Also used to identify the messages generated, as in "syslog messages." Technically, the term "syslog" refers only to the process of logging messages to a remote host or hosts, but is commonly used to refer to all Cisco system logging processes.

trap--A trigger in the system software for sending error messages. "Trap logging" means logging messages to a remote host. The remote host is actually a syslog host from the perspective of the device sending the trap messages, but because the receiving device typically provides collected syslog data to other devices, the receiving device is also referred to as a "syslog server."



CHAPTER 51

Logging to Local Nonvolatile Storage

The Logging to Local Nonvolatile Storage feature enables system logging messages to be saved on an advanced technology attachment flash disk. Messages saved on bootflash or a harddisk persist after a device is rebooted.

Prerequisites for Logging to Local Nonvolatile Storage

The logging buffered Command Must Be Enabled

Before the Logging to Local Nonvolatile Storage feature can be enabled with the **logging persistent** command, you must enable the logging of messages to an internal buffer with the **logging buffered** command. For additional information, see the "Writing Logging Messages to Bootflash or a Harddisk" section.

Restrictions for Logging to Local Nonvolatile Storage

Available Bootflash or Harddisk Space Constrains the Size and Number of Stored Log Files

The amount of bootflash or harddisk space allocated to system logging messages constrains the number of logging files that can be stored. When the allocation threshold is passed, the oldest log file in the directory is deleted to make room for new system logging messages. To permanently store system logging messages, you must archive them to an external device. For more information, see “Copying Logging Messages to an External Disk” section.

Note Logging to local nonvolatile storage can use up to 2 GB of storage space.

Information About Logging to Local Nonvolatile Storage

System Logging Messages

System logging messages include error and debug messages generated by application programming interfaces (APIs) on the device. Typically, logging messages are stored in a device's memory buffer; when the buffer is full, older messages are overwritten by new messages. All logging messages are erased from the memory buffer when the device reboots.

How to Configure Logging to Local Nonvolatile Storage

Writing Logging Messages to Bootflash or a Harddisk

Perform this task to enable the Logging to Local Nonvolatile Storage feature and write logging messages to bootflash or a harddisk.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging buffered** [*buffer-size* | *severity-level*]
4. **logging persistent** [**url harddisk:**/*directory*] [**size** *filesystem-size*] [**filesize** *logging-file-size*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enables global configuration mode.
Step 3	**logging buffered** [*buffer-size* | *severity-level*] **Example:** `Device(config)# logging buffered`	Enables system message logging to a local buffer and limits messages logged to the buffer based on severity. • The optional *buffer-size* argument specifies the size of the buffer. Range is from 4096 to 4294967295. The default size varies by platform. • The optional *severity-level* argument limits the logging of messages to the buffer to those no less severe than the specified level.

	Command or Action	Purpose
Step 4	**logging persistent** [**url harddisk:**/*directory*] [**size** *filesystem-size*] [**filesize** *logging-file-size*] **Example:** `Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384` **Note** The default value is: url: bootflash:/syslog filesystem-size: 10% of total disk space logging-file-size: 262144	Writes logging messages from the memory buffer to the specified directory on the device's bootflash or a harddisk. • Before logging messages are written to a file on the bootflash or harddisk, the Cisco software checks to see if there is sufficient disk space. If not, the oldest file of logging messages (by timestamp) is deleted, and the current file is saved. • The filename format of log files is log_MM:DD:YYYY::hh:mm:ss. For example: log_11:26:2012::01:01:41. **Note** This feature supports only one log file per second due to its filename format, which contains a timestamp suffix down to the seconds level. **Note** The defaults for this command are as follows: • **url: bootflash:/syslog** Filesystem-size: 10% of total disk space. Logging-file-size: 262144

Copying Logging Messages to an External Disk

Perform this task to copy logging messages from the bootflash or a harddisk to an external disk.

SUMMARY STEPS

1. **enable**
2. **copy** *source-url destination-url*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** *source-url destination-url* **Example:** `Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog`	Copies the specified file or directory on the bootflash or a harddisk via FTP to the specified URL.

