Network Powered Lighting Configuration Guide, Cisco IOS XE Fuji 16.9.x (Catalyst 9300 Switches)

First Published: 2018-07-18

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

CONTENTS

PART I

Network Powered Lighting

CHAPTER 1

Configuring COAP Proxy Server

Restrictions for the COAP Proxy Server

The following restrictions apply to COAP proxy server:

- Switch cannot advertise itself as CoAP client using ipv6 broadcast (CSCuw26467).
- Support for Observe Not Implemented.
- Blockwise requests are not supported. We handle block-wise responses and can generate block-wise responses.
- DTLS Support is for the following modes only RawPublicKey and Certificate Based.
- Switch does not act as DTLS client. DTLS for endpoints only.
- Endpoints are expected to handle and respond with CBOR payloads.
- Client side requests are expected to be in JSON.
- Switch cannot advertise itself to other Resource Directories as IPv6, due to an IPv6 broadcast issue.

Information About the COAP Proxy Server

The COAP protocol is designed for use with constrained devices. COAP works in the same way on constrained devices as HTTP works on servers in accessing information.

The comparison of COAP and HTTP is shown below:

- In the case of a webserver: **HTTP** is the protocol; **TCP** is the transport; and **HTML** is the most common information format transported.

- In case of a constrained device: **COAP** is the protocol; **UDP** is the transport; and **JSON/link-format/CBOR** is the popular information format.

COAP provides a means to access and control device using a similar **GET**/**POST** metaphor and restful API as in HTTP.

How to Configure the COAP Proxy Server

To configure the COAP proxy server, you can configure the COAP Proxy and COAP Endpoints in the Configuration mode.

The commands are: **coap [proxy | endpoints]**.

Configuring the COAP Proxy

To start or stop the COAP proxy on the switch, perform the steps given below:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **coap proxy**
4. **security** [**none** [[**ipv4** | **ipv6**] {*ip-address ip-mask/prefix*} | **list** {*ipv4-list name* | *ipv6-list-name*}] | **dtls** [**id-trustpoint** {*identity-trustpoint label*}] [**verification-trustpoint** {*verification-trustpoint*} | [**ipv4** | **ipv6** {*ip-address ip-mask/prefix*}] | **list** {*ipv4-list name* | *ipv6-list-name*}]]
5. **max-endpoints** {*number*}
6. **port-unsecure** {*port-num*}
7. **port-dtls** {*port-num*}
8. **resource-directory** [ipv4 | ipv6] {*ip-address*}]
9. **list** [**ipv4** | **ipv6**] {*list-name*}
10. **start**
11. **stop**
12. **exit**
13. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**coap proxy** **Example:** `Device(config)# coap proxy`	Enters the COAP proxy sub mode. **Note** To stop the coap proxy and delete all configurations under coap proxy, use the **no coap proxy** command.
Step 4	**security** [**none** [[**ipv4** | **ipv6**] {*ip-address ip-mask/prefix*} | **list** {*ipv4-list name* | *ipv6-list-name*}] | **dtls** [**id-trustpoint** {*identity-trustpoint label*}] [**verification-trustpoint** {*verification-trustpoint*} | [**ipv4** | **ipv6** {*ip-address ip-mask/prefix*}] | **list** {*ipv4-list name* | *ipv6-list-name*}]] **Example:** `Device(config-coap-proxy)# security none ipv4 1.1.0.0 255.255.0.0`	Takes the encryption type as argument. The two security modes supported are **none** and **dtls** - none - Indicates no security on that port. With **security none**, a maximum of 5 ipv4 and 5 ipv6 addresses can be associated. - dtls - The DTLS security takes RSA trustpoint and Verification trustpoint which are optional. Without Verification trustpoint it does the normal Public Key Exchange. With **security dtls**, a maximum of 5 ipv4 and 5 ipv6 addresses can be associated. **Note** To delete all security configurations under coap proxy, use the **no security** command.

Step 5	**max-endpoints** {*number*}
	Example:
	`Device(config-coap-proxy)#max-endpoints 10`
		(Optional) Specifies the maximum number of endpoints that can be learnt on the switch. The default value is 10. The range is 1 to 500.
		Note To delete all max-endpoints configured under coap proxy, use the **no max-endpoints** command.

