Configuring the Cisco C-SM-16P4M2X or C-SM-40P8M2X EtherSwitch Service Module

First Published: 2019-06-25

Last Modified: 2020-07-23

Overview of the

Cisco C-SM-16P4M2X or C-SM-40P4M2X is a layer-2 switch module that brings high-density Small Form-Factor Pluggable (SFP) /Small Form-Factor Pluggable Plus (SFP+), 1 Gigabit, 2.5 mGiG, and 10G connectivity to the Cisco 4000 Series Integrated Services Routers (ISRs). It also, provides 10G-capable internal uplink to central forwarding data plane on modular ISR platforms.

The C-SM-16P4M2X or C-SM-40P4M2X service module is capable of supporting standard Power over Ethernet (PoE), Power over Ethernet Plus (PoE+), Cisco Enhanced Power over Ethernet (EPoE), and Cisco Universal Power over Ethernet (UPoE) on all copper ports. A maximum of 60 watts of power for each copper port is supported by leveraging both signal and spare pairs.

This guide describes how to configure the C-SM-16P4M2X or C-SM-40P4M2X service module in the Cisco Catalyst 8300 Series Edge Platforms.

The following is the feature history for the SM-X-16G4M2X or SM-X-40G8M2X service module:

Table 1: Feature History for C-SM-16P4M2X or C-SM-40P4M2X a

Release	Modification
Cisco IOS XE Amsterdam 17.3.2	Cisco C-SM-16P4M2X and C-SM-40P4M2X Service Modules were introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. 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.

Configuring the Cisco C-SM-16P4M2X or C-SM-40P4M2X Service Module

This section describes how to configure the Cisco C-SM-16P4M2X or C-SM-40P4M2X service module features and some important concepts about the Cisco C-SM-16P4M2X or C-SM-40P4M2X service module.

Prerequisites for the Cisco C-SM-16P4M2X or C-SM-40P4M2X Service Module

Cisc IOS XE Amsterdam 17.3.2 release is required to configure the Cisco C-SM-16P4M2X or C-SM-40P4M2X.

To determine the version of Cisco IOS software that is running on your router, log in to the router and enter the **show version** command:

```
Router> show version

Cisco IOS XE Software, Version 17.03.01prd8

Cisco IOS XE Software, Version 17.03.01prd8
Cisco IOS Software [Amsterdam], c8000be Software (X86_64_LINUX_IOSD-UNIVERSALK9-M), Version
17.3.1prd8, RELEASE SOFTWARE

Technical Support: http://www.cisco.com/techsupport

Copyright (c) 1986-2020 by Cisco Systems, Inc.

Compiled Tue 19-May-20 12:00 by mcpre
```

- To view the router (Cisco Catalyst 8300 Series Edge Platforms), Cisco IOS software release, and feature set, enter the **show version** command in privileged EXEC mode.
- To view the Cisco IOS Release number mapping, see Release Notes for the Cisco Catalyst 8300 Series Edge Platforms.

Configuring Power Over Ethernet

Before you begin

Each copper port on the SM-X-16G4M2X service modue can auto detect one of following connected devices, and supply power to them properly:

- An IEEE 802.3af and IEEE 802.3at compliant power device
- Cisco EPOE and UPOE power device

To configure power over ethernet, use these commads:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** interface id **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the physical port to be configured, and enters interface configuration mode.

	Command or Action	Purpose
Step 3	**power inline** [**auto** | **max max-wattage**] *never* **Example:** `router(config-if)# power inline auto`	Configures the PoE mode on the port. The keywords have these meanings: • Auto—Enables powered-device detection. If enough power is available, automatically allocates power to the PoE port after device detection. This is the default setting. • Max max-wattage—Limits the power allowed on the port. The range for PoE+ ports is 4000 to 60000 mW. The range for Cisco UPOE ports is 4000 to 60000 mW. If no value is specified, the maximum is allowed. • Never —Disables device detection, and disable power to the port. **Note** If a port has a Cisco powered device connected to it, do not use the power inline never command to configure the port. A false link-up can occur, placing the port into the error-disabled state.
Step 4	**end** **Example:** `router(config-if)# end`	Returns to privileged EXEC mode.

Verifying the Power Over Ethernet

To verify the power over ethernet configuration, user the **show power inline** command as shown in the following example.

```
Router#show power inline
Available:500.0(w)  Used:100.3(w)  Remaining:399.8(w)

Interface Admin  Oper       Power   Device              Class Max
                            (Watts)
--------- ------ ---------- ------- ------------------- ----- ----
Gi2/0/0   auto   on         30.0    AIR-AP3802I-H-K9    4     60.0
Gi2/0/1   auto   on         10.3    IP Phone 7970       3     60.0
Gi2/0/2   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/3   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/4   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/5   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/6   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/7   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/8   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/9   auto   off        0.0     n/a                 n/a   60.0
Gi2/0/10  auto   off        0.0     n/a                 n/a   60.0
Gi2/0/11  auto   off        0.0     n/a                 n/a   60.0
Gi2/0/12  auto   off        0.0     n/a                 n/a   60.0
```

```
Gi2/0/13  auto   off        0.0     n/a                n/a   60.0
Gi2/0/14  auto   off        0.0     n/a                n/a   60.0
Gi2/0/15  auto   off        0.0     n/a                n/a   60.0
Tw2/0/16  auto   off        0.0     n/a                n/a   60.0
Tw2/0/17  auto   on         30.0    AIR-AP3802I-H-K9   4     60.0
Tw2/0/18  auto   off        0.0     n/a                n/a   60.0
Tw2/0/19  auto   on         30.0    AIR-AP3802I-H-K9   4     60.0
```

Configuring Universal PoE

Cisco UPOE can provide a maximum of 60Watts power over both signal and spare paris of RJ45 cable. UPOE capable switch port can enable spare pair and supply power to it through CDP or LLDP negotiations with UPOE power device automatically.