Configuration Examples for Logging to Local Nonvolatile Storage

Example: Writing Logging Messages to Bootflash or a Harddisk

The following example shows how to write up to 134217728 bytes (128 MB) of logging messages to the syslog directory of disk 0, specifying a file size of 16384 bytes:

```
Device(config)# logging buffered
Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384
```

Example: Copying Logging Messages to an External Disk

The following example shows how to copy logging messages from the device's bootflash or harddisk to an external disk:

```
Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog
```

Additional References

Related Documents

Related Topic	Document Title
copy command	Cisco IOS Configuration Fundamentals Command Reference
Network management commands (including logging commands): complete command syntax, defaults, command mode, command history, usage guidelines, and examples	Cisco IOS Network Management Command Reference

MIBs

MIBs	MIBs Link
• No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

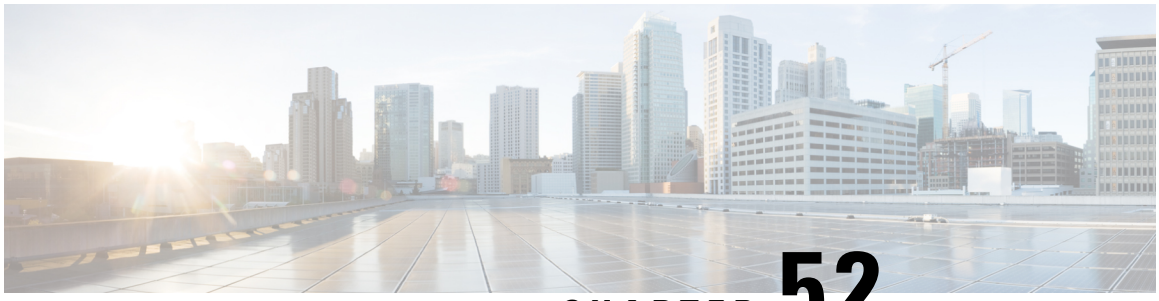
Feature Information for Logging to Local Nonvolatile Storage

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 79: Feature Information for Logging to Local Nonvolatile Storage

Feature Name	Releases	Feature Information
Logging to Local Nonvolatile Storage	Cisco IOS XE Release 2.1	The Logging to Local Nonvolatile Storage feature enables system logging messages to be saved on an advanced technology attachment flash disk. Messages saved on bootflash or a harddisk persist after a device is rebooted. The following command was introduced or modified: **logging persistent**.



CHAPTER 52

Reliable Delivery and Filtering for Syslog

The Reliable Delivery and Filtering for Syslog feature allows a device to be customized for receipt of syslog messages. This feature provides reliable and secure delivery for syslog messages using Blocks Extensible Exchange Protocol (BEEP). Additionally, it allows multiple sessions to a single logging host, independent of the underlying transport method, and provides a filtering mechanism called a message discriminator.

This module describes the functions of the Reliable Delivery and Filtering for Syslog feature and how to configure them in a network.

Prerequisites for Reliable Delivery and Filtering for Syslog

- The device level rate limit is set to meet business needs, network traffic requirements, or performance requirements.
- Each BEEP session must have an RFC 3195-compliant syslog-RAW exchange profile.
- A Simple Authentication and Security Layer (SASL) profile specifying “DIGEST-MD5” for provisioning services must be established when a crypto image is used.
- Syslog servers must be compatible with BEEP.
- Syslog server applications must be capable of handling multiple sessions to use the multiple session capability of the Reliable Delivery and Filtering for Syslog feature.

Restrictions for Reliable Delivery and Filtering for Syslog

- Only the syslog-RAW, SASL, and Transport Layer Security (TLS) profiles are supported.
- Both ends of a syslog session must use the same transport method.

- A message discriminator must be defined before it can be associated with a specific syslog session.
- A syslog session can be associated with only one message discriminator.
- Message delivery with User Datagram Protocol (UDP) will be faster than with either TCP or BEEP.

Information About Reliable Delivery and Filtering for Syslog

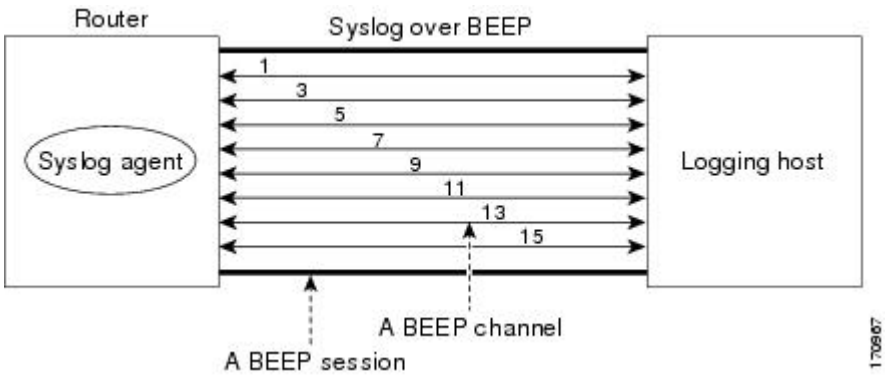
BEEP Transport Support

BEEP is a generic application protocol framework for connection-oriented, asynchronous interactions. It is intended to provide the features that traditionally have been duplicated in various protocol implementations. BEEP typically runs on top of TCP and allows the exchange of messages. Unlike HTTP and similar protocols, either end of the connection can send a message at any time. BEEP also includes facilities for encryption and authentication and is highly extensible.

