CONTENTS

CHAPTER 1

Programmability 1

boot ipxe 3
boot manual 4
boot system 5
clear configuration lock 6
clear netconf-yang session 7
controller (OpenFlow) 8
debug netconf-yang 10
debug restconf 11
default boot 13
dig 14
encoding 16
feature openflow 17
filter xpath 18
guestshell 19
guestshell portforwarding 20
install 22
iox 27
mlog 28
monitor log profile netconf-yang 29
monitor log profile restconf 32
netconf legacy 35
net-dhcp 36
net-debug 37
net-show 39
net-tcp-bufs 40
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>net-tcp-mss</td>
<td>41</td>
</tr>
<tr>
<td>net6-dhcp</td>
<td>42</td>
</tr>
<tr>
<td>net6-show</td>
<td>43</td>
</tr>
<tr>
<td>ping</td>
<td>44</td>
</tr>
<tr>
<td>ping4</td>
<td>45</td>
</tr>
<tr>
<td>ping6</td>
<td>46</td>
</tr>
<tr>
<td>receiver</td>
<td>47</td>
</tr>
<tr>
<td>show install</td>
<td>48</td>
</tr>
<tr>
<td>show iox-service</td>
<td>51</td>
</tr>
<tr>
<td>show log profile netconf-yang</td>
<td>54</td>
</tr>
<tr>
<td>show log profile restconf</td>
<td>57</td>
</tr>
<tr>
<td>show netconf-yang</td>
<td>60</td>
</tr>
<tr>
<td>show platform yang-management process</td>
<td>63</td>
</tr>
<tr>
<td>show openflow hardware capabilities</td>
<td>65</td>
</tr>
<tr>
<td>show openflow interface</td>
<td>68</td>
</tr>
<tr>
<td>show openflow switch flows</td>
<td>70</td>
</tr>
<tr>
<td>show telemetry ietf subscription</td>
<td>72</td>
</tr>
<tr>
<td>switch (OpenFlow)</td>
<td>75</td>
</tr>
<tr>
<td>stream</td>
<td>76</td>
</tr>
<tr>
<td>telemetry ietf subscription</td>
<td>77</td>
</tr>
<tr>
<td>update-policy</td>
<td>78</td>
</tr>
</tbody>
</table>
Programmability

• boot ipxe, on page 3
• boot manual, on page 4
• boot system, on page 5
• clear configuration lock, on page 6
• clear netconf-yang session, on page 7
• controller (OpenFlow), on page 8
• debug netconf-yang, on page 10
• debug restconf, on page 11
• default boot, on page 13
• dig, on page 14
• encoding, on page 16
• feature openflow, on page 17
• filter xpath, on page 18
• guestshell, on page 19
• guestshell portforwarding, on page 20
• install, on page 22
• iox, on page 27
• mlog, on page 28
• monitor log profile netconf-yang, on page 29
• monitor log profile restconf, on page 32
• netconf legacy, on page 35
• net-dhcp, on page 36
• net-debug, on page 37
• net-show, on page 39
• net-tcp-bufs, on page 40
• net-tcp-mss, on page 41
• net6-dhcp, on page 42
• net6-show, on page 43
• ping, on page 44
• ping4, on page 45
• ping6, on page 46
• receiver, on page 47
• show install, on page 48
• show iox-service, on page 51
• show log profile netconf-yang, on page 54
• show log profile restconf, on page 57
• show netconf-yang, on page 60
• show platform yang-management process, on page 63
• show openflow hardware capabilities, on page 65
• show openflow interface, on page 68
• show openflow switch flows, on page 70
• show telemetry ietf subscription, on page 72
• switch (OpenFlow), on page 75
• stream, on page 76
• telemetry ietf subscription, on page 77
• update-policy, on page 78
**boot ipxe**

To configure iPXE boot, use the `boot ipxe` command in global configuration mode. To disable the configuration, use the `no` form of this command.

```
boot ipxe {forever | timeout seconds} switch switch-number
no boot ipxe {forever | timeout seconds} switch switch-number
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forever</td>
<td>Attempts iPXE boot forever.</td>
</tr>
<tr>
<td>timeout seconds</td>
<td>Configures a timeout in seconds for iPXE network boot. Valid values are from 1 to 2147483647.</td>
</tr>
<tr>
<td>switch switch-number</td>
<td>Enables iPXE boot for switches in the stack. Valid values are from 0 to 9.</td>
</tr>
</tbody>
</table>

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Denali 16.3.2</td>
<td>This command was introduced on Cisco Catalyst 3650 and 3850 Series Switches.</td>
</tr>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was implemented on Cisco Catalyst 9300 and 9500 Series Switches</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

iPXE is an open source implementation of the Preboot eXecution Environment (PXE). Bootloaders boot an image located on a File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), or Trivial File Transfer Protocol (TFTP) server.

If the `forever` keyword is configured, the switch sends Dynamic Host Configuration Protocol (DHCP) requests forever. If the `timeout` keyword is configured, DHCP requests are sent for the specified amount of time, and when the timeout expires, the switch reverts to device boot.

**Example**

The following example shows how to configure an iPXE boot timeout for switch 2:

```
Device(config)# boot ipxe timeout 240 switch 2
```
boot manual

To configure manual boot, use the `boot manual` command in global configuration mode. To remove the configuration, use the `no` form of this command.

```
boot manual switch switch-number
no boot manual switch switch-number
```

**Syntax Description**

| switch switch-number | Configures manual boot for the switches in the stack. |

**Command Default**

Manual boot is enabled.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Denali 16.3.2</td>
<td>This command was introduced on Cisco Catalyst 3650 and 3850 Series Switches.</td>
</tr>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was implemented on Cisco Catalyst 9300 and 9500 Series Switches</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

When manual boot is disabled, and the switch reloads, the boot process starts automatically. When manual boot is disabled, the bootloader determines whether to execute a device boot or a network boot based on the configured value of the iPXE ROMMON variable.

**Example**

The following example shows how to configure manual boot for switch 2:

```
Device(config)# boot manual switch 2
```
**boot system**

To enable a system image boot, use the `boot system` command in global configuration mode. To disable the configuration, use the `no` form of this command.

```
boot system switch {all number} {flash: | ftp: | http: | tftp:}
no boot system [{switch |{all number}}] [{flash: | ftp: | http: | tftp:}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flash:</td>
<td>Specifies the flash filesystem to boot an image.</td>
</tr>
<tr>
<td>ftp:</td>
<td>Specifies a File Transfer Protocol (FTP) location to boot an image.</td>
</tr>
<tr>
<td>http:</td>
<td>Specifies a Hypertext Transfer Protocol (HTTP) location to boot an image.</td>
</tr>
<tr>
<td>tftp:</td>
<td>Specifies a Trivial File Transfer Protocol (TFTP) location to boot an image.</td>
</tr>
<tr>
<td>switch number</td>
<td>Enables booting for switches in a stack. Valid values are from 0 to 9.</td>
</tr>
</tbody>
</table>

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Denali 16.3.2</td>
<td>This command was introduced on Cisco Catalyst 3650 and 3850 Series Switches.</td>
</tr>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was implemented on Cisco Catalyst 9300 and 9500 Series Switches</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

You can either use an IPv4 or an IPv6 address for the remote FTP/HTTP/TFTP servers. When using an IPv6 address, you must enter the IPv6 address inside square brackets (as per RFC 2732); otherwise, the device will not boot.

**Note**

IPv6 is not supported on Catalyst 9000 Series Switches.

**Example**

The following example shows how to boot an image from an IPv4 HTTP server:

```
Device(config)# boot system switch 1 http://192.0.2.42/image-filename
```

The following example shows how to boot an image from an IPv6 HTTP server:

```
Device(config)# boot system switch 1 http://[2001:db8::1]/image-filename
```
clear configuration lock

To clear the configuration session lock, use the `clear configuration lock` in privileged EXEC mode.

`clear configuration lock`

This command has no arguments or keywords.

**Command Default**
Session lock times out after 10 minutes.

**Command Modes**
Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Release Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**
Use this command to remove the configuration lock on a session. A full synchronization of the database is triggered when a lock is cleared.

Read operation is allowed by any NETCONF/RESTCONF sessions during the global lock. However, write operation is only allowed by the NETCONF session that owns the lock.

**Example**

The following example shows how to clear a configuration lock:

```
Device# clear configuration lock
```
clear netconf-yang session

To clear NETCONF-YANG sessions, use the `clear netconf-yang session` command in privileged EXEC mode.

```
clear netconf-yang session session-id
[[R0 | R1 | RP {active | standby}]]
```

**Syntax Description**
- `session-id` (Optional) Clears the specified session. Valid values are from 1 to 4294967295.
- `R0` (Optional) Clears the Route Processor (RP) slot 0.
- `R1` (Optional) Clears the RP slot 1.
- `RP` (Optional) Clears the RP.
- `active` (Optional) Clears the active instance of the RP.
- `standby` (Optional) Clears the standby instance of the RP.

**Command Modes**
- Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

You can use this command to unlock a datastore by killing the locked session that has the ownership of the datastore lock. When a global lock is cleared by using the `clear netconf-yang session` command, a full synchronization of the datastore is triggered. However; clearing a session while the global lock is in place, only schedules a full synchronization.

**Examples**

The following example shows how to clear a NETCONF-YANG session:

```
Device# clear netconf-yang session 2 RP active
```
controller (OpenFlow)

To connect to an OpenFlow controller, use the `controller` command in OpenFlow switch configuration mode. To disconnect an OpenFlow controller, use the `no` form of this command.

```
controller ipv4 controller-address [{port [{port-number}]},] [security {none | tls}],] [vrf [{'vrf-name'}]],
no controller ipv4 controller-address [{port [{port-number}]},] [security {none | tls}],] [vrf [{'vrf-name'}]],
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipv4 controller-address</td>
<td>Configures the IP address of the OpenFlow controller.</td>
</tr>
<tr>
<td>port port-number</td>
<td>(Optional) Configures the OpenFlow controller TCP port. The default is 6653.</td>
</tr>
<tr>
<td>security</td>
<td>(Optional) Configures the OpenFlow controller connection security.</td>
</tr>
<tr>
<td>none</td>
<td>(Optional) Configures no authentication or encryption for the controller.</td>
</tr>
<tr>
<td>tls</td>
<td>(Optional) Configures the Transport Layer Security (TLS) protocol for the controller.</td>
</tr>
<tr>
<td>vrf vrf-name</td>
<td>(Optional) Configures a virtual routing and forwarding (VRF) instance for the OpenFlow controller.</td>
</tr>
</tbody>
</table>

**Command Default**

The controller is not configured.

**Command Modes**

- OpenFlow switch configuration (config-openflow-switch)

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The OpenFlow controller is an entity that interacts with the OpenFlow switch using the OpenFlow protocol. In most cases, an OpenFlow controller is a software that controls many OpenFlow logical switches. OpenFlow controllers offer a centralized view of the network, and enable administrators to dictate to the underlying systems (switches and routers) on how to handle the network traffic.

**Example**

The following example shows how to configure an OpenFlow controller:

```
Device# configure terminal
Device(config)# feature openflow
Device(config)# openflow
Device(config-openflow)# switch 1 pipeline 1
```
Device(config-openflow-switch)# controller ipv4
10.2.2.2 port 6633 vrf Mgmt-vrf security none

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>feature openflow</td>
<td>Enables the OpenFlow feature.</td>
</tr>
<tr>
<td>openflow</td>
<td>Enables OpenFlow configuration and enters OpenFlow configuration mode.</td>
</tr>
<tr>
<td>switch</td>
<td>Configures a logical switch and enters OpenFlow switch configuration mode.</td>
</tr>
</tbody>
</table>
debug netconf-yang

To log NETCONF-YANG debug messages, use the debug netconf-yang command in privileged EXEC mode.