If end-point power device is capable to consume power on both signal and spare pairs but without corresponding CDP/LLDP negotiation mechanism available, following configurations can be used to manually force four-pair on specific port.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** interface id **Example:** `Device(config)# interface gigabitethernet2/0/1`	Specifies the physical port to be configured, and enters interface configuration mode.
Step 3	**power inline four-pair forced** **Example:** `router(config-if)# power four-pair forced`	Forces power enabling on both signal and spare pairs from a switch port.
Step 4	**end** **Example:** `router(config-if)# end`	Returns to privileged EXEC mode.

Configuring Gigabit Ethernet Interfaces

To configure speed and duplex operation, follow these steps in interface configuration mode:

Before you begin

The GigabitEthernet interface can be either manually configured as 10Mbps, 100Mbps or 1Gbps mode, or auto-negotiated to proper working mode with link peer.

Procedure

	Command or Action	Purpose
Step 1	**duplex** [**full**|**auto**] **Example:** `router(config-if)# duplex full`	• Auto—Autonegotiates duplex mode with peer. • Half—Forces duplex mode to half. Half mode is supported only for 10Mbps mode. • Full—Forces duplex mode to full.
Step 2	**speed** [**10**|**100**|**1000**|**auto**] **Example:** `router(config-if)# speed auto`	• 10/100/1000—Forces speed to 10/100/1000 Mbps. • Auto—Autonegotiates the speed with the peer.

Configuring Two-Gigabit Ethernet Interfaces

To configure mGig, follow these steps in interface configuration mode:

Before you begin

The mGiG ethernet interface can be manually configured as 100Mbps, 1Gbps or 2.5Gbps mode, or auto-negotiated with peer link over the commonly used cat5e cable or higher cable variants.

Procedure

	Command or Action	Purpose
Step 1	**duplex** [**full**|**auto**] **Example:** `router(config-if)# duplex auto`	• Auto—Autonegotiates duplex mode with peer. • Full—Forces duplex mode to full.
Step 2	**speed** [**100**|**1000**|**2500**|**auto**] **Example:** `router(config-if)# speed auto`	• Auto—Autonegotiates speed with the peer. • 100|1000|2500|—Sets the speed to 100/1000/2500 Mbps.

Configuring Ten-Gigabit Ethernet Interfaces

You cannot configure the duplex and speed on the Ten-Gigabit ethernet interface. Its speed depends on the type of SFP or SFP+ inserted into the port.

Configuring Flowcontrol and Maximum Transmission Unit

Flow control allows congested port to pause traffic at the peer node. If one port experiences congestion on egress direction, it notifies other ports using pause frames to stop transferring packets to it during congestion period.

Note Cisco SM-X-16G4M2X switch ports support only receive direction flow control, which are aligned with other Catalyst switches.

The default maximum transmission unit (MTU) size for frames received and sent on all switch interfaces is 1500 bytes. You can change the MTU size to support jumbo frames on all external interfaces.

Procedure

	Command or Action	Purpose
Step 1	**flowcontrol receive** [**on** | **off**] **Example:** `router(config-if)# flowcontrol receive on`	The default state is off. • On—Enables receiving/handling the pause frames from a peer device. • Off—Disables receiving/handling the pause frames from a peera
Step 2	**mtu** *mtu size* **Example:** `router(config-if)# mtu 9000`	Sets the maximum transmission unit (MTU) size for a frame. The range from 1500 to 9216.

Verifying the Ethernet Interface Status

To view the status of the Gigabit interface, use the **show interfaces GigabitEthernet** command.

```
Router#show interfaces gigabitEthernet 2/0/14
GigabitEthernet2/0/14 is up, line protocol is up (connected)
  Hardware is SM-X-16G4M2X, address is f4db.e673.fa15 (bia f4db.e673.fa15)
  MTU 3000 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
  Full-duplex, 1000Mb/s, link type is auto, media type is 10/100/1000BaseTX
  input flow-control is on, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     258911616529 packets input, 33140686915712 bytes, 0 no buffer
     Received 0 broadcasts (0 multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     0 input packets with dribble condition detected
     258846666089 packets output, 33132365295921 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

To view the status of the mGig interface, use the **show interfaces twoGigabitEthernet** command.