BEEP as a transport protocol for syslog messages provides multiple channels. Each channel can be configured for a separate session to the same host. BEEP provides reliable transport. Syslog messages sent over a BEEP connection are guaranteed to be delivered in sequence.

With command-line interface (CLI) commands introduced in the Reliable Delivery and Filtering for Syslog feature, you can configure a new BEEP session to have a maximum of eight channels.

The figure below shows a BEEP session with eight channels, allowing eight separate syslog sessions.



Channels are identified as 1, 3, 5, 7, 9, 11, 13, and 15. The number of available channels (eight) was designed to correspond to the number of severity levels of classic RFC-3164 syslog messages (0 to 7). Message discriminators can be used such that severity levels are mapped to BEEP channels. An intelligent BEEP syslog server (depending upon the BEEP stack used) could use this mapping to prioritize messages with higher severity (see RFC 3081, section 3.1.4). Unless associated with a message discriminator, all syslog sessions (channels) receive all syslog messages.

Syslog Message

A syslog message has a sequence number that allows the host to use the number as an identifier for the message as well as to detect whether there were any gaps in the messages that were received. Syslog messages are numbered consecutively. The reliability of BEEP does not replace the need for sequence numbers, which are required for the following reasons:

- A sequence number provides an easy way to identify a syslog message. Independent of reliability considerations, the sequence number serves as a message identifier.
- A BEEP session may not be in place for the entire time that a device sending syslog messages is up. Sequence numbers provide a way for management applications to assess whether messages were missed between BEEP sessions.
- BEEP is only one of several transports. Unreliable transports are also used and the syslog protocol should not rely on a reliable transport always being provided.

The existing numbering scheme for syslog messages is limited with the extension of syslog to accommodate advanced message discrimination features and multiple hosts. Message discrimination leads to gaps in the sequence numbers, meaning that hosts lose the ability to detect whether they have missed a message. If syslog messages are numbered consecutively on each session to avoid the gaps in sequence numbers, it will not be possible to easily correlate which messages are the same and which ones are different because the sequence number would no longer uniquely identify a message.

To separate identification from sequencing and reliability, the following changes to syslog messages were made:

- The sequence number is retained as an identifier for the message. Messages with a lower number precede messages with a higher number, but they are not guaranteed to be consecutive.
- An additional field is added in the body portion of a syslog message to help ensure sequencing. The contents of this field contain a sequence number for a particular session. The same message transmitted over different sessions may have a different sequence number.

Syslog Session

A syslog session is a logical link from the syslog agent on a device to the recipient of a syslog message. For example, a syslog session can be established between a syslog agent and any of the following:

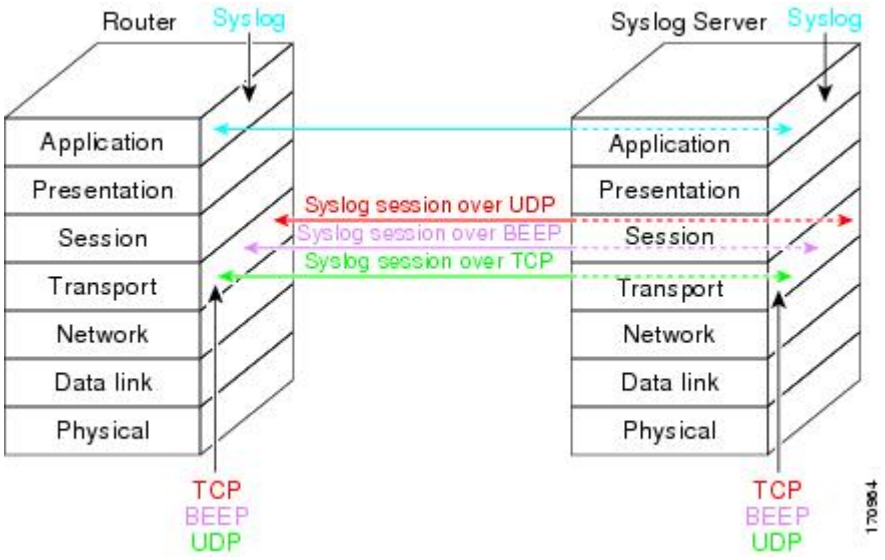
- Device console
- Device logging buffer
- Device monitor
- External syslog server