Step 6	**port-unsecure** {*port-num*}
	Example:
	`Device(config-coap-proxy)#port-unsecure 5683`
		(Optional) Configures a port other than the default 5683. The range is 1 to 65000.
		Note To delete all port configurations under coap proxy, use the **no port-unsecure** command.

Step 7	**port-dtls** {*port-num*}
	Example:
	`Device(config-coap-proxy)#port-dtls 5864`
		(Optional) Configures a port other than the default 5684.
		Note To delete all dtls port configurations under coap proxy, use the **no port-dtls** command.

Step 8	**resource-directory** [ipv4 | ipv6] {*ip-address*}] **Example:**	Configures a unicast upstream resource directory server to which the switch can act as a COAP client.

	Command or Action	Purpose
	`Device(config-coap-proxy)#`**`resource-directory ipv4 192.168.1.1`**	With **resource-directory**, a maximum of 5 of ipv4 and 5 ipv6, ip addresses can be configured. **Note** To delete all resource directory configurations under coap proxy, use the **no resource-directory** command.

Step 9	**list** [**ipv4** | **ipv6**] {*list-name*}
	Example:
	`Device(config-coap-proxy)#`**`list ipv4 trial_list`**
		(Optional) Restricts the IP address range where the lights and their resources can be learnt. Creates a named list of ip address/masks, to be used in the **security** [**none** | **dtls**] command options above.
		With **list**, a maximum of 5 ip-lists can be configured, irrespective of ipv4 or ipv6. We can configure a max of 5 ip addresses per ip-list.
		Note To delete any ip list on the COAP proxy server, use the **no list** [**ipv4** | **ipv6**] {*list-name*} command.

Step 10	**start** **Example:** `Device(config-coap-proxy)#`**`start`**	Starts the COAP proxy on this switch.
Step 11	**stop** **Example:** `Device(config-coap-proxy)#`**`stop`**	Stops the COAP proxy on this switch.
Step 12	**exit** **Example:** `Device(config-coap-proxy)# `**`exit`**	Exits the COAP proxy sub mode.

Step 13	**end**
	Example:
	`Device(config)# `**`end`**
		Returns to privileged EXEC mode.

Configuring COAP Endpoints

To configure the COAP Proxy to support multiple IPv4/IPv6 static-endpoints, perform the steps given below:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **coap endpoint** [**ipv4** | **ipv6**] {*ip-address*}
4. **exit**
5. **end**

DETAILED STEPS

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

Step 3	**coap endpoint** [**ipv4** | **ipv6**] {*ip-address*}
	Example:
	`Device(config)#coap endpoint ipv4  1.1.1.1`
	`Device(config)#coap endpoint ipv6  2001::1`
		Configures the static endpoints on the switch.
		• **ipv4** - Configures the IPv4 Static endpoints.
		• **ipv6** - Configures the IPv6 Static endpoints.
		Note To stop the coap proxy on any endpoint, use the **no coap endpoint** [**ipv4** |**ipv6**] {*ip-address*} command.

Step 4	**exit** **Example:** `Device(config-coap-endpoint)# exit`	Exits the COAP endpoint sub mode.

Step 5	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Configuration Examples for the COAP Proxy Server

Examples: Configuring the COAP Proxy Server

This example shows how you can configure the port number 5683 to support a maximum of 10 endpoints.

```
Device#coap proxy security none ipv4 2.2.2.2 255.255.255.0 port 5683 max-endpoints 10
```

This example shows how to configure COAP proxy on *ipv4 1.1.0.0 255.255.0.0* with **no** security settings.

```
Device(config-coap-proxy)# security ?
  dtls  dtls
  none  no security

Device(config-coap-proxy)#security none ?
  ipv4    IP address range on which to learn lights
  ipv6    IPv6 address range on which to learn lights
  list    IP address range on which to learn lights

Device(config-coap-proxy)#security none ipv4 ?
  A.B.C.D  {/nn || A.B.C.D}  IP address range on which to learn lights

Device(config-coap-proxy)#security none ipv4 1.1.0.0 255.255.0.0
```