```
Router# show int twoGigabitEthernet 2/0/16
TwoGigabitEthernet2/0/16 is up, line protocol is up (connected)
  Hardware is SM-X-16G4M2X, address is f4db.e673.fa17 (bia f4db.e673.fa17)
  MTU 1500 bytes, BW 2500000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
 Full-duplex, 2500Mb/s, link type is force-up, media type is 100/1000/2.5GBaseTX
input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:01, output 00:00:05, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     172 packets input, 41736 bytes, 0 no buffer
     Received 0 broadcasts (172 multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 172 multicast, 0 pause input
     0 input packets with dribble condition detected
     165 packets output, 42501 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

To view the status of the ten GigabitEthernet, use the **show interfaces tenGigabitEthernet** command.

```
Router# show int tenGigabitEthernet 2/0/20
TenGigabitEthernet2/0/20 is up, line protocol is up (connected)
  Hardware is SM-X-16G4M2X, address is f4db.e673.fa1b (bia f4db.e673.fa1b)
  MTU 1500 bytes, BW 10000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
Full-duplex, 10Gb/s, link type is auto, media type is SFP-10Gbase-SR
input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2611024549517 packets input, 334211146017180 bytes, 0 no buffer
     Received 0 broadcasts (0 multicasts)
     0 runts, 28737 giants, 0 throttles
     28738 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     0 input packets with dribble condition detected
     2591035043779 packets output, 331652477689500 bytes, 0 underruns
     0 output errors, 0 collisions, 2 interface resets
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
```

MAC Table Manipulation

This section includes the following:

Creating a Static Entry in the MAC Address Table

Perform the following task to create a static entry in the MAC address table.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static mac-address vlan** *vlan-id* **interface** *Interface-id* **Example:** `Router(config)# mac address-table static 00ff.ff0d.2dc0 vlan 1 interface gigabitethernet 0/1/0`	Creates a static entry in the MAC address table.
Step 4	end **Example:** `Router(config)# end`	Returns to privileged EXEC mode.
Step 5	show mac address-table **Example:** `Router# show mac address-table`	Verifies the MAC address table.

MAC Address-Based Traffic Blocking

Perform the following task to block all traffic to or from a MAC address in a specified VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router#configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static mac-address vlan** *vlan-id* **drop** **Example:** `Router(config)# mac address-table static 00ff.ff0d.2dc0 vlan 1 drop`	Creates a static entry with drop action in the MAC address table.
Step 4	end **Example:** `Router(config)# end`	Returns to privileged EXEC mode.
Step 5	show mac address-table **Example:** `Router# show mac address-table`	Verifies the MAC address table.

Configuring and Verifying the Aging Timer

Perform this task to configure the aging timer.

Procedure

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

	Command or Action	Purpose

Step 3	mac address-table aging-time time
	Example:
	`Router(config)# mac address-table aging-time 600`
	or
	Example:
	`Router(config)# mac address-table aging-time 0`
		Configures the MAC address aging timer age in seconds.
		• The accept value is either 0 or 10-1000000 seconds. Default value is 300 seconds.
		• The maximum aging timer supported by switch chipset is 634 seconds. If configure greater than 634 seconds, MAC address will age out after 634 seconds.
		• The value 0 means dynamic MAC entries will never age out.

Step 4	end **Example:** `Router(config)# end`	Returns to privileged EXEC mode.
Step 5	show mac address-table aging-time **Example:** `Router# show mac address-table aging-time`	Verifies the MAC address table.

MAC Learning on a Vlan

To disable or enable MAC learning on specified vlan, perform these steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	mac address-table learning vlan *vlan-id* **Example:** `Router(config)#mac address-table learning vlan 10`	By default, mac learning is enabled on each vlan. .

	Command or Action	Purpose
Step 4	end **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Software Features

The following are the software features supported on the Cisco SM-X-16G4M2X or SM-X-40G8M2X service module:

Assigning IP Addresses to Switch Virtual Interfaces

To configure IP routing, you need to assign IP addresses to Layer 3 network interfaces. This enables communication with the hosts on those interfaces that use IP. IP routing is disabled by default, and no IP addresses are assigned to Switch Virtual Interfaces (SVIs).

An IP address identifies a destination for IP packets. Some IP addresses are reserved for special uses and cannot be used for host, subnet, or network addresses. RFC 1166, “Internet Numbers,” contains the official description of these IP addresses.

An interface can have one primary IP address. A a subnet mask identifies the bits that denote the network number in an IP address.

Beginning in privileged EXEC mode, follow these steps to assign an IP address and a network mask to an SVI.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**interface vlan** *vlan_id*	Enter interface configuration mode, and specify the Layer 3 VLAN to configure.
Step 3	**ip address** *ip-address subnet-mask*	Configure the IP address and IP subnet mask.
Step 4	**end**	Return to privileged EXEC mode.
Step 5	**show interfaces** [*interface-id*] **show ip interface** [*interface-id*] **show running-config interface** [*interface-id*]	Verify your entries.
Step 6	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

SVI Supported Features

The following table provided the supported features on the SVI.