```
debug netconf-yang [{level |debug | emergency | error | info | noise | notice | verbose | warning}]

no debug netconf-yang [{level |debug | emergency | error | info | noise | notice | verbose | warning}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>(Optional) Specifies the log level of NETCONF-YANG processes.</td>
</tr>
<tr>
<td>debug</td>
<td>(Optional) Logs debug messages.</td>
</tr>
<tr>
<td>emergency</td>
<td>(Optional) Logs emergency messages.</td>
</tr>
<tr>
<td>error</td>
<td>(Optional) Logs error messages.</td>
</tr>
<tr>
<td>info</td>
<td>(Optional) Logs information messages.</td>
</tr>
<tr>
<td>noise</td>
<td>(Optional) Specifies the maximum log level setting. This setting includes all logs in the output such as, emergency, alert, critical, error, warning, notice, debug, verbose and so on.</td>
</tr>
<tr>
<td>notice</td>
<td>(Optional) Logs notice messages.</td>
</tr>
<tr>
<td>verbose</td>
<td>(Optional) Logs debug messages in detail.</td>
</tr>
<tr>
<td>warning</td>
<td>(Optional) Logs warning messages.</td>
</tr>
</tbody>
</table>

**Command Default**

Debug logs are not enabled.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The last enabled debug logging level is used for logging debug messages. For example, if warning level is enabled by NETCONF-YANG, and it is followed by debug level by RESTCONF; then debug messages are logged.

The last enabled debug logging level will remain persistent for data model interface (DMI) processes.

**Examples**

The following is sample output from the debug netconf-yang level debug command:

```
Device# debug netconf-yang level debug
Jan 24 13:33:20.441 EST: yang-infra: netconf-yang server log level set to debug
```
debug restconf

To log RESTCONF debug messages, use the `debug restconf` command in privileged EXEC mode.

```
debug restconf [{level {debug |emergency | error | info | noise | notice | verbose | warning}}]

no debug restconf [{level {debug |emergency | error | info | noise | notice | verbose | warning}}]
```

### Syntax Description

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level</strong></td>
<td>(Optional) Specifies the log level of RESTCONF processes.</td>
</tr>
<tr>
<td><strong>debug</strong></td>
<td>(Optional) Logs debug messages.</td>
</tr>
<tr>
<td><strong>emergency</strong></td>
<td>(Optional) Logs emergency messages.</td>
</tr>
<tr>
<td><strong>error</strong></td>
<td>(Optional) Logs error messages.</td>
</tr>
<tr>
<td><strong>info</strong></td>
<td>(Optional) Logs information messages.</td>
</tr>
<tr>
<td><strong>noise</strong></td>
<td>(Optional) Specifies the maximum log level setting. This setting includes all logs in the output such as, emergency, alert, critical, error, warning, notice, debug, verbose and so on.</td>
</tr>
<tr>
<td><strong>notice</strong></td>
<td>(Optional) Logs notice messages.</td>
</tr>
<tr>
<td><strong>verbose</strong></td>
<td>(Optional) Logs debug messages in detail.</td>
</tr>
<tr>
<td><strong>warning</strong></td>
<td>(Optional) Logs warning messages.</td>
</tr>
</tbody>
</table>

### Command Default

Debug logs are not enabled.

### Command Modes

Privileged EXEC (#)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

The last enabled debug logging level will be used for logging debug messages. For example, if `warning` level is enabled by NETCONF-YANG, and it is followed by `debug` level by RESTCONF; then debug level messages will be logged.

The last enabled debug logging level will remain persistent for data model interface (DMI) processes.

### Examples

The following is sample output from the `debug restconf` command:

```
Device# debug restconf
Device# show debug
IOSXE Conditional Debug Configs:
Conditional Debug Global State: Stop
IOSXE Packet Tracing Configs:
```
license policy manager client:
  platform software policy_manager_error debugging is on

Packet Infra debugs:

<table>
<thead>
<tr>
<th>Ip Address</th>
<th>Port</th>
</tr>
</thead>
</table>

netconf-yang:
  netconf-yang debugging is on at level debug

restconf:
  restconf debugging is on at level debug
default boot

To modify the default boot system parameters, use the `default boot` command in global configuration mode.

```
default boot  {ipxe  {forever | timeout | seconds}  | manual  | system {flash: | ftp: | http: | tftp:}  } switch number
```

**Syntax Description**

- **ipxe**: Enables iPXE boot.
- **forever**: Attempts iPXE boot forever.
- **timeout seconds**: Configures a boot timeout in seconds. Valid values are from 1 to 2147483647.
- **manual**: Enables manual boot.
- **system**: Enables a system image boot.
- **flash:** Specifies the flash filesystem to boot an image.
- **ftp:** Specifies an File Transfer Protocol (FTP) location to boot an image.
- **http:** Specifies an Hypertext Transfer Protocol (HTTP) location to boot an image.
- **tftp:** Specifies a Trivial File Transfer Protocol (TFTP) location to boot an image.
- **switch number**: Enables booting for switches in a stack. Valid values are from 0 to 9.

**Command Default**

Device boot is enabled.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Denali 16.3.2</td>
<td>This command was introduced on Cisco Catalyst 3650 and 3850 Series Switches.</td>
</tr>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was implemented on Cisco Catalyst 9300 and 9500 Series Switches</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

You can either use the `no boot ipxe` or the `default boot ipxe` command to configure device boot.

If the `forever` keyword is configured, the switch sends Dynamic Host Configuration Protocol (DHCP) requests forever. If the `timeout` keyword is configured, DHCP requests are sent for the specified amount of time, and when the timeout expires, the switch reverts to device boot.

**Examples**

The following example shows how to enable the default boot mode:

```
Device(config)# default boot ipxe
```
To do a lookup of the Domain Name System (DNS) server, use the `dig` command in rommon mode.

```
dig hostname {v4 v6} [{dns-server-address}]
```

**Syntax Description**

- `hostname`: DNS host name.
- `v4`: IPv4 address.
- `v6`: IPv6 address.
- `dns-server-address`: (Optional) DNS Server IP address.

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command does a look up of the DNS name and displays the IP/IPv6 address of the DNS server.

**Example**

The following is sample output from the `dig hostname` command:

```
Device: dig example.org
DNS lookup using 2001:DB8::1
addr = 2001:DB8:0000:0000:0000:0000:0000:0001
```

The following is sample output from the `dig hostname v4` command:

```
Device: dig example.org v4
DNS lookup using 10.29.27.5
addr = 172.16.0.1
```

The following is sample output from the `dig hostname v4 dns-server-address` command:

```
Device: dig example.org v4 10.29.27.5
DNS lookup using 10.29.27.5
addr = 172.16.0.1
```

The following is sample output from the `dig hostname v6` command:

```
Device: dig example.org v6
DNS lookup using 2001:DB8::1
addr = 2001:DB8:0000:0000:0000:0000:0000:0001
```
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net-debug</td>
<td>Displays or changes the network debug values.</td>
</tr>
</tbody>
</table>
encoding

To configure telemetry encoding for the subscription, use the `encoding` command in telemetry-subscription configuration mode. To disable the configuration, use the `no` form of this command.

```
encoding encode-kvGPB
no encoding encode-kvGPB
```

**Syntax Description**
- `encode-kvGPB` Configures Key-value Google Protocol Buffers (kvGPB) encoding.

**Command Modes**
- Telemetry-subscription configuration (config)

**Command History**
- **Release** Cisco IOS XE Gibraltar 16.10.1
  - **Modification** This command was introduced.

**Example**

The following example shows how to configure a telemetry encoding for the subscription:

```
Device(config)# telemetry ietf subscription 101
Device(config-mdt-subs)# encoding encode-kvGPB
```
feature openflow

To enable the OpenFlow feature, use the `feature openflow` command in global configuration mode. To disable the OpenFlow feature, use the `no` form of this command.

```
feature openflow
no feature openflow
```

This command has no arguments or keywords.

### Command Default
OpenFlow is not configured.

### Command Modes
- Global configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines
Before configuring this command, you must configure the `boot mode openflow` command to enable OpenFlow forwarding mode on your device.

### Example
The following example shows how to enable the OpenFlow configuration:

```
Device# configure terminal
Device(config)# feature openflow
Device(config)#
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boot mode openflow</td>
<td>Enables OpenFlow forwarding mode.</td>
</tr>
</tbody>
</table>
filter xpath

To configure XPath filter, use the `filter xpath` command in telemetry-subscription configuration mode. To disable the configuration, use the `no` form of this command.

```
filter xpath path
nofilter xpath path
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Specifies XPath filter.</td>
</tr>
</tbody>
</table>

**Command Modes**

Telemetry-subscription configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Gibraltar 16.10.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The set of events from a stream are filtered. Different filter types are used for different stream types. Cisco IOS XE supports the yang-push stream.

The dataset within the yang-push stream to be subscribed to is specified by the use of an XPath filter.

**Example**

The following example shows how to configure XPath filter for subscription:

```
Device(config)# telemetry ietf subscription 101
Device(config-mdt-sub)# filter xpath /memory-ios-xe-oper:memory-statistics/memory-statistic
```
guestshell

To configure the Guest Shell infrastructure functionality, use the `guestshell` command in privileged EXEC mode.

```
guestshell  {destroy | disable | enable | run [{linux-executable}]
```

**Syntax Description**

- `destroy`  
  Deactivates and uninstalls the Guest Shell service.

- `disable`  
  Disables the Guest Shell service.

- `enable`  
  Disables the Guest Shell service.

- `run  [linux-executable]`  
  Executes or runs a Linux program in the Guest Shell.

**Command Default**

Guest Shell is not enabled.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Guest Shell is an embedded Linux environment that allows customers to develop and run custom Python applications for automated control and management of Cisco switches. Guest Shell is packaged as a Cisco application hosting framework (CAF)-formatted tar file (guest_shell.tar) into the Cisco IOS XE Everest 16.5.x release image read-only file system.

Configure the `iox` command in global configuration mode, before configuring this command. IOx is the Cisco-developed framework for hosting customer-deployed Linux applications on Cisco networking systems.

**Examples**

The following example shows how to enable and run the Guest Shell:

```
Device# configure terminal
Device(config)# iox
Device(config)# exit
Device# guestshell enable
Device# guestshell run
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iox</td>
<td>Configure IOx services.</td>
</tr>
</tbody>
</table>
guestshell portforwarding

To enable Guest Shell port forwarding, use the guestshell portforwarding command in privileged EXEC mode.

guestshell portforwarding {add table-entry entry-name service {tcp |udp }source-port port-number destination-port port-number |delete table-entry entry-name }

**Syntax Description**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>add</strong></td>
<td>Adds an IP table entry.</td>
</tr>
<tr>
<td><strong>table-entry entry-name</strong></td>
<td>Specifies the IP table name. The <em>table-name</em> argument must be unique, and it can be alphanumeric characters.</td>
</tr>
<tr>
<td><strong>service</strong></td>
<td>Specifies the service protocol.</td>
</tr>
<tr>
<td><strong>tcp</strong></td>
<td>Specifies TCP as the service protocol.</td>
</tr>
<tr>
<td><strong>udp</strong></td>
<td>Specifies UDP as the service protocol.</td>
</tr>
<tr>
<td><strong>source-port port-number</strong></td>
<td>Specifies the source port. Valid values for the <em>port-number</em> argument are from 1 to 65535.</td>
</tr>
<tr>
<td><strong>destination-port port-number</strong></td>
<td>Specifies the destination port. Valid values for the <em>port-number</em> argument are from 1 to 65535.</td>
</tr>
<tr>
<td><strong>delete</strong></td>
<td>Deletes an IP table entry.</td>
</tr>
</tbody>
</table>

**Command Default**

Port forwarding is not enabled.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Use this command to enable port forwarding for Guest Shell, when it connected through the GigabitEthernet 0/0 management interface

**Examples**

The following example shows how to enable port forwarding for Guest Shell:

```
Device# configure terminal
Device(config)# iox
```
Device(config)# exit
Device# guestshell portforwarding add table-entry table1 service tcp
   source-port 32 destination-port 9
Device#

The following example shows how to disable port forwarding for Guest Shell:

Device# guestshell portforwarding delete table-entry table1
Device#

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>guestshell</td>
<td>Configures the Guest Shell infrastructure functionality.</td>
</tr>
</tbody>
</table>
To install data model update packages, use the `install` command in privileged EXEC mode.