This example shows how to configure COAP proxy on *ipv4 1.1.0.0 255.255.0.0* with **dtls id trustpoint** security settings.

```
Device(config-coap-proxy)#security dtls ?
  id-trustpoint DTLS RSA and X.509 Trustpoint Labels
   ipv4 IP address range on which to learn lights
   ipv6 IPv6 address range on which to learn lights
   list IP address range on which to learn lights

Device(config-coap-proxy)#security dtls id-trustpoint ?
  WORD  Identity TrustPoint Label

Device(config-coap-proxy)#security dtls id-trustpoint RSA-TRUSTPOINT ?
  verification-trustpoint  Certificate Verification Label
   <cr>

Device(config-coap-proxy)#security dtls id-trustpoint RSA-TRUSTPOINT

Device(config-coap-proxy)#security dtls ?
  id-trustpoint DTLS RSA and X.509 Trustpoint Labels
   ipv4 IP address range on which to learn lights
   ipv6 IPv6 address range on which to learn lights
   list IP address range on which to learn lights

Device(config-coap-proxy)# security dtls ipv4 1.1.0.0 255.255.0.0
```

Note For configuring **ipv4 / ipv6 / list**, the **id-trustpoint** and (optional) **verification-trustpoint**, should be pre-configured, else the system shows an error.

This example shows how to configure a Trustpoint. This is a pre-requisite for COAP **security dtls** with **id trustpoint** configurations.

```
ip domain-name myDomain
crypto key generate rsa general-keys exportable label MyLabel modulus 2048

Device(config)#crypto pki trustpoint MY_TRUSTPOINT
Device(ca-trustpoint)#rsakeypair MyLabel 2048
Device(ca-trustpoint)#enrollment selfsigned
Device(ca-trustpoint)#exit

Device(config)#crypto pki enroll MY_TRUSTPOINT
% Include the router serial number in the subject name? [yes/no]: no
% Include an IP address in the subject name? [no]: no
Generate Self Signed Router Certificate? [yes/no]: yes
```

This example shows how to configure COAP proxy on *ipv4 1.1.0.0 255.255.0.0* with **dtls verification trustpoint** (DTLS with certificates or verification trustpoints)

```
Device(config-coap-proxy)#security dtls ?
  id-trustpoint DTLS RSA and X.509 Trustpoint Labels
   ipv4 IP address range on which to learn lights
   ipv6 IPv6 address range on which to learn lights
   list IP address range on which to learn lights

Device(config-coap-proxy)#security dtls id-trustpoint ?
  WORD  Identity TrustPoint Label

Device(config-coap-proxy)#security dtls id-trustpoint RSA-TRUSTPOINT ?
  verification-trustpoint  Certificate Verification Label
   <cr>

Device(config-coap-proxy)#security dtls id-trustpoint RSA-TRUSTPOINT verification-trustpoint
 ?
  WORD  Identity TrustPoint Label

Device(config-coap-proxy)#security dtls id-trustpoint RSA-TRUSTPOINT verification-trustpoint
 CA-TRUSTPOINT ?
  <cr>
```

This example shows how to configure Verification Trustpoint. This is a pre-requisite for COAP **security dtls** with **verification trustpoint** configurations.

```
Device(config)#crypto pki import CA-TRUSTPOINT pkcs12 flash:hostA.p12 password cisco123
% Importing pkcs12...
```

```
Source filename [hostA.p12]?
Reading file from flash:hostA.p12
CRYPTO_PKI: Imported PKCS12 file successfully.
```

This example shows how to create a list named trial-list, to be used in the security [none | dtls] command options.

```
Device(config-coap-proxy)#list ipv4 trial_list
Device (config-coap-proxy-iplist)#1.1.0.0 255.255.255.0
Device (config-coap-proxy-iplist)#2.2.0.0 255.255.255.0
Device (config-coap-proxy-iplist)#3.3.0.0 255.255.255.0
Device (config-coap-proxy-iplist)#exit
Device (config-coap-proxy)#security none list trial_list
```

This example shows all the negation commands available in the coap-proxy sub mode.

```
Device(config-coap-proxy)#no ?
  ip-list             Configure IP-List
  max-endpoints       maximum number of endpoints supported
  port-unsecure       Specify a port number to use
  port-dtls           Specify a dtls-port number to use
  resource-discovery  Resource Discovery Server
  security            CoAP Security features
```