Table 2: SVI Supported Features

Techolongy	Feature	Use Case
Routing	Routing Protocol	Interconnects Layer 3 networks using protocols such as Routing Information Protocol (RIP), Open Shortest Path First (OSPF) Protocol, and Enhanced Interior Gateway Routing Protocol (EIGRP) configured under SVI. For more informaton on routing protocol, see the IP Routing: Protocol-Independent Configuration Guide.
	Hot Standby Router Protocol (HSRP)	Supports redundancy and high availability with a secondary device connected to the LAN with SVI, using HSRP. For more informaton on HSRP, see the *First Hop Redundancy Protocols Configuration Guide*.
	DHCP	Cisco devices running Cisco software include Dynamic Host Configuration Protocol (DHCP) server and the relay agent software. The Cisco IOS DHCP server is a full DHCP server implementation that assigns and manages IP addresses from specified address pools within the device to DHCP clients. The DHCP server can be configured to assign additional parameters such as the IP address of the Domain Name System (DNS) server and the default device. For more informaton on HSRP, see the, IP Addressing: DHCP Configuration Guide
	Multicast (IPv4)	Provides multicast support for clients connected to the switch ports. For more informaton on HSRP, see the, IP Multicast: PIM Configuration Guide

Techolongy	Feature	Use Case
	VRF	Associates a VRF instance with an SVI to map VLANs to different logical or physical VPN WAN connections. For more informaton on VRF protocol, see the IP Routing: Protocol-Independent Configuration Guide.
Security	ACL	Provides packet filtering to control network traffic and restrict the access of users and devices to the network For more informaton on ACL protocol, see the Security Configuration Guide: Access Control Lists.
	NAT	Provides NAT under SVI. For more information on NAT, see the IP Addressing: NAT Configuration Guide.
Qos	Classification with standard and extended access list	Provides QoS classification with standard and extended access lists. For more informtion on QoS, see the Security Configuration Guide: Access Control Lists.
	Class-based marking	Provides QoS marking based on user-defined traffic class with DSCP and IP precedence values. For more information on QoS Marking, see the QoS: Classification Configuration Guide.
	Policing	Limits the input or output transmission rate on SVI and specifies traffic handling policies when the traffic either conforms to or exceeds the specified rate limits. For more informtion on Policing, see the QoS: Policing and Shaping Configuration Guide

Techolongy	Feature	Use Case
Bridging	EVC under SVI	Supports a default encapsulation EFP under SVI, to have VLAN/BD integrated.
	EVC with MAC ACL under SVI	For more information on EVC, see the https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/cether/configuration/xe-3s/asr903/16-11-1/b-ce-layer2-xe-xe-16-11-asr900/b-ce-layer2-xe-xe-16-11-asr900_chapter_011.html

IEEE 802.1x Protocol

The IEEE 802.1x standard defines a client/server-based access control and authentication protocol that prevents clients from connecting to a LAN through publicly accessible ports unless they are authenticated. The authentication server authenticates each client connected to a port before making available any services offered by the router or the LAN.

Until the client is authenticated, IEEE 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol (CDP), and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication, normal traffic can pass through the port. For more information on IEEE 802.1x port-based authentication, see the Configuring IEEE 802.1x Port-Based Authentication chapter of the *Security Configuration Guide, Cisco IOS XE Gibraltar 16.10.x*.

Configuring IEEE 802.1X Port-Based Authentication

IEEE 802.1X port-based authentication is configured on a device to prevent unauthorized devices (supplicants) from gaining access to the network. The device can combine the function of a router, switch, and access point, depending on the fixed configuration or installed modules. The switch functions are provided by either built-in switch ports or a plug-in module with switch ports. This feature supports both access ports and trunk ports. For more informaton on 802.1X port-based authentication, see the Configuring IEEE 802.1X Port-Based Authentication Guide.

.

Enabling AAA Authorization for VLAN Assignment

AAA authorization limits the services available to a user. When AAA authorization is enabled, the device uses information retrieved from the user's profile, which is in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

Procedure

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authorization network radius if-authenticated** **Example:** `Device(config)# aaa authorization network radius if-authenticated`	Configures the device for user RADIUS authorization for all network-related service requests. RADIUS authorization succeeds if the user has authenticated.
Step 5	**aaa authorization exec radius if-authenticated** **Example:** `Device(config)# aaa authorization exec radius if-authenticated`	Configures the device for user RADIUS authorization if the user has privileged EXEC access. RADIUS authorization succeeds if the user has authenticated.
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Enabling IEEE 802.1X Authentication and Authorization

Procedure

	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	**aaa authentication dot1x** {**default** | *listname*} *method1* [*method2...*] **Example:**	Creates a series of authentication methods that are used to determine user privilege to access the privileged command level so that the device can communicate with the AAA server.

	Command or Action	Purpose
	`Device(config)# aaa authentication dot1x default group radius`	
Step 4	**dot1x system-auth-control** **Example:** `Device(config)# dot1x system-auth-control`	Globally enables 802.1X port-based authentication.
Step 5	**identity profile default** **Example:** `Device(config)# identity profile default`	Creates an identity profile and enters dot1x profile configuration mode.
Step 6	**exit** **Example:** `Device(config-identity-prof)# exit`	Exits dot1x profile configuration mode and returns to global configuration mode.
Step 7	**interface** *type slot/port* **Example:** `Device(config)# interface Gigabitethernet 1/0/1`	Enters interface configuration mode and specifies the interface to be enabled for 802.1X authentication.
Step 8	**access-session port-control** {**auto** | **force-authorized** | **force-unauthorized**} **Example:** `Device(config-if)# access-session port-control auto`	Enables 802.1X port-based authentication on the interface. • **auto**—Enables IEEE 802.1X authentication and causes the port to begin in the unauthorized state, allowing only EAPOL frames to be sent and received through the port. The authentication process begins when the link state of the port changes from down to up or when an EAPOL-start frame is received. The Device requests the identity of the supplicant and begins relaying authentication messages between the supplicant and the authentication server. Each supplicant attempting to access the network is uniquely identified by the Device by using the supplicant MAC address. • **force-authorized**-—Disables IEEE 802.1X authentication and causes the port to change to the authorized state without any authentication exchange required. The port sends and receives normal traffic without IEEE 802.1X-based authentication of the client. This is the default setting.