```
```

### Syntax Description

**activate**

Validates whether the model update package is added through the `install add` command, and restarts NETCONF processes (confd and opdatamgrd).

This keyword runs a compatibility check, updates package status, and if the package can be restarted, it triggers post-install scripts to restart the necessary processes, or triggers a reload for non-restartable packages.

**file**

Specifies the package to be activated.


(Optional) Prompts the user about installation activities.

For example, the `activate` keyword, automatically triggers a reload for packages that require a reload. Before activating the package, a message will prompt users as to whether they want to continue.

The `all` keyword allows you to enable prompts. The `none` keyword disables prompts.

**prompt-level {all | none}**

Specifies the location of the installed package.

**add**

Copies files from a remote location (via FTP, TFTP) to a device, and performs a compatibility check for the platform and image versions.

This keyword runs base compatibility checks to ensure that a specified package is supported on a platform. It also adds an entry in the package file, so that the status can be monitored and maintained.

**{http: | https: | rcp: | scp: | tftp:}**

Specifies the package to be added.
### commit
Makes changes persistent over reloads.
You can do a commit after activating a package, while the system is up, or after the first reload. If a package is activated, but not committed, it remains active after the first reload, but not after the second reload.

### deactivate
Deactivates an installed package.
Deactivating a package also updates the package status and triggers a process restart or a reload.

### remove
Remove installed packages.
The package file is removed from the file system. The `remove` keyword can only be used on packages that are currently inactive.

### inactive
Removes all inactive packages from the device.

### rollback
Rolls back the data model update package to the base version, the last committed version, or a known commit ID, and restarts NECONF processes.

### to base
Returns to the base image.

### committed
Returns to the installation state when the last commit operation was performed.

### id install-ID
Returns to the specific install point ID. Valid values are from 1 to 4294967295.

---

**Command Default**
Model update packages are not installed.

**Command Modes**
Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced on the following platforms:</td>
</tr>
<tr>
<td></td>
<td>• Cisco 4000 Series Integrated Services Routers</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 9300 Series Switches</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 9500 Series Switches</td>
</tr>
<tr>
<td></td>
<td>• Cisco Cloud Services Router 1000v</td>
</tr>
<tr>
<td></td>
<td>• Cisco Integrated Services Virtual Routers (ISRv)</td>
</tr>
</tbody>
</table>
Modification

This command was implemented on the following platforms:

- Cisco Catalyst 3650 Series Switches
- Cisco Catalyst 3850 Series Switches

Usage Guidelines

In Service Model Update adds new data models or extend functionality to existing data models. The update package provides YANG model enhancements outside of a release cycle. The update package is a superset of all existing models; it includes all existing models as well as updated YANG models.

A model update package must be added prior to activating the update package. A package must be deactivated, before it is removed from the bootflash.

Cisco 4000 Series Integrated Services Routers

The following example shows how to add an install package on a device:

```
Device# install add file tftp://172.16.0.1//tftpboot/folder1/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
install_add: START Sun Feb 26 05:57:04 UTC 2017
Downloading file tftp://172.16.0.1//tftpboot/folder1/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
Finished downloading file tftp://172.16.0.1//tftpboot/folder1/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
SUCCESS: install_add /bootflash/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
Sun Feb 26 05:57:22 UTC 2017
```

The following example shows how to activate an install package:

```
Device# install activate file bootflash:/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
install_activate: START Sun Feb 26 05:58:41 UTC 2017
DMP package.
Netconf processes stopped
SUCCESS: install_activate /bootflash/isr4300-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.dmp.bin
Sun Feb 26 05:58:58 UTC 2017
```

The following example shows how to commit an installed package:

```
Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
```

The running configuration will be synchronized to the NETCONF running data store.

The running configuration has been synchronized to the NETCONF running data store.
Device# install commit

install_commit: START Sun Feb 26 06:46:48 UTC 2017
SUCCESS: install_commit Sun Feb 26 06:46:52 UTC 2017

The following example shows how to rollback to the base package:

Device# install rollback to base

install_rollback: START Sun Feb 26 06:50:29 UTC 2017
!7 install_rollback: Restarting impacted processes to take effect
!7 install_rollback: restarting confd

*Feb 26 06:50:34.957: %DMI-4-CONTROL_SOCKET_CLOSED: SIP0: syncfd: Confd control socket closed Lost connection to ConfD (45): EOF on socket to ConfD.
*Feb 26 06:50:34.962: %DMI-4-CONTROL_SOCKET_CLOSED: SIP0: nesd: Confd control socket closed Lost connection to ConfD (45): EOF on socket to ConfD.
*Feb 26 06:50:34.963: %DMI-4-SUB_READ_FAIL: SIP0: vtyserverutild: Confd subscription socket read failed Lost connection to ConfD (45): EOF on socket to ConfD.Netconf processes stopped

7 install_rollback: DMP activate complete
SUCCESS: install_rollback Sun Feb 26 06:50:41 UTC 2017

*Feb 26 06:51:28.901: %DMI-5-SYNC_START: SIP0: syncfd: External change to running configuration detected.
The running configuration will be synchronized to the NETCONF running data store.
*Feb 26 06:51:30.339: %DMI-5-SYNC_COMPLETE: SIP0: syncfd: The running configuration has been synchronized to the NETCONF running data store.

Cisco Catalyst 3000 Series Switches

The following example shows how to add an install package on a device:

Device# install add file tftp://172.16.0.1//tftpboot/folder1/cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin

install_add: START Sat Jul 29 05:57:04 UTC 2017
Downloading file tftp://172.16.0.1//tftpboot/folder1/cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin
Finished downloading file tftp://172.16.0.1//tftpboot/folder1/cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.Sdmp.bin to bootflash:cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin
SUCCESS: install_add /bootflash/
cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin
Sat Jul 29 05:57:22 UTC 2017

The following sample output from the show install summary command displays that the update package is now committed, and that it will be persistent across reloads:

Device# show install summary

Active Packages:
bootflash:cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin
Inactive Packages:
No packages
Committed Packages:
bootflash:cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin
Uncommitted Packages:
No packages
Device#
### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>show install</td>
<td>Displays information about model update packages.</td>
</tr>
</tbody>
</table>
**iox**

To configure IOx services, use the `iox` command in global configuration mode. To remove the configuration, use the `no` form of this command.

```plaintext
iox
no iox
```

This command has no arguments or keywords.

**Command Default**

IOx services are not configured.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

IOx is the Cisco-developed framework for hosting customer-deployed Linux applications on Cisco networking systems. IOx facilitates the life-cycle management of app and data exchange by providing a set of services that helps developers to package pre-built apps, and host them on a target device. IOx life-cycle management includes distribution, deployment, hosting, starting, stopping (management), and monitoring of apps and data. IOx services also include app distribution and management tools that help users discover and deploy apps to the IOx framework.

**Examples**

The following example shows how to configure IOx services:

```plaintext
Device# configure terminal
Device(config)# iox
Device(config)# exit
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guestshell</td>
<td>Configures Guest Shell infrastructure functionality.</td>
</tr>
</tbody>
</table>
To direct log messages to a memory buffer instead of the serial port, use the `mlog` command in rommon mode.

```
mlog  [{show |reset |ctrl [{on |off |toggle}]}]
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>show</code></td>
<td>(Optional) Displays memory log messages.</td>
</tr>
<tr>
<td><code>reset</code></td>
<td>(Optional) Resets the logging of messages to the memory log.</td>
</tr>
<tr>
<td><code>ctrl</code></td>
<td>(Optional)</td>
</tr>
<tr>
<td><code>on</code></td>
<td>(Optional)</td>
</tr>
<tr>
<td><code>off</code></td>
<td>(Optional)</td>
</tr>
<tr>
<td><code>toggle</code></td>
<td>(Optional)</td>
</tr>
</tbody>
</table>

### Command Modes

Rommon

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

This command directs protocol log (that is all logs controlled by the `net-debug` command) messages to a memory buffer instead of the serial port.

With memory logging, log messages are displayed after a test is run. For example, HTTP debugs can be enabled through memory logging. Log messages are displayed in the memory buffer after running a copy from `http://server/name` to `null:command`.

### Example

The following example shows how to direct log messages to the memory buffer:

Device: `mlog show`

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>net-debug</code></td>
<td>Displays or changes the network debug values.</td>
</tr>
</tbody>
</table>
monitor log profile netconf-yang

To display debug logs for NETCONF-YANG processes, use the `monitor log profile netconf-yang` command in privileged EXEC mode.

**monitor log profile netconf-yang internal**

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th><code>internal</code></th>
<th>Displays all debug logs.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note</strong></td>
<td></td>
<td>This keyword is mainly used by customer support.</td>
</tr>
</tbody>
</table>

**Command Modes**

Privileged EXEC (`#`)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Logs generated by this command are rendered on the device console.

**Example**

The following example shows how to enable the `monitor log profile netconf-yang internal` command:

```
Device# monitor log profile netconf-yang internal
```

```
2018/01/24 15:58:50.356 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): gdb port 9919 allocated
2018/01/24 15:58:50.365 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): swift_repl port 8019 allocated
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): process scoreboard /tmp/rp/process/pttcd%rp_0_0%0.pid is 12040
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): pttcd%rp_0_0%0.gdbport is 9919
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): pttcd%rp_0_0%0.swift_replport is 8019
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Launching pttcd on fru rp slot 0 bay 0 instance 0 log /tmp/rp/trace/pttcd_pmanlog
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Hold failures 2, hold interval 1800
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/sbin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/.
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/.
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/.
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/.
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/.
```

Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): LD_LIBRARY_PATH is
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PREPROC_OPTIONS ==
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used: pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): full_path is /tmp/sw/rp/0/0
/rp_daemons/mount/usr/binos/bin/pttcd
2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Resolved readlink process

2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PREPROC_OPTIONS ==
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used: pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): full_path is /tmp/sw/rp/0/0
/rp_daemons/mount/usr/binos/bin/pttcd
2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Resolved readlink process

2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PREPROC_OPTIONS ==
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used: pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): full_path is /tmp/sw/rp/0/0
/rp_daemons/mount/usr/binos/bin/pttcd
2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Resolved readlink process

2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PREPROC_OPTIONS ==
2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used: pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): full_path is /tmp/sw/rp/0/0
/rp_daemons/mount/usr/binos/bin/pttcd
2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Resolved readlink process
2018/01/24 15:58:52.167 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): command line used pubd >>
/tmp/trace/pubd_pmanlog_cmd 2>&1 &
2018/01/24 15:58:52.170 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): full_path is /tmp/sw/rp/0/0
/rpdataemons/mount/usr/binos/bin/pubd
2018/01/24 15:58:52.172 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Resolved readlink process

/tmp/sw/mount/1000px86-rpcontrol.BLD_V168_THROTTLE_LATEST_20180122_164958_V16.8.0_177.SSA.pkg/usr/binos/bin/pubd
2018/01/24 15:58:52.172 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Full path used to spawn the process:
/tmp/sw/rp/0/0/rpdataemons/mount/usr/binos/bin/pubd
2018/01/24 15:58:52.177 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Binary_arch set to: [x86_64_cge7]
2018/01/24 15:58:52.184 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): actual pubd pid is 14920
2018/01/24 15:58:52.184 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Checking for cgroup for PID 14920
2018/01/24 15:58:52.184 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Setting cgroup iosxe_control_processes
/iosxe_mgmt_processes for PID 14920 and PID 14416
2018/01/24 15:58:52.188 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): /tmp/rp/pvp/process_state/pubd%rp_0_0%0#14416_state marked up
2018/01/24 15:58:52.193 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): oom_score_adj value is 399
2018/01/24 15:58:52.194 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Wait for signal or process exit: 14920
2018/01/24 15:58:52.540 {pttcd_R0-0}{1}: [pttcd] [12542]: (ERR): PPTCD_1_abcdefghi transaction id = 1
2018/01/24 15:58:57.296 {syncfd_pmanlog_R0-0}{1}: [syncfd_pmanlog] [19542]: (note): process scoreboard /tmp/rp/process/syncfd%rp_0_0 syncfd%rp_0_0.pid is 19470
monitor log profile restconf

To display debug logs for RESTCONF processes, use the **monitor log profile restconf** command in privileged EXEC mode.