This example shows how you can configure multiple IPv4/IPv6 static-endpoints on the coap proxy.

```
Device (config)# coap endpoint ipv4 1.1.1.1
Device (config)# coap endpoint ipv4 2.1.1.1
Device (config)# coap endpoint ipv6 2001::1
```

This example shows how you can display the COAP protocol details.

```
Device#show coap version
CoAP version 1.0.0
RFC 7252
```

```
Device#show coap resources
Link format data =
</>
</1.1.1.6/cisco/context>
</1.1.1.6/cisco/actuator>
</1.1.1.6/cisco/sensor>
</1.1.1.6/cisco/lldp>
</1.1.1.5/cisco/context>
</1.1.1.5/cisco/actuator>
</1.1.1.5/cisco/sensor>
</1.1.1.5/cisco/lldp>
</cisco/flood>
</cisco/context>
</cisco/showtech>
</cisco/lldp>
```

```
Device#show coap globals
Coap System Timer Values :
   Discovery  : 120 sec
   Cache Exp  : 5 sec
   Keep Alive : 120 sec
   Client DB  : 60 sec
   Query Queue: 500 ms
   Ack delay  : 500 ms
   Timeout    : 5 sec

Max Endpoints      : 10
Resource Disc Mode : POST

-------------------------------------------------------------------

Device#show coap stats
Coap Stats :
Endpoints : 2
Requests : 20
Ext Queries : 0

-------------------------------------------------------------------

Device#show coap endpoints
List of all endpoints :

Code : D - Discovered , N - New
#    Status   Age(s)  LastWKC(s)  IP
-------------------------------------------------------------------------
1    D        10        94        1.1.1.6
2    D        6         34        1.1.1.5

Endpoints - Total : 2 Discovered : 2 New : 0

-------------------------------------------------------------------

Device#show coap dtls-endpoints
#     Index State  String State     Value   Port IP
---------------------------------------------------------------
1     3     SSLOK       3           48969   20.1.1.30
2     2     SSLOK       3           53430   20.1.1.31
3     4     SSLOK       3           54133   20.1.1.32
4     7     SSLOK       3           48236   20.1.1.33


-------------------------------------------------------------------
```

This example shows all options available to debug the COAP protocol.

```
Device#debug coap ?
all       Debug CoAP all
database  Debug CoAP Database
errors    Debug CoAP errors
events    Debug CoAP events
packet    Debug CoAP packet
trace     Debug CoAP Trace
warnings  Debug CoAP warnings
```

Monitoring COAP Proxy Server

To display the COAP protocol details, use the commands in the following table:

Table 1: Commands to Display to COAP specific data

show coap version	Shows the IOS COAP version and the RFC information.
show coap resources	Shows the resources of the switch and those learnt by it.
show coap endpoints	Shows the endpoints which are discovered and learnt.
show coap globals	Shows the timer values and end point values.
show coap stats	Shows the message counts for endpoints, requests and external queries.
show coap dtls-endpoints	Shows the dtls endpoint status.

Table 2: Commands to Clear COAP Commands

clear coap database	Clears the COAP learnt on the switch, and the internal database of endpoint information.

To debug the COAP protocol, use the commands in the following table:

Table 3: Commands to Debug COAP protocol

debug coap database	Debugs the COAP database output.
debug coap errors	Debugs the COAP errors output.
debug coap events	Debugs the COAP events output.
debug coap packets	Debugs the COAP packets output.
debug coap trace	Debugs the COAP traces output.
debug coap warnings	Debugs the COAP warnings output.
debug coap all	Debugs all the COAP output.

Note If you wish to disable the debugs, prepend the command with a "**no**" keyword.

Feature Information for COAP

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

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

Table 4: Feature Information for COAP

Feature Name	Releases	Feature Information
COAP	Cisco IOS XE Everest 16.5.1a	The COAP protocol is designed for use with constrained devices. COAP works in the same way on constrained devices as HTTP works on servers in accessing information.

CHAPTER 2

Configuring Auto SmartPorts

Restrictions for Configuring Auto SmartPorts

Although Auto SmartPort detects the Cisco switch it does not invoke the event trigger automatically. The event trigger needs to be manually invoked to map the switch to macros.