	Command or Action	Purpose
		• **force-unauthorized**—Causes the port to remain in the unauthorized state, ignoring all attempts by the supplicant to authenticate. The Device cannot provide authentication services to the supplicant through the port.
Step 9	**dot1x pae** [**supplicant** | **authenticator** | **both**] **Example:** `Device(config-if)# dot1x pae authenticator`	Sets the Port Access Entity (PAE) type. • **supplicant**—The interface acts only as a supplicant and does not respond to messages that are meant for an authenticator. • **authenticator-**—The interface acts only as an authenticator and does not respond to any messages meant for a supplicant. • **both**—The interface behaves both as a supplicant and as an authenticator and thus does respond to all dot1x messages.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 11	**show dot1x** **Example:** `Device# show dot1x`	Displays whether 802.1X authentication has been configured on the device.

IGMP Snooping for IPv4

IGMP snooping allows switches to examine IGMP packets and make forwarding decisions based on their content. You can configure the switch to use IGMP snooping in subnets that receive IGMP queries from either IGMP or the IGMP snooping querier. IGMP snooping constrains IPv4 multicast traffic at Layer 2 by configuring Layer 2 LAN ports dynamically to forward IPv4 multicast traffic only to those ports that want to receive it.

Layer 2 switches can use IGMP snooping to constrain the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast devices. As the name implies, IGMP snooping requires the LAN switch to snoop on the IGMP transmissions between the host and the router and to keep track of multicast groups and member ports. When the switch receives an IGMP report from a host for a particular multicast group, the switch adds the host port number to the forwarding table entry; when it receives an IGMP Leave Group message from a host, it removes the host port from the table entry. It also periodically deletes entries if it does not receive IGMP membership reports from the multicast clients. For more information on this feature, see https://www.cisco.com/c/en/us/td/docs/routers/7600/ios/15S/configuration/guide/7600_15_0s_book/snooigmp.html .

MLD Snooping

In IP Version 4 (IPv4), Layer 2 switches can use Internet Group Management Protocol (IGMP) snooping to limit the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast devices. In IPv6, MLD snooping performs a similar function. With MLD snooping, IPv6 multicast data is selectively forwarded to a list of ports that want to receive the data, instead of being flooded to all ports in a VLAN. This list is constructed by snooping IPv6 multicast control packets.

MLD is a protocol used by IPv6 multicast routers to discover the presence of multicast listeners (nodes wishing to receive IPv6 multicast packets) on the links that are directly attached to the routers and to discover which multicast packets are of interest to neighboring nodes. MLD is derived from IGMP; MLD Version 1 (MLDv1) is equivalent to IGMPv2, and MLD Version 2 (MLDv2) is equivalent to IGMPv3. MLD is a subprotocol of Internet Control Message Protocol Version 6 (ICMPv6), and MLD messages are a subset of ICMPv6 messages, identified in IPv6 packets by a preceding Next Header value of 58.

MLD Snooping Configuration Guidelines

When configuring MLD snooping, consider these guidelines:

- You can configure MLD snooping characteristics at any time, but you must globally enable MLD snooping by using the **ipv6 mld snooping** global configuration command for the configuration to take effect.
- MLD snooping and IGMP snooping act independently of each other. You can enable both features at the same time on the switch.

Default MLD Snooping Configuration

Table 3: Default MLD Snooping Configuration

Feature	Default Setting
MLD snooping (Global)	Disabled.
MLD snooping (per VLAN)	Enabled. MLD snooping must be globally enabled for VLAN MLD snooping to take place.
IPv6 Multicast addresses	None configured.
IPv6 Multicast router ports	None configured.
MLD snooping Immediate Leave	Disabled.
MLD snooping robustness variable	Global: 2; Per VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global count.
Last listener query count	Global: 2; Per VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global count.

Feature	Default Setting
Last listener query interval	Global: 1000 (1 second); VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global interval.
TCN query solicit	Disabled.
TCN query count	2.
MLD listener suppression	

Enabling or Disabling MLD Snooping on a VLAN

To enable MLD snooping on a VLAN, perform this procedure:

Procedure

	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	**ipv6 mld snooping** **Example:** `Device(config)# ipv6 mld snooping`	Enables MLD snooping on the switch.
Step 4	**ipv6 mld snooping vlan** *vlan-id* **Example:** `Device(config)# ipv6 mld snooping vlan 1`	Enables MLD snooping on the VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094. **Note** MLD snooping must be globally enabled for VLAN snooping to be enabled.
Step 5	**end** **Example:** `Device(config)# ipv6 mld snooping vlan 1`	Returns to privileged EXEC mode.