```
monitor log profile netconf-yang internal
```

**Syntax Description**

| **internal** | Displays all debug logs. |
| **Note** | This keyword is used by customer support. |

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th><strong>Release</strong></th>
<th><strong>Modification</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Logs generated by this command are rendered on the device console.

**Example**

The following example shows how to enable the **monitor log profile restconf internal** command:

```
Device# monitor log profile restconf internal
Displaying traces starting from 2018/03/23 09:10:02.000. If no traces are present, the command will wait until one is.
```

```
2018/03/23 13:05:13.945 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): gdb port 9908 allocated
2018/03/23 13:05:13.962 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): swift_repl port 8008 allocated
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): process scoreboard /tmp/rp/process/pttcd%rp_0_0%0 pttcd%rp_0_0%0.pid is 2550
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): pttcd%rp_0_0%0.gdbport is 9908
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): pttcd%rp_0_0%0.swift_replport is 8008
2018/03/23 13:05:14.060 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Launching pttcd on fru rp slot 0 bay 0 instance 0 log /tmp/rp/trace/pttcd_pmanlog
2018/03/23 13:05:14.060 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Hold failures 2, hold interval 1800
2018/03/23 13:05:14.060 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:
```

```
/usr/bin:/sbin/usr/binos/conf:/usr/binos/bin:/bin/usr/bin:/usr/sbin:/usr/binos/conf:
```

Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
2018/03/23 13:05:14.060 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note):
LD_LIBRARY_PATH is
2018/03/23 13:05:14.063 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note):
PREPROC_OPTIONS --
2018/03/23 13:05:14.063 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): command line used pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/03/23 13:05:14.068 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note):
full_path is /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd
2018/03/23 13:05:14.069 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note):
Resolved readlink process /tmp/sw/mount/asr1000rpx86-rpcontrol.2018-03-07_18.30_rifu.SSA.pkg
/usr/binos/bin/pttcd
2018/03/23 13:05:14.069 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Full path used to spawn the process:
/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd
2018/03/23 13:05:14.076 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Binary_arch set to: [x86_64_cge7]
2018/03/23 13:05:14.088 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): actual pttcd pid is 2936
2018/03/23 13:05:14.088 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Checking for cgroup for PID 2936
2018/03/23 13:05:14.088 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): oom score adj value is 399
2018/03/23 13:05:14.102 {pttcd_R0-0}{1}: [pttcd] [2936]: (ERR): init_callhome() failed
2018/03/23 13:05:14.102 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Wait for signal or process exit: 2936
2018/03/23 13:05:16.895 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): gdb port 9920 allocated
2018/03/23 13:05:16.904 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): swift_repl port 8020 allocated
2018/03/23 13:05:16.987 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): process scoreboard
/tmp/rp/process/pubd%rp_0_0%0.pubd%rp_0_0%0.pid is 4922
2018/03/23 13:05:16.987 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): pubd%rp_0_0%0.gdbport is 9920
2018/03/23 13:05:16.987 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): pubd%rp_0_0%0.swift_replport is 8020
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PROCESS scoreboard exited
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/tmp/sw/rp/0/0/rp_daemons/mount/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/conf:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/binos/conf:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/binos/conf:/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/binos/conf:/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/sbin:/usr/binos/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
/usr/sbin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is
LD_LIBRARY_PATH is
2018/03/23 13:05:17.001 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note):
PREPROC_OPTIONS --
2018/03/23 13:05:17.001 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): command line used pubd >>
/tmp/rp/trace/pubd_pmanlog_cmd 2>&1 &
Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
2018/03/23 13:05:17.007 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Full path is /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pubd
2018/03/23 13:05:17.009 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Resolved readlink process:
    /tmp/sw/mount/asr1000rpx86-rpcontrol.2018-03-07_18.30_rifu.SSA.pkg/usr/binos/bin/pubd
2018/03/23 13:05:17.009 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Full path used to spawn the process:
    /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pubd
2018/03/23 13:05:17.017 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Binary_arch set to: [x86_64_cge7]
2018/03/23 13:05:17.031 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): actual pubd pid is 5303
2018/03/23 13:05:17.031 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Checking for cgroup for PID 5303
2018/03/23 13:05:17.031 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Setting cgroup iosxe_control_processes/iosxe_mgmt_processes for PID 5303 and PID 4922
2018/03/23 13:05:17.045 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note):
    /tmp/rp/pvp/process_state/pubd%rp_0_0%0#4922_state marked up
2018/03/23 13:05:17.047 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): oom score adj value is 399
**netconf legacy**

To enable legacy NETCONF protocol, use the `netconf legacy` command in global configuration mode. To disable the legacy NETCONF protocol, use the `no` form of this command.

```
netconf legacy
no netconf legacy
```

This command has no arguments or keywords.

**Command Default**

Legacy NETCONF protocol is not enabled.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Denali 16.3.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

If this command is enabled, the RFC-compliant NETCONF client (ncclient) does not work. This command enables the legacy NETCONF protocol that is non-RFC-compliant.

**Example**

The following example shows how to disable the legacy NETCONF protocol:

```
Device> enable
Device# configure terminal
Device(config)# no netconf legacy
```
**net-dhcp**

To initiate an IPv4 Dynamic Host Control Protocol (DHCP) request for remote configuration, use the `net-dhcp` command in rommon mode.

```
net-dhcp [{timeout}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>(Optional) Timeout in seconds.</td>
</tr>
</tbody>
</table>

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command initiates an IPv4 DHCP request and processes the reply.

**Example**

The following example shows how to enable the `net-dhcp` command:

```
Device: net-dhcp
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net-debug</td>
<td>Displays or changes the network debug values.</td>
</tr>
<tr>
<td>net-show</td>
<td>Displays network parameters.</td>
</tr>
<tr>
<td>net6-dhcp</td>
<td>Initiates an IPv6 DHCP request for remote configuration.</td>
</tr>
</tbody>
</table>
net-debug

To display or change the network debug values use the net-debug command in rommon mode.

```
net-debug [{new-value}]  
```

**Syntax Description**

| new-value | (Optional) New debug value to use. |

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command enables or disables log levels for each of the following functional areas:

- Domain Name System (DNS)
- Dynamic Host Control Protocol (DHCP)
- File Transfer Protocol (FTP)
- Hypertext Transfer Protocol (HTTP)
- IP
- TCP
- UDP
- Uniform Resource Identifier (URI)

**Example**

This following is sample output from the net-debug command:

```
Device: net-debug

ether: 0
ip: 0
dhcp: 0
udp: 0	cp: 0
http: 0
dns: 0
uri: 0
t/ftp: 2
ip6: 0
dhcp6: 0:000 200 000 000
```
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlog</td>
<td>Directs log messages to a memory buffer instead of the serial port.</td>
</tr>
</tbody>
</table>
**net-show**

To display network parameters, use the `net-show` command in rommon mode.

**net-show**

This command has no arguments or keywords.

**Command Modes**

<table>
<thead>
<tr>
<th>Command Modes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rommon</td>
<td></td>
</tr>
</tbody>
</table>

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command displays network configuration such as IP address, gateway, MAC address and so on.

**Example**

The following is sample output from the `net-show` command:

```
Device: net-show
Network params:
IPv4:
    ip addr 10.29.27.150
    netmask 255.255.0.0
    gateway 10.29.0.1
IPv6:
    link-local addr fe80::366f:90ff:feb8:cb80
    site-local addr fec0::366f:90ff:feb8:cb80
    DHCP addr 2001:dead:beef:cafe::9999
    router addr fe80::7ada:6eff:fe13:8580
    SLAAC addr f00d::366f:90ff:feb8:cb80 /64
    SLAAC addr feed::366f:90ff:feb8:cb80 /64
Common:
    macaddr 34:6f:90:b8:cb:80
    dns 2001:dead:beef:cafe::5
    bootfile http://www.example.org/ed10m
    domain ip6.example.org
```

**Command**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net6-show</td>
<td>Displays IPv6 network parameters.</td>
</tr>
</tbody>
</table>
To display TCP buffers, use the **net-tcp-bufs** command in rommon mode.

```
net-tcp-bufs [{mss}]
```

### Syntax Description

**mss**

(Optional) The Maximum Segment Size (MSS) of TCP buffers.

### Command Modes

Rommon

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

You can set the MSS of TCP buffers using the *mss* argument.

### Example

The following is sample output from the **net-tcp-bufs** command:

```
Device: net tcp-bufs
tcp_num_buffs 4
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net-tcp-mss</td>
<td>View or set the TCP MSS.</td>
</tr>
</tbody>
</table>
**net-tcp-mss**

To view or set the TCP Maximum Segment Size (MSS), use the `net-tcp-mss` command in rommon mode.

```
net-tcp-mss [{mss}]
```

**Syntax Description**

- **mss**
  
  (Optional) The Maximum Segment Size (MSS) of TCP buffers.

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Use the `mss` argument to change the MSS size.

**Example**

The following is sample output from the `net-tcp-mss` command:

```
Device: net-tcp-mss
switch: net-tcp-mss
tcp_segment_size 1024
```

The following is sample output from the `net-tcp-mss mss` command:

```
Device: net-tcp-mss 700
switch: net-tcp-mss 700
tcp_segment_size 700
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net-tcp-bufs</td>
<td>Displays TCP buffers.</td>
</tr>
</tbody>
</table>
**net6-dhcp**

To initiate an IPv6 Dynamic Host Control Protocol (DHCP) request for remote configuration, use the `net6-dhcp` command in rommon mode.

```
net6-dhcp [timeout]
```

**Syntax Description**

- **timeout**
  (Optional) Timeout in seconds.

**Command Modes**

- Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

You can change the timeout by specifying a time in seconds.

**Example**

The following example shows how to enable the `net6-dhcp` command:

```
Device: net6-dhcp
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>net-debug</code></td>
<td>Displays or changes the network debug values.</td>
</tr>
<tr>
<td><code>net-dhcp</code></td>
<td>Initiates an IPv4 DHCP request and processes the reply.</td>
</tr>
<tr>
<td><code>net-show</code></td>
<td>Displays network parameters.</td>
</tr>
</tbody>
</table>
To display IPv6 network parameters, use the `net6-show` command in rommon mode.

**net6-show**

This command has no arguments or keywords.

**Command Modes**

- Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

**Example**

The following is sample output from the `net6-show` command:

```
Device: net6-show

switch: net6-show
IP6 addresses
  link-local addr fe80::366f:90ff:feb8:cb80
  site-local addr fec0::366f:90ff:feb8:cb80
  DHCP addr 2001:dead:beef:cafe::9999
  router addr fe80::7ada:6eff:fe13:8580
  SLAAC addr f00d::366f:90ff:feb8:cb80 /64
  SLAAC addr feed::366f:90ff:feb8:cb80 /64
  null addr ::
  all-nodes addr ff02::1
  all-routers addr ff02::2
     all-dhcp addr ff02::1:2
  Slct-node addr ff02::1::ff8:cb80
  ll mmac addr 33:33:00:00:00:01
  sl mmac addr 33:33:00:00:00:02
  sn mmac addr 33:33:ff:b8:cb:80
  dhcp mmac addr 33:33:ff:00:99:99

IP6 neighbour table
  0: ip6 fec0::366f:90ff:feb8:cb80 MAC 34:6f:90:b8:cb:80
  4: ip6 fe80::32f7:dff:fe08:7e:bd MAC 30:f7:0d:08:7e:bd
```
To diagnose basic network connectivity, use the **ping** command in rommon mode.

```
ping [{host_ip_address}] [{retries}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>host_ip_address</strong></td>
<td>(Optional) IP address of the host.</td>
</tr>
<tr>
<td><strong>retries</strong></td>
<td>(Optional) Number of retries.</td>
</tr>
</tbody>
</table>

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The **ping** and **ping4** commands are the same.

The **ping** command is a very common method for troubleshooting the accessibility of devices. A timeout is implemented at the bootloader device prompt, that allows the bootloader to poll the TCP stack every 200 ms. As a result, the bootloader may take up to 200 ms to respond to pings. However, when the bootloader is downloading a file, and thus actively polling for new packets, it responds to ping quickly.

**Example**

The following is sample output from the **ping** command:

```
Device: ping 10.29.27.5
Ping 10.29.27.5 with 32 bytes of data ... Host 10.29.27.5 is alive.
```

The following is sample output from the **ping** **host_ip_address retries** command:

```
Device: ping 10 6.29.27.5 6
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 1 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
Ping 10.29.27.5 with 32 bytes of data ... reply received in 0 ms
```

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ping4</strong></td>
<td>Diagnoses basic network connectivity.</td>
<td></td>
</tr>
<tr>
<td><strong>ping6</strong></td>
<td>Determines the network connectivity to another device using IPv6 addressing.</td>
<td></td>
</tr>
</tbody>
</table>
ping4

To diagnose basic network connectivity, use the ping4 command in rommon mode.