Information about Auto SmartPorts

Auto SmartPort macros dynamically configure ports based on the device type detected on the port. When the switch detects a new device on a port, it applies the appropriate Auto SmartPorts macro. When a link-down event occurs on the port, the switch removes the macro. For example, when you connect a Cisco IP phone to a port, Auto SmartPorts automatically applies the Cisco IP phone macro. The Cisco IP phone macro enables quality of service (QoS), security features, and a dedicated voice VLAN to ensure proper treatment of delay-sensitive voice traffic.

Auto SmartPorts uses event triggers to map devices to macros. The most common event triggers are based on Cisco Discovery Protocol (CDP) messages received from connected devices. The detection of a device (Cisco IP phone, Cisco wireless access point, or Cisco router) invokes an event trigger for that device.

Link Layer Discovery Protocol (LLDP) is used to detect devices that do not support CDP. Other mechanisms used as event triggers include the 802.1X authentication result and MAC-address learned.

System built-in event triggers exist for various devices based mostly on CDP and LLDP messages and some MAC address. These triggers are enabled as long as Auto SmartPort is enabled.

You can configure user-defined trigger groups for profiles and devices. The name of the trigger group is used to associate a user-defined macro.

Auto SmartPort Macros

The Auto SmartPort macros are groups of CLI commands. Detection of devices on a port triggers the application of the macro for the device. System built-in macros exist for various devices, and, by default, system built-in triggers are mapped to the corresponding built-in macros. You can change the mapping of built-in triggers or macros as needed.

A macro basically applies or removes a set of CLIs on an interface based on the link status. In a macro, the link status is checked. If the link is up, then a set of CLIs is applied; if the link is down, the set is removed (the no format of the CLIs are applied). The part of the macro that applies the set of CLIs is termed macro. The part that removes the CLIs (the no format of the CLIs) are termed antimacro.

When a device is connected to an Auto SmartPort, if it gets classified as a lighting end point, it invokes the event trigger **CISCO_LIGHT_EVENT**, and the macro **CISCO_LIGHT_AUTO_SMARTPORT** is executed.

Commands executed by CISCO_LIGHT_AUTO_SMARTPORT

When the macro is executed, it runs a series of commands on the switch.

The commands that are executed by running the macro **CISCO_LIGHT_AUTO_SMARTPORT** are:

- switchport mode access
- switchport port-security violation restrict
- switchport port-security mac-address sticky
- switchport port-security
- power inline port poe-ha
- storm-control broadcast level 50.00
- storm-control multicast level 50.00
- storm-control unicast level 50.00
- spanning-tree portfast
- spanning-tree bpduguard enable

Enabling Auto SmartPort

Note Auto SmartPort is disabled by default.

To disable Auto SmartPorts macros on a specific port, use the **no macro auto global processing** interface command before enabling Auto SmartPort globally.

To enable Auto SmartPort globally, use the **macro auto global processing** global configuration command.

To enable Auto SmartPorts, perform this task:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **device classifier**
4. **macro auto global processing**
5. **end**
6. **show running-config**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**device classifier** **Example:** `Device(config)# device classifier`	Enables the device classifier. Use **no device classifier** command to disable the device classifier.
Step 4	**macro auto global processing** **Example:** `Device(config)# macro auto global processing`	Enables Auto SmartPorts on the switch globally. Use **no macro auto global processing** command to disable Auto SmartPort globally.

Step 5	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Mapping Between Event Triggers and Built-in Macros

> **Note** You need to perform this task when a Cisco switch is connected to the Auto SmartPort.

To map an event trigger to a built-in macros, perform this task:

Before you begin

You need to enable auto smartport macros globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **macro auto execute** *event trigger* **builtin** *built-in macro name*
4. **macro auto trigger** *event trigger*
5. **device** *device_ID*
6. **end**
7. **show shell triggers**
8. **show running-config**
9. **copy running-config startup-config**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**macro auto execute** *event trigger* **builtin** *built-in macro name* **Example:** `Switch(config)# macro auto execute CISCO_SWITCH_EVENT builtin CISCO_SWITCH_AUTO_SMARTPORT`	Specifies a user-defined event trigger and a macro name. This action configures mapping from an event trigger to a built-in Auto Smartports macro.
Step 4	**macro auto trigger** *event trigger* **Example:** `Switch(config)# macro auto trigger CISCO_SWITCH_EVENT`	Invokes the user-defined event trigger.
Step 5	**device** *device_ID* **Example:** `Switch(config)# device cisco WS-C3560CX-8PT-S`	Matches the event trigger to the device identifier.