Configuring UniDirectional Link Detection

UniDirectional Link Detection (UDLD) is a Layer 2 protocol that enables devices connected through fiber-optic or twisted-pair Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. All connected devices must support UDLD for the protocol to successfully identify and disable unidirectional links. When UDLD detects a unidirectional link, it disables the affected port and alerts you. Unidirectional links can cause a variety of problems, including spanning-tree topology loops.

Enabling UDLD Globally

Follow these steps to enable UDLD in the aggressive or normal mode and to set the configurable message timer on all fiber-optic ports on the device.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**udld** {**aggressive** | **enable** | **message time** *message-timer-interval*} **Example:** `Device(config)# udld enable message time 10`	Specifies the UDLD mode of operation: • **aggressive**—Enables UDLD in aggressive mode on all fiber-optic ports. • **enable**—Enables UDLD in normal mode on all fiber-optic ports on the . UDLD is disabled by default. An individual interface configuration overrides the setting of the **udld enable** global configuration command. • **message time** *message-timer-interval*—Configures the period of time between UDLD probe messages on ports that are in the advertisement phase and are detected to be bidirectional. The range is from 1 to 90 seconds; the default value is 15. **Note** This command affects fiber-optic ports only. Use the **udld** interface configuration command to enable UDLD on other port types. Use the **no** form of this command, to disable UDLD.

	Command or Action	Purpose
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling UDLD on an Interface

Follow these steps either to enable UDLD in the aggressive or normal mode or to disable UDLD on a port.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port to be enabled for UDLD, and enters interface configuration mode.
Step 3	**udld port** [**aggressive**] **Example:** `Device(config-if)# udld port aggressive`	UDLD is disabled by default. • **udld port**—Enables UDLD in normal mode on the specified port. • **udld port aggressive**—(Optional) Enables UDLD in aggressive mode on the specified port. **Note** Use the **no udld port** interface configuration command to disable UDLD on a specified fiber-optic port.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the Switched Port Analyzer

This section describes how to configure a Switched Port Analyzer (SPAN) session on SM-X-16G4M2X or SM-X-40G8M2X service module. The following restrictions apply to the SM-X-16G4M2X or SM-X-40G8M2X service module:

- Only intra-module local SPAN is supported and cross module SPAN is not supported.
- Each SM-X-16G4M2X or SM-X-40G8M2X service module can support 66 SPAN sessions in all ports. However, only eight of them can be used as source sessions which includes local SPAN sessions and remote SPAN source sessions. The remaining sessions can be used as remote SPAN destination sessions.
- The session ID range is from 1 to 66.

Note Tx, Rx, or both Tx and Rx monitoring is supported.

SPAN and RSPAN

You can analyze network traffic passing through ports or VLANs by using SPAN or RSPAN to send a copy of the traffic to another port on the device or on another device that has been connected to a network analyzer or other monitoring or security device. SPAN copies (or mirrors) traffic received or sent (or both) on source ports or source VLANs to a destination port for analysis. SPAN does not affect the switching of network traffic on the source ports or VLANs. You must dedicate the destination port for SPAN use. Destination ports do not receive or forward traffic by default. It can receive or forward traffic when ingress-forwarding is enabled on the destination ports.

Only traffic that enters or leaves source ports or traffic that enters or leaves source VLANs can be monitored by using SPAN; traffic routed to a source VLAN cannot be monitored. For example, if incoming traffic is being monitored, traffic that gets routed from another VLAN to the source VLAN cannot be monitored; however, traffic that is received on the source VLAN and routed to another VLAN can be monitored.

You can use the SPAN or RSPAN destination port to inject traffic from a network security device. For example, if you connect a Cisco Intrusion Detection System (IDS) sensor appliance to a destination port, the IDS device can send TCP reset packets to close down the TCP session of a suspected attacker.

Creating a Local SPAN Session

Follow these steps to create a SPAN session and specify the source (monitored) ports or VLANs and the destination (monitoring) ports.

Procedure

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 1 source interface gigabitethernet1/0/1`	Specifies the SPAN session and the source port/Vlan (monitored port). • For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel** *port-channel-number*). Valid port-channel numbers are 1 to 32. • For *vlan-id*, specify the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN). **Note** A single session can include multiple sources (ports or VLANs) defined in a series of commands, but you cannot combine source ports and source VLANs in one session. • (Optional) [**,** | **-**] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **both** | **rx** | **tx**—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the source interface sends both sent and received traffic. • **both**—Monitors both received and sent traffic. • **rx**—Monitors received traffic.

	Command or Action	Purpose
		• **tx**—Monitors sent traffic. **Note** You can use the **monitor session** *session_number* **source** command multiple times to configure multiple source ports.
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | **-**] [**encapsulation {replicate | dot1q}**]} **Example:** `Device(config)# monitor session 1 destination interface gigabitethernet1/0/2 encapsulation replicate`	**Note** For local SPAN, you must use the same session number for the source and destination interfaces. • For *session_number*, specify the session number entered in step 4. • (Optional) [**,** | **-**] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged). (Optional) **encapsulation dot1q** specifies that the destination interface accepts the source interface incoming packets with IEEE 802.1Q encapsulation. **Note** You can use **monitor session** *session_number* **destination** command multiple times to configure multiple destination ports.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating a Local SPAN with Incoming Traffic Allowed on Destination

Follow these steps to create a SPAN session, to specify the source ports or VLANs and the destination ports, and to enable incoming traffic on the destination port for a network security device (such as a Cisco IDS Sensor Appliance).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 2 source gigabitethernet1/0/1 rx`	Specifies the SPAN session and the source port (monitored port).