```
ping4 [{host_ip_address}][{retries}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host_ip_address</td>
<td>(Optional) IP address of the host to be pinged.</td>
</tr>
<tr>
<td>retries</td>
<td>(Optional) Number of retries.</td>
</tr>
</tbody>
</table>

**Command Modes**

Rommon

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The ping and ping4 commands are the same.

A timeout is implemented at the bootloader device prompt, that allows the bootloader to poll the TCP stack every 200 ms. As a result, the bootloader may take up to 200 ms to respond to pings. However, when the bootloader is downloading a file, and thus actively polling for new packets, it responds to ping quickly.

**Example**

The following is sample output from the ping4 host_ip_address command:

```
Device: ping4 10.29.27.5

Ping 10.29.27.5 with 32 bytes of data ...
Host 10.29.27.5 is alive.
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ping</td>
<td>Diagnoses basic network connectivity.</td>
</tr>
<tr>
<td>ping6</td>
<td>Determines the network connectivity to another device using IPv6 addressing.</td>
</tr>
</tbody>
</table>
To determine the network connectivity to another device using IPv6 addressing, use the **ping6** command, rommon mode.

```
ping6 [{host}] [{repeats}] [{len}]
```

### Syntax Description

- **host** (Optional) IP address of the host to be pinged.
- **repeats** (Optional) Number of times to repeat the ping.
- **len**

### Command Modes

Rommon

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

A timeout is implemented at the bootloader device prompt, that allows the bootloader to poll the TCP stack every 200 ms. As a result, the bootloader may take up to 200 ms to respond to pings. However, when the bootloader is downloading a file, and thus actively polling for new packets, it responds to ping quickly.

### Example

The following is sample output from the `ping6 host retries len` command:

```
Device: ping6 2001:dead:beef:cafe::5 6 1000

Ping host 2001:dead:beef:cafe::5, 6 times, 1000 bytes
Ping 2001:dead:beef:cafe::5 ... reply in 0 ms
Ping 2001:dead:beef:cafe::5 ... reply in 1 ms
Ping 2001:dead:beef:cafe::5 ... reply in 1 ms
Ping 2001:dead:beef:cafe::5 ... reply in 0 ms
Ping 2001:dead:beef:cafe::5 ... reply in 0 ms
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ping</td>
<td>Diagnoses basic network connectivity.</td>
</tr>
<tr>
<td>ping4</td>
<td>Diagnoses basic network connectivity.</td>
</tr>
</tbody>
</table>
receiver

To configure receiver to receive update notifications, use the **receiver** command in telemetry-subscription configuration mode. To disable the configuration, use the **no** form of this command.

```
receiver ip address {ipv4-address | ipv6-address} port protocol protocol
no receiver ip address {ipv4-address | ipv6-address} port protocol protocol
```

### Syntax Description

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip address</td>
<td>Configures receiver IP address.</td>
</tr>
<tr>
<td>ipv4-address</td>
<td>IPv4 receiver address.</td>
</tr>
<tr>
<td>ipv6-address</td>
<td>IPv6 receiver address.</td>
</tr>
<tr>
<td>port</td>
<td>Configures receiver port.</td>
</tr>
<tr>
<td>protocol</td>
<td>Configures protocol for notification. The following protocol is supported:</td>
</tr>
<tr>
<td></td>
<td>• grpc-tcp</td>
</tr>
</tbody>
</table>

### Command Modes

Telemetry-subscription configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Gibraltar 16.10.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

A receiver is a network element that receives the telemetry data. Configured subscriptions are created by management operations on the publisher by controllers, and explicitly include the specification of the receiver of the telemetry data defined by the subscription. These subscriptions persist across reboots of the publisher. Configured subscriptions can be configured with multiple receivers, however; only the first valid receiver is used. Other receivers are not be tried, if a receiver is already connected, or in the process of being connected. If that receiver is deleted, another receiver is connected.

### Example

The following example shows how to configure receiver information for receiving notifications:

```
Device(config)# telemetry ietf subscription 101
Device(config-mdt-subs)# receiver ip address 10.28.35.45 57555 protocol grpc-tcp
```
show install

To display information about data model update packages, use the `show install` command in privileged EXEC mode.

```
show install {active |committed |inactive |log |package {bootflash: |flash: |webui:}|rollback |summary |uncommitted}
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>active</code></td>
<td>Displays information about active packages.</td>
</tr>
<tr>
<td><code>committed</code></td>
<td>Displays package activations that are persistent.</td>
</tr>
<tr>
<td><code>inactive</code></td>
<td>Displays inactive packages.</td>
</tr>
<tr>
<td><code>log</code></td>
<td>Displays entries stored in the logging installation buffer.</td>
</tr>
<tr>
<td><code>package</code></td>
<td>Displays metadata information about the package, including description, restart information, components in the package, and so on.</td>
</tr>
<tr>
<td>`{bootflash:</td>
<td>flash:</td>
</tr>
<tr>
<td><code>rollback</code></td>
<td>Displays the software set associated with a saved installation.</td>
</tr>
<tr>
<td><code>summary</code></td>
<td>Displays information about the list of active, inactive, committed, and superseded packages.</td>
</tr>
<tr>
<td><code>uncommitted</code></td>
<td>Displays package activations that are non persistent.</td>
</tr>
</tbody>
</table>

### Command Modes

Privileged EXEC (`#`)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced on the following platforms:</td>
</tr>
<tr>
<td></td>
<td>• Cisco 4000 Series Integrated Services Routers</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 9300 Series Switches</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 9500 Series Switches</td>
</tr>
<tr>
<td></td>
<td>• Cisco Cloud Services Router 1000v</td>
</tr>
<tr>
<td></td>
<td>• Cisco Integrated Services Virtual Routers (ISRv)</td>
</tr>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was implemented on the following platforms:</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 3650 Series Switches</td>
</tr>
<tr>
<td></td>
<td>• Cisco Catalyst 3850 Series Switches</td>
</tr>
</tbody>
</table>
Usage Guidelines

Use the show commands to view the status of an installed model update package.

Cisco 4000 Series Integrated Services Routers

The following is sample output from the **show install package** command:

Device# show install package bootflash:
isr4300-universalk9.16.05.01.CSCxxxxxxx.dmp.bin

Name: isr4300-universalk9.16.05.01.CSCxxxxxxx.dmp.bin
Version: 16.5.1.0.199.1484082952..Everest
Platform: ISR4300
Package Type: dmp
Defect ID: CSCxxxxxxx
Package State: Added
Supersedes List: {}
Smu ID: 1
Device#

The following is sample output from the **show install summary** command:

Device# show install summary

Active Packages:
bootflash:isr4300-universalk9.16.05.01.CSCxxxxxxx.dmp.bin
Inactive Packages:
No packages
Committed Packages:
No packages
Uncommitted Packages:
bootflash:isr4300-universalk9.16.05.01.CSCxxxxxxx.dmp.bin
Device#

The following is sample output from the **show install log** command:

Device# show install log

[0|install_op_boot]: START Fri Feb 24 19:20:19 Universal 2017
[0|install_op_boot]: END SUCCESS Fri Feb 24 19:20:23 Universal 2017
[3|install_add]: START Sun Feb 26 05:55:31 UTC 2017
[3|install_add( FATAL)]: File path (scp) is not yet supported for this command
[4|install_add]: START Sun Feb 26 05:57:04 UTC 2017
[4|install_add]: END SUCCESS /bootflash/isr4300-universalk9.16.05.01.CSCxxxxxxx.dmp.bin
Sun Feb 26 05:57:22 UTC 2017
[5|install_activate]: START Sun Feb 26 05:58:41 UTC 2017

The table below lists the significant fields shown in the display.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Packages</td>
<td>Name of the active model update package.</td>
</tr>
<tr>
<td>Inactive Packages</td>
<td>List of inactive packages.</td>
</tr>
<tr>
<td>Committed Packages</td>
<td>Installed model update packages that have saved or committed changes to the</td>
</tr>
<tr>
<td></td>
<td>hard disk, so that the changes become persistent across reloads.</td>
</tr>
</tbody>
</table>
Cisco Catalyst 3000 Series Switches

The following sample output from the `show install summary` command displays that the update package is now committed, and that it will be persistent across reloads:

```
Device# show install summary

Active Packages:
bootflash:cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin

Inactive Packages:
No packages

Committed Packages:
bootflash:cat3k_caa-universalk9.16.06.01.CSCxxxxxxx.dmp.bin

Uncommitted Packages:
No packages

Device#
```

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncommitted Packages</td>
<td>Model update package activations that are non persistent.</td>
</tr>
</tbody>
</table>

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>install</td>
<td>Installs data model update packages.</td>
</tr>
</tbody>
</table>
show iox-service

To display the status of all IOx services, use the `show iox-service` command in privileged EXEC mode.

```
show iox-service [ { detail } ]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detail</td>
<td>(Optional) Displays detailed information about the application/appliance.</td>
</tr>
</tbody>
</table>

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.5.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

IOx is a Cisco-developed end-to-end application framework that provides application hosting capabilities for different application types on Cisco network platforms. Cisco application hosting framework (CAF) is an IOx Python process that manages virtualized and container applications that run on devices. To enable IOx, configure the `iox` command.

IOXMAN is a process that establishes a tracing infrastructure to provide logging or tracing services for guest applications, except Libvirt, that emulates serial devices.

After configuring this command, you can update the application hosting configuration.

**Example**

The following is sample output from the `show iox-service` command:

```
Device# show iox-service

IOx Infrastructure Summary:
-----------------------------
IOx service (CAF) : Running
IOx service (HA) : Running
IOx service (IOxman) : Running
Libvirtd : Running

The table below lists the fields shown in the display.

Table 2: show iox-service Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOx service (CAF)</td>
<td>Status of the Cisco Application Framework (CAF).</td>
</tr>
<tr>
<td>IOx service (HA)</td>
<td>Status of high availability. High availability must be running, if you have redundant hardware, like a redundant route processor (RP).</td>
</tr>
<tr>
<td>IOx service (IOxman)</td>
<td>Status of the IOx Manager.</td>
</tr>
</tbody>
</table>
The following is sample output from the `show iox-service detail` command:

```
Device# show iox-service detail

IOx Infrastructure Summary:
---------------------------
IOx service (CAF) : Running
IOx service (HA) : Running
IOx service (IOxman) : Running
Libvirtd : Running

------------------ show platform software process list switch active r0 name caf
------------------
Name: run_ioxn_caf.sh
  Process id  : 28445
  Parent process id: 28155
  Group id  : 28445
  Status    : S
  Session id : 9123
  User time  : 5
  Kernel time : 2
  Priority  : 20
  Virtual bytes  : 19939328
  Resident pages : 1036
  Resident limit : 18446744073709551615
  Minor page faults: 4833
  Major page faults: 0

------------------ show platform software process list switch active r0 name libvirtd
------------------
Name: libvirtd.sh
  Process id  : 5757
  Parent process id: 1
  Group id  : 5757
  Status    : S
  Session id : 5757
  User time  : 0
  Kernel time : 0
  Priority  : 20
  Virtual bytes  : 18661376
  Resident pages : 692
  Resident limit : 18446744073709551615
  Minor page faults: 208
  Major page faults: 0

Name: libvirtd
  Process id  : 5782
  Parent process id: 5757
  Group id  : 5757
  Status    : S
  Session id : 5757
  User time  : 6
  Kernel time : 74
  Priority  : 20
```
Virtual bytes : 883945472
Resident pages : 2122
Resident limit : 18446744073709551615
Minor page faults: 2398
Major page faults: 59

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iox</td>
<td>Configure IOx services.</td>
</tr>
</tbody>
</table>
show log profile netconf-yang

To write NETCONF-YANG process logs to a file, use the **show log profile netconf-yang** command in privileged EXEC mode.

**show log profile netconf-yang internal**

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal</td>
<td>Selects all debug logs.</td>
</tr>
</tbody>
</table>

**Note**

This keyword for use by customer support.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Logs are displayed on the device console when the command is executed.

**Example**

The following is sample output from the **show log profile netconf-yang internal** command:

```
Device# show log profile netconf-yang internal
executing cmd on chassis local ...
Collecting files on current[local] chassis.