Step 6	**end**
	Example:
	`Switch(config)# end`
		Returns to privileged EXEC mode.

Step 7	**show shell triggers** **Example:** `Switch# show shell triggers`	Displays the event triggers on the switch.
Step 8	**show running-config** **Example:** `Switch# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Switch# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example: Enabling Auto SmartPorts

This example shows how you can enable to Auto SmartPort.

```
Device> enable
Device# configure terminal
Device(config)# device classifier
Device(config)# macro auto global processing
Device(config)# end
```

Example: Configuring Mapping Between Event Triggers and Built-in Macros

This example shows how you can configure mapping between event triggers and built-in macros.

```
Switch> enable
Switch# configure terminal
Switch(config)# macro auto execute CISCO_SWITCH_EVENT builtin CISCO_SWITCH_AUTO_SMARTPORT
Switch(config)# macro auto trigger CISCO_SWITCH_EVENT
Switch(config)# device cisco WS-C3560CX-8PT-S
Switch(config)# end
```

Feature Information for Auto SmartPorts

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

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

Table 5: Feature Information for Auto SmartPorts

Feature Name	Releases	Feature Information
Auto SmartPorts	Cisco IOS XE Everest 16.5.1a	Auto SmartPort macros dynamically configure ports based on the device type detected on the port. When the switch detects a new device on a port, it applies the appropriate Auto SmartPorts macro.

CHAPTER 3

Configuring 2-event Classification

Restrictions for 2-event classification

The following restrictions apply to 2-event classification:

- Configuration of 2-event classification has to be done before physically connecting any endpoint. Alternatively do a manual shut/no-shut of the ports drawing power.
- Power to the ports will be interrupted in case of MCU firmware upgrade and ports will be back up immediately after the upgrade.

Information about 2-event Classification

When a class 4 device gets detected, IOS allocates 30W without any CDP or LLDP negotiation. This means that even before the link comes up the class 4 power device gets 30W.

Also, on the hardware level the PSE does a 2-event classification which allows a class 4 PD to detect PSE capability of providing 30W from hardware, register itself and it can move up to PoE+ level without waiting for any CDP/LLDP packet exchange.

Once 2-event is enabled on a port, you need to manually shut/un-shut the port or connect the PD again to start the IEEE detection again. Power budget allocation for a class-4 device will be 30W if 2-event classification is enabled on the port, else it will be 15.4W.

Configuring 2-event Classification

To configure the switch for a 2-event Classification, perform the steps given below:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **power inline port 2-event**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the physical port to be configured, and enters interface configuration mode.
Step 4	**power inline port 2-event** **Example:** `Device(config-if)# power inline port 2-event`	Configures 2-event classification on the switch.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Returns to privileged EXEC mode.

Example: Configuring 2-Event Classification

This example shows how you can configure 2-event classification.

```
Device> enable
Device# configure terminal
Device(config)#  interface gigabitethernet2/0/1
Device(config-if)# power inline port 2-event
Device(config-if)# end
```

Feature Information for 2-event Classification

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

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

Table 6: Feature Information for 2-event Classification

Feature Name	Releases	Feature Information
2-event classification	Cisco IOS XE Everest 16.5.1a	When a class 4 device gets detected, IOS allocates 30W without any CDP or LLDP negotiation. This means that even before the link comes up the class 4 power device gets 30W.

CHAPTER 4

Configuring Perpetual PoE and Fast POE

Restrictions for Perpetual and Fast PoE

The following restrictions apply to perpetual and fast PoE :

- Configuration of Fast PoE or Perpetual PoE has to be done before physically connecting any endpoint. Alternatively do a manual shut/no-shut of the ports drawing power.
- Power to the ports will be interrupted in case of MCU firmware upgrade and ports will be back up immediately after the upgrade.
- When the switches are power-stacked, perpetual and fast PoE functionalities may not work as expected. This is due to power budget shortage.
- The CREE light powered device (PD) may flap at regular intervals if not configured with IP assigned from the DHCP server.
- If the PD doesn’t support LLDP user can configure with either static or 2-event to receive required power as per the PD specification.