	Command or Action	Purpose

Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | **-**] [**encapsulation replicate**] [**ingress** {**dot1q vlan** *vlan-id* | **untagged vlan** *vlan-id* | **vlan** *vlan-id*}]}
	Example:
	`Device(config)# monitor session 2 destination interface gigabitethernet1/0/2 encapsulation replicate ingress dot1q vlan 6`
		Specifies the SPAN session, the destination port, the packet encapsulation, and the ingress VLAN and encapsulation.
		• For *session_number*, specify the session number entered in Step 4.
		• For *interface-id*, specify the destination port. The destination interface must be a physical port or port-channel; it cannot be an EtherChannel, and it cannot be a VLAN.
		• (Optional) [**,** | **-**]—Specifies a series or range of interfaces. Enter a space before and after the comma or hyphen.
		• (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged).
		• (Optional) **encapsulation dot1q**specifies that the destination interface accepts the source interface incoming packets with IEEE 802.1Q encapsulation.
		• **ingress** enables forwarding of incoming traffic on the destination port and to specify the encapsulation type:
		  • **dot1q vlan** *vlan-id*—Accepts incoming packets with IEEE 802.1Q encapsulation with the specified VLAN as the default VLAN.
		  • **untagged vlan** *vlan-id* or **vlan** *vlan-id*—Accepts incoming packets with untagged encapsulation type with the specified VLAN as the default VLAN.

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

	Command or Action	Purpose
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Specifying VLANs to Filter

Follow these steps to limit SPAN source traffic to specific VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source interface** *interface-id* **Example:** `Device(config)# monitor session 2 source interface gigabitethernet1/0/2 rx`	Specifies the characteristics of the source port (monitored port) and SPAN session. • For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. The interface specified must already be configured as a trunk port.

	Command or Action	Purpose
Step 5	**monitor session** *session_number* **filter vlan** *vlan-id* [**,** | **-**] **Example:** `Device(config)# monitor session 2 filter vlan 1 - 5 , 9`	Limits the SPAN source traffic to specific VLANs. • For *session_number*, enter the session number specified in Step 4. • For *vlan-id*, the range is 1 to 4094. • (Optional) Use a comma (**,**) to specify a series of VLANs, or use a hyphen (**-**) to specify a range of VLANs. Enter a space before and after the comma; enter a space before and after the hyphen.
Step 6	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | **-**] [**encapsulation replicate** |**encapsulation dot1q]**} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet1/0/1`	Specifies the SPAN session and the destination port (monitoring port). • For *session_number*, specify the session number entered in Step 4. • For *interface-id*, specify the destination port. The destination interface must be a physical port or port-channel; it cannot be an EtherChannel, and it cannot be a VLAN. • (Optional) [**,** | **-**] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged). • (Optional) **encapsulation dot1q** IEEE 802.1Q is a standard protocol for interconnecting multiple switches and routers and for defining VLAN topologies. Applies a VLAN ID to the subinterface.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Verifying the SPAN Session

Use the **show monitor session** command to verify the sources and destinations configured for the SPAN session.

```
Router#show monitor session 1

Session 1
---------
Session 1
---------
Type : Local Session
Source Ports :
Both : Gi0/1/0
Destination Ports : Gi0/1/1
```

Removing a SPAN Session

To remove sources or destinations from the SPAN session, use the **no monitor session** *session* command in global configuration mode as shown in the following example:

Router(config)#**no monitor session** *1*

Configuring a VLAN as an RSPAN VLAN

Follow these steps to create a new VLAN, then configure it to be the RSPAN VLAN for the RSPAN session.

Procedure

	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	**vlan** *vlan-id* **Example:** `Device(config)# vlan 100`	Enters a VLAN ID to create a VLAN, or enters the VLAN ID of an existing VLAN, and enters VLAN configuration mode. The range is 2 to 1001 and 1006 to 4094. The RSPAN VLAN cannot be VLAN 1 (the default VLAN) or VLAN IDs 1002 through 1005 (reserved for Token Ring and FDDI VLANs).
Step 4	**remote-span** **Example:** `Device(config-vlan)# remote-span`	Configures the VLAN as an RSPAN VLAN.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

You must create the RSPAN VLAN in all devices that will participate in RSPAN. If the RSPAN VLAN-ID is in the normal range (lower than 1005) and VTP is enabled in the network, you can create the RSPAN VLAN in one device, and VTP propagates it to the other devices in the VTP domain. For extended-range VLANs (greater than 1005), you must configure RSPAN VLAN on both source and destination devices and any intermediate devices.

Use VTP pruning to get an efficient flow of RSPAN traffic, or manually delete the RSPAN VLAN from all trunks that do not need to carry the RSPAN traffic.