DECODER ERROR: NOTE: Tracelog may not be generated from clang binary, and is not encoded. Please use native linux tools (vi/less/more/cat...) to read the file

2018/01/24 15:58:50.356 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): gdb port 9919 allocated
2018/01/24 15:58:50.365 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): swift_repl port 8019 allocated
2018/01/24 15:58:50.422 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (info): (std): cat: /tmp/sw/boot/boot_debug.conf: No such file or directory
2018/01/24 15:58:50.427 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (info): (std): /usr/binos/conf/pman.sh: line 424: sigusr1_func: readonly function
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): process scoreboard /tmp/rp/process/pttcd%rp_0_0%0 pttcd%rp_0_0%0.pid is 12040
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): pttcd%rp_0_0%0.gdbport is 9919
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): pttcd%rp_0_0%0.swift_replport is 8019
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): 12040 (process ID) old priority 0, new priority 0
2018/01/24 15:58:50.430 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Launching pttcd on fru rp slot 0 bay 0 instance 0 log /tmp/rp/trace/pttcd_pmanlog
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Hold failures 2, hold interval 1800
2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note):
```

Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
PATH is
/tmp/sw/rp/0/0/rp_daemons/mount/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/conf:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/cpp/bin:/usr/bin:/bin:/sbin:
/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/conf:/usr/binos/bin:/sbin:/bin:/usr/bin:/usr/sbin:/usr/binos/conf:/sbin:/bin:/usr/bin:/usr/sbin:/usr/binos/conf

2018/01/24 15:58:50.439 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): LD_LIBRARY_PATH is

2018/01/24 15:58:50.441 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): PREPROC_OPTIONS ==

2018/01/24 15:58:50.444 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): command line used pttcd >> /tmp/rp/trace/pttcd_pmanlog_cmd 2&>1 &

2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): full_path is /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd

2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Resolved readlink process /tmp/sw/mount
/asr1000rpx86-rpcontrol.BLD_V168_THROTTLE_LATEST_20180122_164958_V16_8_0_177.SSA.pkg

2018/01/24 15:58:50.446 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Full path used to spawn the process: /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd

2018/01/24 15:58:50.452 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Binary_arch set to: [x86_64_cge7]

2018/01/24 15:58:50.460 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (info): (std): chmod: cannot access '/tmp/tmppub/tracekey_cache//tmp/sw/mount/asr1000rpx86-rpcontrol.BLD_V16_8_0_177.SSA.pkg/usr/binos/bin/pttcd': No such file or directory

2018/01/24 15:58:50.461 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): actual pttcd pid is 12542

2018/01/24 15:58:50.461 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Checking for cgroup for PID 12542

2018/01/24 15:58:50.461 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): process scoreboard /tmp/rp/process/pttcd%rp_0_0%0#12040_state marked up

2018/01/24 15:58:50.474 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (ERR): init_callhome() failed

2018/01/24 15:58:50.475 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): oom score adj value is 399

2018/01/24 15:58:50.475 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (info): (std): 12040 (process ID) old priority 0, new priority -6

2018/01/24 15:58:50.475 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [12142]: (note): Wait for signal or process exit: 12542

2018/01/24 15:58:52.077 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): gdb port 9920 allocated

2018/01/24 15:58:52.085 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): swift_repl port 8020 allocated

2018/01/24 15:58:52.150 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (info): (std): cat: /tmp/sw/boot/boot_debug.conf: No such file or directory

2018/01/24 15:58:52.153 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (info): (std): /usr/binos/conf/pman.sh: line 424: sigusr1_func: readonly function

2018/01/24 15:58:52.157 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): process scoreboard /tmp/rp/process/pubd%rp_0_0%0 pubd%rp_0_0%0.pid is 14416

2018/01/24 15:58:52.165 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (info): (std): 14416 (process ID) old priority 0, new priority 0

2018/01/24 15:58:52.166 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): Hold failures 2, hold interval 1800

2018/01/24 15:58:52.166 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): PATH is

2018/01/24 15:58:52.166 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [14520]: (note): PATH is

Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
show log profile netconf-yang
show log profile restconf

To write RESTCONF process logs to a file, use the `show log profile restconf` command in privileged EXEC mode.

```
show log profile restconf internal
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Internal</th>
<th>Selects all debug logs.</th>
</tr>
</thead>
</table>

**Note**

This keyword for use by customer support.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Logs are displayed on the device console when the command is executed.

**Example**

The following is sample output from the `show log profile restconf` command:

```
Device# show log profile restconf internal

executing cmd on chassis local ...
Collecting files on current[local] chassis.
Total # of files collected = 17
Decoding files:
DECODER ERROR: NOTE: Trace log may not be generated from clang binary, and is not encoded. Please use native linux tools (vi/less/more/cat...) to read the file
2018/03/23 13:05:13.945 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): gdb port 9908 allocated
2018/03/23 13:05:13.962 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): swift_repl port 8008 allocated
2018/03/23 13:05:14.041 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (info): (std): cat:
/tmp/sw/boot/boot_debug.conf: No such file or directory
2018/03/23 13:05:14.046 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (info): (std):
/usr/bin/os/conf/pman.sh: line 424: sigusr1_func: readonly function
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): process scoreboard
/tmp/rp/process/pttcdrp_0_0%0 pttcd%rp_0_0%0.pid is 2550
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): pttcd%rp_0_0%0.pid is 2550
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): pttcd%rp_0_0%0.gdbport is 9908
2018/03/23 13:05:14.050 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): pttcd%rp_0_0%0.swift_replport is 8008
2018/03/23 13:05:14.059 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (info): (std): 2550
(process ID) old priority 0, new priority 0
2018/03/23 13:05:14.060 {pttcd_pmanlog_R0-0}{1}: [pttcd_pmanlog] [2628]: (note): Launching
```
pttcd
on fru rp slot 0 bay 0 instance 0 log /tmp/rp/trace/pttcd_pmanlog
2018/03/23 13:05:14.060 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Hold
failures 2,
hold interval 1800
2018/03/23 13:05:14.060 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): PATH is
/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/conf:
/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin:

LD_LIBRARY_PATH is
PREPROC_OPTIONS ==
command line used pttcd >>
/tmp/rp/trace/pttcd_pmanlog_cmd 2>&1 &
2018/03/23 13:05:14.068 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): full_path is
/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd
2018/03/23 13:05:14.069 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Resolved
readlink process
2018/03/23 13:05:14.069 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Full path used
to spawn the process:
/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pttcd
2018/03/23 13:05:14.076 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Binary_arch
set to: [x86_64_cge7]
2018/03/23 13:05:14.087 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (info): (std):
chmod: cannot access
"/tmp/tmppub/tracekey_cache"/tmp/sw/mount/asr1000px86-rpcontrol.2018-03-07_18.30_rifu.SSA.pkg
2018/03/23 13:05:14.088 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): actual
pttcd pid is 2936
2018/03/23 13:05:14.088 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Checking
cfgroup for PID 2936
2018/03/23 13:05:14.088 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): /tmp/rp/pwp/process_state/pttcd%rp_0_0%0#2550_state marked up
2018/03/23 13:05:14.097 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): oom score
adj value is 399
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (ERR): init_callhome() failed
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (info): (std):
2550 (process ID) old priority 0, new priority -6
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Wait for
signal or process exit: 2936
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Wait for
signal or process exit: 2936
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Wait for
signal or process exit: 2936
2018/03/23 13:05:14.102 [pttcd_pmanlog_R0-0]{1}: [pttcd_pmanlog] [2628]: (note): Wait for
signal or process exit: 2936
2018/03/23 13:05:16.895 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (note): gdb port
9920 allocated
2018/03/23 13:05:16.904 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (note): swift repl
port 8020 allocated
2018/03/23 13:05:16.978 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (info): (std):
cat: /tmp/sw/boot/boot_debug.conf: No such file or directory
2018/03/23 13:05:16.983 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (info): (std):
/usr/binos/conf/pman.sh: line 424: sigusr1_func: readonly function
2018/03/23 13:05:16.987 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (note): process
scoreboard
2018/03/23 13:05:16.996 [pubd_pmanlog_R0-0]{1}: [pubd_pmanlog] [4998]: (note): state is
programmability
2018/03/23 13:05:16.987 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): pubd\rp_0_0\0.gdbport is 9920
2018/03/23 13:05:16.987 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): pubd\rp_0_0\0.swift_replport is 8020
2018/03/23 13:05:16.996 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (info): (std): 4922 (process ID) old priority 0, new priority 0
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Launching pubd on fru rp slot 0 bay 0 instance 0 log /tmp/rp/trace/pubd_pmanlog
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Hold failures 2, hold interval 1800
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PATH is /tmp/sw/rp/0/0/rp_daemons/mount/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/conf:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/sbin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin:/tmp/sw/rp/0/0/rp_daemons/mount/usr/cpp/bin:/usr/bin:/bin:/sbin:/usr/binos/conf:/usr/binos/bin:/usr/sbin:/usr/bin:/usr/binos/conf:/usr/binos/bin:
2018/03/23 13:05:16.997 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): LD_LIBRARY_PATH is
2018/03/23 13:05:17.001 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): PREPROC_OPTIONS ==
2018/03/23 13:05:17.001 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): command line used pubd >> /tmp/rp/trace/pubd_pmanlog_cmd 2>&1 &
2018/03/23 13:05:17.007 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): full_path is /tmp/sw/rp/0/0/rp_daemons/mount/usr/binos/bin/pubd
2018/03/23 13:05:17.009 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Resolved readlink process /tmp/sw/mount/asr1000rpx86-rpcontrol.2018-03-07_18.30_rifu.SSA.pkg/usr/binos/bin/pubd
2018/03/23 13:05:17.009 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (note): Binary_arch set to: [x86_64_cge7]
2018/03/23 13:05:17.030 {pubd_pmanlog_R0-0}{1}: [pubd_pmanlog] [4998]: (info): (std): chmod: cannot access
! ! !
show netconf-yang

To display information about NETCONF-YANG processes, use the `show netconf-yang` command in privileged EXEC mode.

```
show netconf-yang {datastores |sessions [detail |session-id session-id] |statistics} [R0 |R1 |RP {active |standby}]
```

**Syntax Description**

- **datastores**: Displays information about NETCONF-YANG datastores.
- **sessions**: Displays information about NETCONF-YANG sessions.
- **detail**: (Optional) Displays detailed information about NETCONF-YANG sessions.
- **session-id**: (Optional) Displays information about the specified session. Valid values are from 1 to 4294967295.
- **statistics**: Displays information about NETCONF-YANG statistics.
- **R0**: (Optional) Displays information about the Route Processor (RP) slot 0.
- **R1**: (Optional) Displays information about the RP slot 1.
- **RP**: (Optional) Displays information about the RP.
- **active**: (Optional) Displays information about the active instance of the RP.
- **standby**: (Optional) Displays information about the standby instance of the RP.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.8.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command displays information about global locks applied on the running datastore, candidate datastore, and startup datastore.