A syslog session runs over a transport connection between the syslog source and the syslog destination. A transport connection can use any of the following protocols:

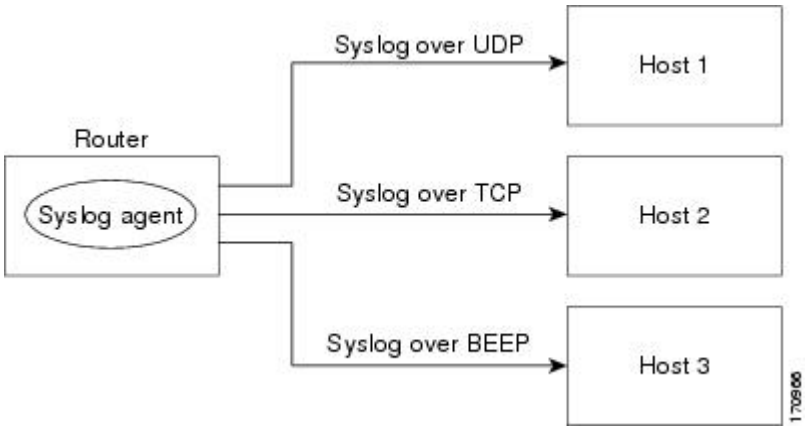
- TCP
- UDP (association to one remote address and port)
- BEEP (channel within a BEEP session)

The figure below shows a mapping of syslog sessions and transport protocols between a device and a syslog server using an Open Systems Interconnection (OSI) model.

Note The figure below is best viewed using Internet Explorer.



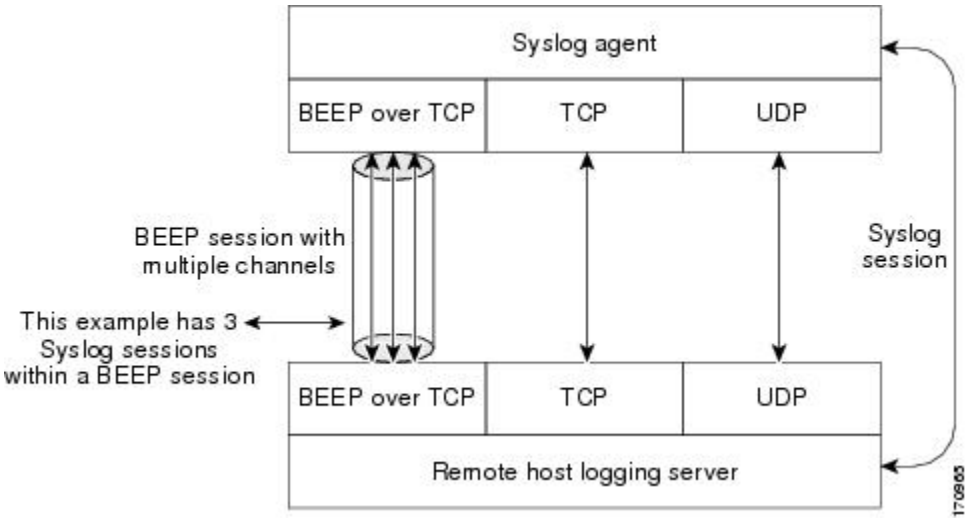
The figure below shows multiple syslog sessions from a single syslog agent to different hosts using UDP, TCP and BEEP.



Multiple Syslog Sessions

A syslog session is independent of a transport connection. A Cisco device can support multiple syslog sessions, each running over its own transport connection. Multiple syslog sessions cannot share the same transport connection, but multiple syslog sessions may terminate at the same remote host, each running over its own transport connection. An example is a BEEP session in which multiple channels are used.

The figure below shows an end-to-end view of a syslog session. Note the three syslog sessions within a single BEEP session.



The TCP and UDP protocols do not have multiplexed channels but the protocols do allow for using multiple ports to establish multiple syslog sessions to the same syslog host. To enable the UDP and TCP transport methods to have capability similar to BEEP's multiple channel capability, the Reliable Delivery and Filtering for Syslog feature allows multiple syslog sessions to be established via the UDP and TCP transport methods to the same logging host. Multiple syslog sessions going over BEEP sessions is also supported.