Perpetual POE

The Perpetual POE provides uninterrupted power to connected powered device (PD) even when the power sourcing equipment (PSE) switch is booting.

Note Power to the ports will be interrupted in case of MCU firmware upgrade and ports will be back up immediately after the upgrade.

Fast POE

This feature remembers the last power drawn from a particular PSE port and switches on power the moment AC power is plugged in (within 15 to 20 seconds of switching on power) without waiting for IOS to boot up. When **poe-ha** is enabled on a particular port, the switch on a recovery after power failure, provides power to the connected endpoint devices within short duration before even the IOS forwarding starts up.

Note In case of UPOE, even though Fast POE is available on the switch side, the PD endpoints may not be able to take advantage of the same, due to the reliance on LLDP to signal the UPOE power availability. This reliance on LLDP requires that the PD endpoint still needs to wait till the IOS comes up and LLDP packet exchanges can happen, signaling the availability of UPOE power.

Configuring Perpetual and Fast POE

To configure perpetual and Fast PoE, perform the following steps:

Note You will need to configure the **perpetual-poe-ha** command before connecting the PD, or you will need to manually shut/unshut the port after configuring **poe-ha**.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-id*
4. **power inline port perpetual-poe-ha**
5. **power inline port poe-ha**
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 2/0/1`	Specifies the physical port to be configured, and enters interface configuration mode.
Step 4	**power inline port perpetual-poe-ha** **Example:** `Device(config-if)# power inline port perpetual-poe-ha`	Configures perpetual PoE. When you configure perpetual PoE on a port connected to a PD device, the PD device remains powered on during reload.
Step 5	**power inline port poe-ha** **Example:** `Device(config-if)# power inline port poe-ha`	Configures Fast PoE. When you configure Fast PoE, if switch is power cycled, PD device is powered on within 10-15 seconds of plugging into a power source without waiting for IOS to boot up. **Note** You should configure perpetual PoE using **power inline port perpetual-poe-ha** command before configuring Fast PoE using **power inline port poe-ha** command.

Step 6	**end**
	Example:
	`Device(config-if)# end`
		Returns to privileged EXEC mode.

Example: Configuring Perpetual and Fast POE

This example shows how you can configure perpetual PoE on the switch.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# power inline port perpetual-poe-ha
Device(config-if)# end
```

This example shows how you can configure fast PoE on the switch.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# power inline port perpetual-poe-ha
Device(config-if)# power inline port poe-ha
Device(config-if)# end
```

This example shows what happens if you configure fast PoE before configuring perpetual PoE.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# power inline port poe-ha
```

```
Interface Gi2/0/1:INFO: Please execute "power inline port perpetual-poe-ha"
 configuration command when "power inline port poe-ha" is configured on
the interface to enable fast poe
Device(config-if)# power inline port perpetual-poe-ha
Device(config-if)# end
```

This example shows what happens when you disable perpetual PoE without disabling fast PoE on the interface.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# power inline port poe-ha
Interface Gi2/0/1:INFO: Please execute "power inline port perpetual-poe-ha"
 configuration command when "power inline port poe-ha" is configured on
the interface to enable fast poe
Device(config-if)# power inline port perpetual-poe-ha
Device(config-if)# no power inline port poe-ha
Device(config-if)# power inline port poe-ha
Device(config-if)# no power inline port perpetual-poe-ha
Interface Gi2/0/1:INFO: Please execute "no power inline port poe-ha"
configuration command, as fast poe has no effect without "power inline
port perpetual-poe-ha" configuration on the interface
Device(config-if)# end
```