The `active` and `standby` keywords are only applicable to devices that supports both active and redundant route processors.

**Example**

This sample output from the `show netconf-yang datastores` commands displays the sessions that have global locks:

```
Device# show netconf-yang datastores

Datastore Name : running
Globally Locked By Session : 42
```
The table below lists the significant fields shown in the display.

Table 3: show netconf-yang datastores Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore Name</td>
<td>Name of the datastore supported by the device.</td>
</tr>
<tr>
<td>Globally Locked By Session</td>
<td>Number of NETCONF-YANG sessions that have the lock on the running datastore.</td>
</tr>
<tr>
<td>Globally Locked Time</td>
<td>Time when a NETCONF-YANG session acquires the lock.</td>
</tr>
</tbody>
</table>

The following is sample output from the `show netconf-yang sessions` command:

```
Device# show netconf-yang sessions

R: Global-lock on running datastore
C: Global-lock on candidate datastore
S: Global-lock on startup datastore

Number of sessions : 10

+----------+----------+----------+----------+----------+----------+
<table>
<thead>
<tr>
<th>session-id</th>
<th>transport</th>
<th>username</th>
<th>source-host</th>
<th>global-lock</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>42</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>44</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>46</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>48</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>50</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>52</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>54</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>56</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>58</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
</tbody>
</table>
```

The table below lists the significant fields shown in the display.

Table 4: show netconf-yang sessions Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session-id</td>
<td>Session identifier.</td>
</tr>
<tr>
<td>transport</td>
<td>Transport protocol used for session.</td>
</tr>
<tr>
<td>username</td>
<td>Client that is authenticated by the NETCONF-YANG system.</td>
</tr>
<tr>
<td>source-host</td>
<td>IP address of the client.</td>
</tr>
<tr>
<td>global-lock</td>
<td>True for sessions holding a global lock, and NONE, if there are no global locks.</td>
</tr>
</tbody>
</table>

Globally Locked Time : 2018-01-15T14:25:14-05:00
This is sample output from the **show netconf-yang statistics** command:

```
Device# show netconf-yang statistics

netconf-start-time : 2018-01-15T12:51:14-05:00
in-rpcs : 0
in-bad-rpcs : 0
out-rpc-errors : 0
out-notifications : 0
in-sessions : 10
dropped-sessions : 0
in-bad-hellos : 0
```

The table below lists the significant fields shown in the display.

**Table 5: show netconf-yang statistics Field Descriptions**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netconf-start-time</td>
<td>Session establishment time.</td>
</tr>
<tr>
<td>in-rpcs</td>
<td>Total number of correct incoming RPCs.</td>
</tr>
<tr>
<td>in-bad-rpcs</td>
<td>Total number of incorrect incoming RPCs.</td>
</tr>
<tr>
<td>out-rpc-errors</td>
<td>Total number of RPC reply messages that indicate RPC errors.</td>
</tr>
<tr>
<td>out-notifications</td>
<td>Total number of outgoing notifications.</td>
</tr>
<tr>
<td>in-sessions</td>
<td>Total number of active NETCONF sessions.</td>
</tr>
<tr>
<td>dropped-sessions</td>
<td>Total number of dropped NETCONF sessions.</td>
</tr>
</tbody>
</table>
To display the status of the software processes required to support NETCONF-YANG, use the `show platform yang-management process` in privileged EXEC mode.

```
show platform yang-management process
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>monitor</code></td>
<td>(Optional) Displays detailed information about processes that are running.</td>
</tr>
<tr>
<td><code>switch switch-number</code></td>
<td>(Optional) Displays information about the specified switch.</td>
</tr>
<tr>
<td><code>active</code></td>
<td>(Optional) Displays information about the active instance of the switch.</td>
</tr>
<tr>
<td><code>standby</code></td>
<td>(Optional) Displays information about the standby instance of the switch.</td>
</tr>
<tr>
<td><code>R0</code></td>
<td>(Optional) Displays information about the Route Processor (RP) slot zero.</td>
</tr>
</tbody>
</table>

### Command Modes

Privileged EXEC (#)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.3.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

The following is sample output from the `show platform software yang-management process` command:

```
Device# show platform software yang-management process

confd : Running
nessd : Running
syncfd : Running
ncsshd : Running
dmiauthd : Running
vtyserverutilld : Running
opdatamgrd : Running
nginx : Running
ndbmand : Running
```

The table below lists the significant fields shown in the display.
Table 6: show platform software yang-management process Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>confd</td>
<td>Configuration daemon</td>
</tr>
<tr>
<td>nesd</td>
<td>Network element synchronizer daemon</td>
</tr>
<tr>
<td>syncfd</td>
<td>Sync from daemon</td>
</tr>
<tr>
<td>ncsshd</td>
<td>NETCONF Secure Shell (SSH) daemon</td>
</tr>
<tr>
<td>dmiauthd</td>
<td>Device management interface (DMI) auth</td>
</tr>
<tr>
<td>vtyserverutild</td>
<td>VTY server util daemon</td>
</tr>
<tr>
<td>opdatamgrd</td>
<td>Operational Data Manager daemon</td>
</tr>
<tr>
<td>nginx</td>
<td>NGINX web server</td>
</tr>
<tr>
<td>ndbmand</td>
<td>NETCONF database manager</td>
</tr>
</tbody>
</table>

The following is sample output from the `show platform software yang-management process monitor` command:

```
Device# show platform software yang-management process monitor
COMMAND PID S VSZ RSS %CPU %MEM ELAPSED
nginx 24689 S 139328 11996 0.0 0.2 24-02:00:55
nginx 24695 S 146544 6824 0.0 0.1 24-02:00:55
```

The table below lists the significant fields shown in the display.

Table 7: show platform software yang-management process monitor Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMAND</td>
<td>Command name</td>
</tr>
<tr>
<td>PID</td>
<td>Process ID</td>
</tr>
<tr>
<td>S</td>
<td>Process state</td>
</tr>
<tr>
<td>VSZ</td>
<td>Virtual memory size (in KB)</td>
</tr>
<tr>
<td>RSS</td>
<td>Resident set size (in KB)</td>
</tr>
<tr>
<td>%CPU</td>
<td>CPU usage percentage</td>
</tr>
<tr>
<td>%MEM</td>
<td>Memory usage percentage</td>
</tr>
<tr>
<td>ELAPSED</td>
<td>Elapsed execution time</td>
</tr>
</tbody>
</table>
show openflow hardware capabilities

To display information about OpenFlow hardware capabilities, use the show openflow hardware capabilities command in privileged EXEC mode.

show openflow hardware capabilities [{pipeline 1}]

Syntax Description

<table>
<thead>
<tr>
<th>pipeline 1</th>
<th>Displays information about the OpenFlow pipeline ID.</th>
</tr>
</thead>
</table>

Command Modes

Privileged EXEC (#)

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

Usage Guidelines

Example

The following is sample output from the show openflow hardware capabilities command:

Device# show openflow hardware capabilities

Max Interfaces: 1000
Aggregated Statistics: YES
Pipeline ID: 1
Pipeline Max Flows: 2322
Max Flow Batch Size: 100
Statistics Max Polling Rate (flows/sec): 10000
Pipeline Default Statistics Collect Interval: 5
Flow table ID: 0
Max Flow Batch Size: 100
Max Flows: 1022
Bind Subintfs: FALSE
Primary Table: TRUE
Table Programmable: TRUE
Miss Programmable: TRUE
Number of goto tables: 1
Goto table id: 1
Number of miss goto tables: 1
Miss Goto table id: 1
Stats collection time for full table (sec): 1
!
!

The following is sample output from the show openflow hardware capabilities pipeline 1 command:

Device# show openflow hardware capabilities pipeline 1

Max Interfaces: 1000
Aggregated Statistics: YES
Pipeline ID: 1
Pipeline Max Flows: 128
Max Flow Batch Size: 100
Statistics Max Polling Rate (flows/sec): 10000
Pipeline Default Statistics Collect Interval: 5

Flow table ID: 0

Max Flow Batch Size: 100
Max Flows: 32
Bind Subintfs: FALSE
Primary Table: TRUE
Table Programmable: TRUE
Miss Programmable: TRUE
Number of goto tables: 1
Goto table id: 1
Number of miss goto tables: 1
Miss Goto table id: 1
Stats collection time for full table (sec): 1

<table>
<thead>
<tr>
<th>Match Capabilities</th>
<th>Match Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>ethernet mac</td>
<td>bitmask</td>
</tr>
<tr>
<td>ethernet type</td>
<td>optional</td>
</tr>
<tr>
<td>VLAN ID</td>
<td>optional</td>
</tr>
<tr>
<td>in port (virtual or physical)</td>
<td>optional</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Actions</th>
<th>Count</th>
<th>Limit</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>set vlan id</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>push vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>pop vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>drop packet</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>perform another lookup in the specified table</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>forward pkt via the specific group</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>specified interface</td>
<td>64</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>controller</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>set input port</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Miss actions</th>
<th>Count</th>
<th>Limit</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>set vlan id</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>push vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>pop vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>drop packet</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>perform another lookup in the specified table</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>forward pkt via the specific group</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>specified interface</td>
<td>64</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>controller</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>set input port</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

Flow table ID: 1

Max Flow Batch Size: 100
Max Flows: 32
Bind Subintfs: FALSE
Primary Table: FALSE
Table Programmable: TRUE
Miss Programmable: TRUE
Number of goto tables: 2
Goto table id: 2 3
Number of miss goto tables: 1
Miss Goto table id: 2
Stats collection time for full table (sec): 1

<table>
<thead>
<tr>
<th>Match Capabilities</th>
<th>Match Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>ethernet mac</td>
<td>bitmask</td>
</tr>
<tr>
<td>ethernet type</td>
<td>optional</td>
</tr>
<tr>
<td>VLAN ID</td>
<td>optional</td>
</tr>
<tr>
<td>in port (virtual or physical)</td>
<td>optional</td>
</tr>
</tbody>
</table>

Programmability Command Reference, Cisco IOS XE Gibraltar 16.10.x
**ethernet mac destination**  bitmask
**ethernet mac source**  optional
**ethernet type**  optional
**VLAN ID**  optional
**in port (virtual or physical)**  optional

<table>
<thead>
<tr>
<th>Actions</th>
<th>Count</th>
<th>Limit</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>set eth destination mac</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>set vlan id</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>push vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>pop vlan tag</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>drop packet</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>perform another lookup in the specified table</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>forward pkt via the specific group</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>specified interface</td>
<td>64</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>controller</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>set input port</td>
<td>1</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

The output fields are self-explanatory.
show openflow interface

To display information about OpenFlow interfaces, use the `show openflow interface` command in privileged EXEC mode.

```
show openflow interface {[detail]}
```

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Command Modes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>detail</td>
<td>Privileged EXEC(#)</td>
<td>Displays detailed administrative and operational state information.</td>
</tr>
</tbody>
</table>

**Command Modes**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

**Example**

The following is sample output from the `show openflow interface detail` command:

```
Device# show openflow interface detail
GigabitEthernet1/0/1, admin up, oper up
GigabitEthernet1/0/2, admin up, oper up
GigabitEthernet1/0/3, admin up, oper up
GigabitEthernet1/0/4, admin up, oper up
GigabitEthernet1/0/5, admin up, oper down
GigabitEthernet1/0/6, admin up, oper down
GigabitEthernet1/0/7, admin up, oper down
GigabitEthernet1/0/8, admin up, oper down
GigabitEthernet1/0/9, admin up, oper up
GigabitEthernet1/0/10, admin up, oper up
GigabitEthernet1/0/11, admin up, oper up
GigabitEthernet1/0/12, admin up, oper up
GigabitEthernet1/0/13, admin up, oper down
GigabitEthernet1/0/14, admin up, oper down
GigabitEthernet1/0/15, admin up, oper down
GigabitEthernet1/0/16, admin up, oper down
GigabitEthernet1/0/17, admin up, oper down
GigabitEthernet1/0/18, admin up, oper down
GigabitEthernet1/0/19, admin up, oper up
GigabitEthernet1/0/20, admin up, oper up
GigabitEthernet1/0/21, admin up, oper up
GigabitEthernet1/0/22, admin up, oper up
GigabitEthernet1/0/23, admin up, oper down
GigabitEthernet1/0/24, admin up, oper down
GigabitEthernet1/1/1, admin up, oper down
GigabitEthernet1/1/2, admin up, oper down
GigabitEthernet1/1/3, admin up, oper down
GigabitEthernet1/1/4, admin up, oper down
TenGigabitEthernet1/1/1, admin up, oper down
TenGigabitEthernet1/1/2, admin up, oper down
TenGigabitEthernet1/1/3, admin up, oper down
TenGigabitEthernet1/1/4, admin up, oper down
TenGigabitEthernet1/1/5, admin up, oper down
TenGigabitEthernet1/1/6, admin up, oper down
```
show openflow interface

TenGigabitEthernet1/1/7, admin up, oper down
TenGigabitEthernet1/1/8, admin up, oper down
FortyGigabitEthernet1/1/1, admin up, oper down
FortyGigabitEthernet1/1/2, admin up, oper down
TwentyFiveGigE1/1/1, admin up, oper down
TwentyFiveGigE1/1/2, admin up, oper down

The output fields are self-explanatory.
To display OpenFlow switch flows, use the `show openflow switch flows` command in privileged EXEC mode.