Message Discriminator

A message discriminator is a syslog processor. A message discriminator is associated with a syslog session and binds that session to a transport connection.

Prior to message delivery, the message is subject to the message discriminator with a user-specified list of criteria. After the first filtering criterion results in a message being blocked, the filtering check stops.

Note The sequence of criteria in the CLI does not affect the sequence in which criteria is checked.

- Following are filtering criteria. These criteria are checked in the order listed here:
  - Severity level or levels specified
  - Facility within the message body that matches a regular expression
  - Mnemonic that matches a regular expression
  - Part of the body of a message that matches a regular expression

A message discriminator offers the following capabilities:

- Optional rate limiting--Specifying a transmission rate of messages per time interval that is not to be exceeded. If the rate limit is exceeded, messages are either delayed or dropped, at the discretion of the device. The application of a rate limiter means that reliable delivery of syslog messages over that syslog session is no longer guaranteed. The purpose of a rate limiter is to avoid potential "flooding" at recipient syslog servers for applications that do not require guaranteed syslog delivery.
- Correlating--Inspecting candidate event messages and possibly aggregating information across events, creating a new event that contains the aggregated information. Correlating functions include:

- Elimination of duplicate messages by maintaining a message count and waiting a specific time period between sending the first message of a certain type and sending the next message of that type
- Elimination of oscillating messages
- Simple message correlation; for example, if one message is a symptom of a cause reported by another message, one consolidated message is reported

A message discriminator can be associated with a specific destination and transport; that is, the filter can be host dependent. For this reason, a message discriminator is attached to a syslog session, transport, or channel, with possible device support for multiple sessions, transports, or channels, each of which can be attached to a different discriminator.

The establishment of a message discriminator should be separate from the establishment of a syslog session. A message discriminator should refer to the syslog session, transport, or channel to which it should be attached. The reasons for the separation are the following:

- Message discriminators can be managed separately from the connections, and refinements in the capabilities available to set up message discriminators need not affect how syslog sessions are set up and vice versa.
- Multiple connections can be attached to the same message discriminator, allowing for various syslog redundancy topologies.

When an explicit message discriminator is not associated with a syslog session, the generic message discriminator from the device-wide global settings is used. You can create an “empty” message discriminator without specifying attribute values (no rate limit and no filter configured).

Rate Limiting

The device-wide rate limiting capability in Cisco IOS XE syslog is preserved in the Reliable Delivery and Filtering for Syslog feature and is referred to as “global rate limiting.” If you do not use global rate limiting, all event messages are sent to remote syslog hosts if system resources can support the volume. When global rate limiting is set, it applies to all destinations. The value is set to the rate-limit attribute of the “generic message discriminator” if one has been set. The disadvantage of global rate limiting is that the rate limit of the least performing remote syslog host sets the rate for how fast a device can send out syslog messages.

The Reliable Delivery and Filtering for Syslog feature provides syslog session-based rate limiting to bypass the effects of global rate limiting. This session-based rate limiting is associated with a specific message discriminator and allows you to set the rate acceptance level independently for each syslog session.

Use of global rate limiting is not recommended when session-based rate limiting is in effect. A rate limit in a message discriminator specifies a not-to-exceed rate of syslog messages but does not guarantee that this rate will be reached. A configured global rate limit may cause messages on a session to be dropped even if the rate limit for that session has not been reached. These actions are important to understand if global rate limiting and session-based rate limiting are used concurrently.

Benefits of Reliable Delivery and Filtering for Syslog

- Authentication and encryption capabilities in BEEP provide reliable and secure delivery for syslog messages
- Multiple sessions to a single logging host independent of the underlying transport method
- Session-based message filtering and rate limiting

- Multiple connections can be attached to the same message discriminator, allowing various syslog redundancy topologies
- New CLI command to disable the default syslog count
- New CLI command to help identify relative positions of syslog messages that are dropped due to rate limiting

How to Configure Reliable Delivery and Filtering for Syslog

Creating a Message Discriminator

Perform this task to create a message discriminator for syslog messages.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num*	**includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr1 facility includes facl357`	Creates a message discriminator with a facility subfilter. In this example, all messages with "facl357" in the facility field will be delivered.
Step 4	**end** **Example:**	Returns the CLI to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Associating a Message Discriminator with a Logging Buffer