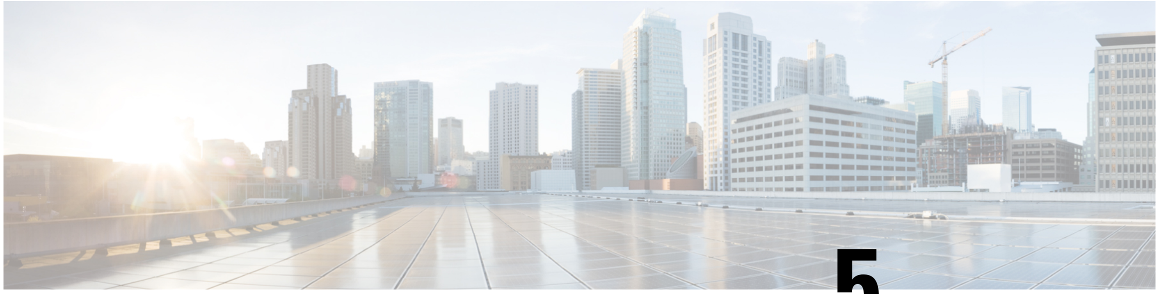
Feature Information for Persistent and Fast PoE

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

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

Table 7: Feature Information for Persistent and Fast PoE

Feature Name	Releases	Feature Information
Perpetual and Fast PoE	Cisco IOS XE Everest 16.5.1a	The Perpetual POE provides uninterrupted power to connected PD device even when the PSE switch is booting. Fast PoE remembers the last power drawn from a particular PSE port and switches on power without waiting for IOS to boot up.



CHAPTER 5

Frequently Asked Questions

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

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

Frequently Asked Questions

This section list some frequently asked questions about Network Powered Lighting.

- **Question:**

  What does "New Endpoint" in the "show coap stats" output mean? When does "New Endpoint" migrate to "Endpoint"?

  Answer:

  New endpoint means that an endpoint has been seen (Discovery packets received) but not yet registered by the CoAP proxy. The CoAP proxy will periodically look at the new endpoint and then send them a GET on “./well-known/core” to get more details, and once RSP is received, it is moved to “Endpoint".

- **Question:**

  Why can I not do a "CoAP start” unless there is a security configuration?

  Answer:

  We need to ensure that all configurations related to CoAP are done and then it can be explicitly enabled. This avoids any intermittent unstable states across configurations.

- **Question:**

Why do we need to enforce drop into the “coap proxy” configuration mode “coap proxy <cr>”? When I have completed the configuration, I have to exit twice to get back to the switch prompt. I do not find this very user friendly.

Answer:

We would alternatively have to type “coap proxy” as prefix for each configuration that we do. It is a better option to get into a sub-mode, as all the configurations under the sub-mode relating to coap-proxy can be done.

- **Question:**

  Why am I not able to unconfigure security or other parameters without first stopping the coap process?

  Answer:

  We need to ensure that all configurations related to CoAP are done and then it can be explicitly enabled. This also avoids and controls the complexity where the user might configure settings on the fly, when CoAP is enabled.

- **Question:**

  When I stop coap, all configurations associated with the CoAP process are not removed automatically (or return to defaults). Why does the CoAP remember previous configuration? This seems very hard for users to start fresh.

  Answer:

  The system has been intentionally designed this way and this is expected behavior. Sometimes we just want to make minor changes, like change max-endpoints and re-start the proxy. It is a better option to hold all other configurations in place, else the user has to configure everything all over again.

- **Question:**

  How can I see what the security configurations have been set?

  Answer:

  The command “show run” shows all the configurations.

- **Question:**

  How can I tune the timer values?

  ```
  Example:
  Device#sho coap glo
  Coap System Timer Values:
  Discovery : 120 sec
  Cache Exp : 5 sec
  Keep Alive : 120 sec
  Client DB : 5 sec
  Query Queue : 500 ms
  Ack delay : 500 ms
  Timeout : 5 sec
  Max Endpoints : 500
  Resource Disc Mode : POST
  ```

  Answer:

  The timer values are fixed and are not tunable at the moment. The reason for this is to avoid inconsistency across systems.

- **Question:**

  What are the commands “list” and “endpoint” used for?

  Answer:

  The “list” command is to make it easier to configure multiple ip-addresses and give a name to it. Then you can assign the name instead of a single ip, to represent multiple ip’s. The "endpoint" command is used to configure a static end point, in cases where the endpoints do not advertise themselves.

- **Question:**

  How can I find the endpoint-to-port mapping by using the “show” command?

  Answer:

  We do not support that of now. However, other commands can be run to fetch this data. Currently, we can still get all the details mentioned using individual commands like “lldp neighbours”, “ip dhcp”, “power inlines” and so on.