```
show openflow switch number flows [{brief | list | summary }][{controller | default | del-pending | fixed | pending}][{brief | list | summary }]]
```

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>number</code></td>
<td>OpenFlow switch number.</td>
</tr>
<tr>
<td><code>brief</code></td>
<td>(Optional) Displays brief information about OpenFlow switch flows.</td>
</tr>
<tr>
<td><code>list</code></td>
<td>Displays all flows; one flow entry per line.</td>
</tr>
<tr>
<td><code>summary</code></td>
<td>Displays the count of flows.</td>
</tr>
<tr>
<td><code>configured</code></td>
<td>Displays information about the configured flows.</td>
</tr>
<tr>
<td><code>controller</code></td>
<td>Displays information about the controller-programmed flows.</td>
</tr>
<tr>
<td><code>default</code></td>
<td>Displays information about the default flows.</td>
</tr>
<tr>
<td><code>del-pending</code></td>
<td>Displays information about flows that are scheduled to be deleted.</td>
</tr>
<tr>
<td><code>fixed</code></td>
<td>Displays information about fixed flows.</td>
</tr>
<tr>
<td><code>pending</code></td>
<td>Displays all pending flows.</td>
</tr>
</tbody>
</table>

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Example**

The following is sample output from the `show openflow switch number flows brief` command:

```
Device# show openflow switch 1 flows brief
Logical Switch Id: 1
Total flows: 10

Flow: 1 Match: any Actions: drop
```
Priority: 0, Table: 0, Cookie: 0x0, Duration: 140.088s, Packets: 2, Bytes: 500
Flow: 2 Match: tcp, in_port=0, tp_src=1 Actions: output:1
  Priority: 11111, Table: 0, Cookie: 0x1, Duration: 130.642s, Packets: 0, Bytes: 0
Flow: 3 Match: any Actions: drop
  Priority: 0, Table: 1, Cookie: 0x0, Duration: 140.088s, Packets: 0, Bytes: 0
Flow: 4 Match: any Actions: drop
  Priority: 0, Table: 2, Cookie: 0x0, Duration: 140.088s, Packets: 0, Bytes: 0
Flow: 5 Match: any Actions: drop
  Priority: 0, Table: 3, Cookie: 0x0, Duration: 140.087s, Packets: 0, Bytes: 0
Flow: 6 Match: any Actions: drop
  Priority: 0, Table: 4, Cookie: 0x0, Duration: 140.087s, Packets: 0, Bytes: 0
Flow: 7 Match: any Actions: drop
  Priority: 0, Table: 5, Cookie: 0x0, Duration: 140.086s, Packets: 0, Bytes: 0
Flow: 8 Match: any Actions: drop
  Priority: 0, Table: 6, Cookie: 0x0, Duration: 140.086s, Packets: 0, Bytes: 0
Flow: 9 Match: any Actions: drop
  Priority: 0, Table: 7, Cookie: 0x0, Duration: 140.085s, Packets: 0, Bytes: 0
Flow: 10 Match: any Actions: drop
  Priority: 0, Table: 8, Cookie: 0x0, Duration: 140.085s, Packets: 0, Bytes: 0

Device#

The following is sample from the `show openflow switch number flows summary` command:

Device# `show openflow switch 1 flows summary`

Logical Switch Id: 1
Switch flow count: 10

The output fields are self-explanatory.

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>feature openflow</td>
<td>Enables the OpenFlow feature.</td>
</tr>
<tr>
<td>openflow</td>
<td>Enables OpenFlow configuration and enters OpenFlow configuration mode.</td>
</tr>
</tbody>
</table>
show telemetry ietf subscription

To display information about telemetry subscriptions on a device, use the show telemetry ietf subscription command in user EXEC or privileged EXEC mode.

```
show telemetry ietf subscription {subscription-ID | all | configured | dynamic} [{brief | detail}]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subscription-ID</td>
<td>Subscription ID. Valid values are from 0 to 4294967295.</td>
</tr>
<tr>
<td>all</td>
<td>Displays all subscription information.</td>
</tr>
<tr>
<td>configured</td>
<td>Displays a list of subscriptions configured via CLI or NETCONF set config.</td>
</tr>
<tr>
<td>dynamic</td>
<td>Displays information about dynamic subscriptions created using the &lt;establish-subscription&gt; RPC.</td>
</tr>
<tr>
<td>brief</td>
<td>(Optional) Displays a brief summary of the subscription information.</td>
</tr>
<tr>
<td>detail</td>
<td>(Optional) Displays the subscription information in detail.</td>
</tr>
</tbody>
</table>

**Command Modes**

User EXEC (>

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Everest 16.6.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Use the show telemetry ietf subscription command or the <get> RPC to retrieve the list of current subscription details on a device.

**Example**

The following is sample output from the show telemetry ietf subscription subscription-ID detail command:

```
Device# telemetry ietf subscription 2147483667 detail

Telemetry subscription detail:
  Subscription ID: 2147483667
  State: Valid
  Stream: yang-push
  Encoding: encode-xml
  Filter:
    Filter type: xpath
```
XPath: /mdt-oper:mdt-oper-data/mdt-subscriptions
Update policy:
  Update Trigger: periodic
  Period: 1000
Notes:

The following is sample output from the **show telemetry ietf subscription dynamic brief** command:

```
Device# show telemetry ietf subscription dynamic brief
Telemetry subscription brief

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>State</th>
<th>Filter type</th>
</tr>
</thead>
<tbody>
<tr>
<td>2147483667</td>
<td>Dynamic</td>
<td>Valid</td>
<td>xpath</td>
</tr>
<tr>
<td>2147483668</td>
<td>Dynamic</td>
<td>Valid</td>
<td>xpath</td>
</tr>
<tr>
<td>2147483669</td>
<td>Dynamic</td>
<td>Valid</td>
<td>xpath</td>
</tr>
</tbody>
</table>
```

The table below lists the significant fields shown in the display.

**Table 8: show telemetry ietf subscription Field Descriptions**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscription ID</td>
<td>Subscription identifier</td>
</tr>
<tr>
<td>State</td>
<td>Validity of a configured subscription. State will always be valid for dynamic subscriptions. For example, a configured subscription can be in a half-configured state, and therefore invalid. However, if a dynamic establish subscription is invalid, an error RPC response is sent back, and the subscription will not appear in this table.</td>
</tr>
<tr>
<td>Stream</td>
<td>Type of streaming used for subscriptions. Only YANG-push is supported.</td>
</tr>
<tr>
<td>Encoding</td>
<td>Specifies encode-xml as the encoding type.</td>
</tr>
<tr>
<td>Filter Type</td>
<td>Type of filter used for subscriptions. Only XPath is supported.</td>
</tr>
<tr>
<td>XPath</td>
<td>XPath filter type or how the subscribed information was selected.</td>
</tr>
<tr>
<td>Update Trigger</td>
<td>Type of trigger to update subscriptions.</td>
</tr>
<tr>
<td>Period</td>
<td>Periodic timer configured to trigger an update. Values are specified in centiseconds (1/100 of a second).</td>
</tr>
<tr>
<td>Notes</td>
<td>A brief explanation about why a subscription is invalid. But for dynamic subscriptions, this field is always be empty.</td>
</tr>
</tbody>
</table>
show telemetry ietf subscription

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Subscription ID.</td>
</tr>
</tbody>
</table>
To configure a logical switch and enter OpenFlow switch configuration mode, use the `switch` command in OpenFlow configuration mode. To disable the logical switch configuration, use the `no` form of this command.

```
switch 1 pipeline 1
no switch 1 pipeline 1
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>1</code></td>
<td>Configures the OpenFlow logical switch ID.</td>
</tr>
<tr>
<td><code>pipeline 1</code></td>
<td>Configures the OpenFlow pipeline ID.</td>
</tr>
</tbody>
</table>

**Command Default**

The OpenFlow logical switch is not configured.

**Command Modes**

OpenFlow configuration (config-openflow)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Fuji 16.9.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Example**

The following example shows how to configure a logical switch and pipeline:

```
Device# configure terminal
Device(config)# feature openflow
Device(config)# openflow
Device(config-openflow)# switch 1 pipeline 1
Device(config-openflow-switch)#
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>feature openflow</td>
<td>Enables the OpenFlow feature.</td>
</tr>
<tr>
<td>openflow</td>
<td>Enables OpenFlow configuration and enters OpenFlow configuration mode.</td>
</tr>
</tbody>
</table>
stream

To configure telemetry stream for the subscription, use the `stream` command in telemetry-subscription configuration mode. To disable the configuration, use the `no` form of this command.

```
stream yang-push
no stream yang-push
```

**Syntax Description**

| yang-push | Configures a yang-push stream. |

**Command Modes**

Telemetry-subscription configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Gibraltar 16.10.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Sources of telemetry data in a subscription are specified by the use of a stream and a filter. The term stream refers to a related set of events. RFC 5277 defines an event stream as a set of event notifications matching some forwarding criteria.

The yang-push stream is the data in configuration and operational databases that is described by a supported YANG model. This stream supports an XPath filter to specify what data is of interest within the stream, and where the XPath expression is based on the YANG model that defines the data of interest. Update notifications for this stream may be sent either when data changes or at fixed periods, but not for both, for a given subscription. Subscriptions for data that does not currently exist are permitted, and these run as normal subscriptions.

**Example**

The following example shows how to configure telemetry stream for the subscription:

```
Device(config)# telemetry ietf subscription 101
Device(config-mdt-subs)# stream yang-push
```
telemetry ietf subscription

To configure telemetry subscription, use the `telemetry ietf subscription` command in global configuration mode. To disable the configuration, use the `no` form of this command.

```
telemetry ietf {subscription sub-id}
no telemetry ietf {subscription sub-id}
```

### Syntax Description

**subscription sub-id** Configures a telemetry subscription. Valid values are from 0 to 2147483647.

### Command Modes

Global configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Gibraltar 16.10.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Example

The following example shows how to configure an telemetry subscription:

```
Device(config)# telemetry ietf subscription 101
```
update-policy

To configure update policy for subscription, use the **update-policy** command in telemetry-subscription configuration mode. To disable the configuration, use the **no** form of this command.

```
update-policy {on-change|periodic period}
no update-policy {on-change|periodic period}
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>on-change</strong></td>
<td>Enables on-change updates.</td>
</tr>
<tr>
<td><strong>periodic period</strong></td>
<td>Enable on-change updates. Valid values are from 100 to 4294967295.</td>
</tr>
</tbody>
</table>

**Command Modes**

Telemetry-subscription configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS XE Gibraltar 16.10.1</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

**Example**

The following example shows how to configure a periodic update policy for a subscription:

```
Device(config)# telemetry ietf subscription 101
Device(config-mdt-subs)# update-policy periodic 6000
```