Perform this task to associate a message discriminator with a specific buffer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging buffered** [**discriminator** *discr-name* | **xml**] [*buffer-size*] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num*	**includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr2`	Creates a message discriminator.
Step 4	**logging buffered** [**discriminator** *discr-name*	**xml**] [*buffer-size*] [*severity-level*] **Example:** `Device(config)# logging buffered discriminator pacfltr2 5`	Enables logging to a local buffer and specifies a message discriminator.
Step 5	**end** **Example:**	Returns the CLI to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Associating a Message Discriminator with a Console Terminal

Perform this task to associate a message discriminator with a console terminal.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging console** [**discriminator** *discr-name* | **xml**] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num*	**includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr3`	Creates a message discriminator.
Step 4	**logging console** [**discriminator** *discr-name*	**xml**] [*severity-level*] **Example:** `Device(config)# logging console discriminator pacfltr3 1`	Enables logging to the console and specifies a message discriminator filtering messages at a specific severity level.
Step 5	**end** **Example:**	Returns the CLI to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Associating a Message Discriminator with Terminal Lines

Perform this task to associate a message discriminator with terminal lines and have messages display at a monitor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging monitor** [**discriminator** *discr-name*| **xml**] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num*	**includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr4`	Creates a message discriminator.
Step 4	**logging monitor** [**discriminator** *discr-name*| **xml**] [*severity-level*] **Example:** `Device(config)# logging monitor discriminator pacfltr4 2`	Specifies a message discriminator named pacfltr4 and enables logging to the terminal lines of messages at severity level 2 and lower.
Step 5	**end** **Example:**	Returns the CLI to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Enabling Message Counters

Perform this task to enable logging of debug, log, or syslog messages.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging message-counter** {**debug** | **log** | **syslog**}
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging message-counter** {**debug**	**log**	**syslog**} **Example:** `Device(config)# logging message-counter syslog`	Enables logging of syslog messages.
Step 4	**end** **Example:** `Device(config)# end`	Returns the CLI to privileged EXEC mode.

Adding and Removing a BEEP Session

Perform this task to add and remove a BEEP session.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **logging host** {{*ip-address* | *hostname*} [**vrf** *vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator** *discr-name* | [[**filtered** [**stream** *stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel** *chnl-number*] [**sasl** *profile-name*] [**tls cipher** [*cipher-num*] **trustpoint** *trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port** *port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string** *custom-string*}]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging host** {{*ip-address* | *hostname*} [**vrf** *vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator** *discr-name* | [[**filtered** [**stream** *stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel** *chnl-number*] [**sasl** *profile-name*] [**tls cipher** [*cipher-num*] **trustpoint** *trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port** *port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string** *custom-string*}] **Example:** `Device(config)# logging host host3 transport beep port 600 channel 3`	Identifies a logging host and specifies the transport protocol, port, and channel for logging messages.
Step 4	**end** **Example:** `Device(config)# end`	Returns the CLI to privileged EXEC mode.

Configuration Examples for Reliable Delivery and Filtering for Syslog

Configuring Transport and Logging Example

```
Device(config)# do show running-config
 | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
```

```
Device(config)# logging host 209.165.201.1 transport beep port 600 channel 3
Device(config)# logging host 209.165.201.1 transport tcp port 602

Device(config)# show running-config | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
logging host 209.165.201.1 transport beep port 600 channel 3
logging host 209.165.201.1 transport tcp port 602
Device(config)#
```

Additional References for VRF-Aware Source Interfaces for Syslog Transactions

Related Documents

Related Topic	Document Title
Network Management commands (including logging commands): complete command syntax, defaults, command mode, command history, usage guidelines, and examples	*Cisco IOS Network Management Command Reference*
Syslog logging	*Troubleshooting and Fault Management module*

Standards and RFCs

Standard/RFC	Title
No new or modified standards/RFCs are supported by this feature, and support for existing standards/RFCs has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Reliable Delivery and Filtering for Syslog

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 80: Feature Information for Reliable Delivery and Filtering for Syslog

Feature Name	Releases	Feature Information
Reliable Delivery and Filtering for Syslog	Cisco IOS XE Release 2.1	The Reliable Delivery and Filtering for Syslog feature allows a device to be customized for receipt of syslog messages. This feature provides for reliable and secure delivery for syslog messages using BEEP. Additionally it allows multiple sessions to a single logging host, independent of the underlying transport method, and provides a filtering mechanism called a message discriminator. In Cisco IOS XE Release 2.1, this feature was introduced on Cisco ASR 1000 Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced or modified: **logging buffered**, **logging console**, **logging discriminator**, **logging host**, **logging message-counter**, **logging monitor**, **show logging**.