To remove the remote SPAN characteristic from a VLAN and convert it back to a normal VLAN, use the **no remote-span** VLAN configuration command.

To remove a source port or VLAN from the SPAN session, use the **no monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} global configuration command. To remove the RSPAN VLAN from the session, use the **no monitor session** *session_number* {**Source|destination** }**remote vlan***vlan-id*.

Creating an RSPAN Source Session

Follow these steps to create and start an RSPAN source session and to specify the monitored source and the destination RSPAN VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 1`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.

Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**]
	Example:
	`Device(config)# monitor session 1 source interface gigabitethernet1/0/1 tx`
		Specifies the RSPAN session and the source port (monitored port).
		• For *session_number*, the range is 1 to 66.
		• Enter a source port or source VLAN for the RSPAN session:
		  • For *interface-id*, specifies the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel** *port-channel-number*). Valid port-channel numbers are 1 to 32.
		  • For *vlan-id*, specifies the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN).
		  A single session can include multiple sources (ports or VLANs), defined in a series of commands, but you cannot

	Command or Action	Purpose
		combine source ports and source VLANs in one session. • (Optional) [**,** | -]—Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **both** | **rx** | **tx**—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the source interface sends both sent and received traffic. • **both**—Monitors both received and sent traffic. • **rx**—Monitors received traffic. • **tx**—Monitors sent traffic.
Step 5	**monitor session** *session_number* **destination remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 1 destination remote vlan 100`	Specifies the RSPAN session, the destination RSPAN VLAN, and the destination-port group. • For *session_number*, enter the number defined in Step 4. • For *vlan-id*, specify the RSPAN VLAN in source session, which will transport mirrored traffic to destination session.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Specifying VLANs to Filter on RSPAN Source Session

Follow these steps to configure the RSPAN source session to limit RSPAN source traffic to specific VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source interface** *interface-id* **Example:** `Device(config)# monitor session 2 source interface gigabitethernet1/0/2 rx`	Specifies the characteristics of the source port (monitored port) and SPAN session. • For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. The interface specified must already be configured as a trunk port.
Step 5	**monitor session** *session_number* **filter vlan** *vlan-id* [**,** | -] **Example:** `Device(config)# monitor session 2 filter vlan 1 - 5 , 9`	Limits the SPAN source traffic to specific VLANs. • For *session_number*, enter the session number specified in step 4. • For *vlan-id*, the range is 1 to 4094. • (Optional) **,** | - Use a comma (**,**) to specify a series of VLANs or use a hyphen (-) to specify a range of VLANs. Enter a space before and after the comma; enter a space before and after the hyphen.

	Command or Action	Purpose
Step 6	**monitor session** *session_number* **destination remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 2 destination remote vlan 902`	Specifies the RSPAN session and the destination remote VLAN (RSPAN VLAN). • For *session_number*, enter the session number specified in Step 4. • For *vlan-id*, specify the RSPAN VLAN to carry the monitored traffic to the destination port.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating an RSPAN Destination Session and Configuring Incoming Traffic

Follow these steps to create an RSPAN destination session, to specify the source RSPAN VLAN and the destination port, and to enable incoming traffic on the destination port for a network security device (such as a Cisco IDS Sensor Appliance).

Procedure

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

	Command or Action	Purpose
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 2 source remote vlan 901`	Specifies the RSPAN session and the source RSPAN VLAN. • For *session_number*, the range is 1 to 66. • For *vlan-id*, specify the RSPAN VLAN in destination session, which will receive mirrored traffic from the source session.
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | -] [**ingress** {**dot1q vlan** *vlan-id* | **untagged vlan** *vlan-id* | **vlan** *vlan-id*}]} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet1/0/2 ingress vlan 6`	Specifies the SPAN session, the destination port, the packet encapsulation, and the incoming VLAN and encapsulation. • For *session_number*, enter the number defined in Step 5. In an RSPAN destination session, you must use the same session number for the source RSPAN VLAN and the destination port. • For *interface-id*, specify the destination interface. The destination interface must be a physical interface. • Though visible in the command-line help string, **encapsulation replicate** is not supported for RSPAN. The original VLAN ID is overwritten by the RSPAN VLAN ID, and all packets appear on the destination port as untagged. • (Optional) [**,** | -] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • Enter **ingress** with additional keywords to enable forwarding of incoming traffic on the destination port and to specify the encapsulation type:

	Command or Action	Purpose
		• **dot1q vlan** *vlan-id*—Forwards incoming packets with IEEE 802.1Q encapsulation with the specified VLAN as the default VLAN. • **untagged vlan** *vlan-id* or **vlan** *vlan-id*—Forwards incoming packets with untagged encapsulation type with the specified VLAN as the default VLAN.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

VLANs

A VLAN is a switched network that is logically segmented by function or application, without regard to the physical locations of the users. VLANs have the same attributes as physical LANs. However, you can group end-stations even if they are not physically located on the same LAN segment. Any device port can belong to a VLAN, unicast, broadcast, and multicast packets are forwarded and flooded only to end-stations in the VLAN. Each VLAN is considered a logical network, and packets destined for stations that do not belong to the VLAN must be forwarded through a router or a device supporting fallback bridging. In a device stack, VLANs can be formed with ports across the stack. Because a VLAN is considered a separate logical network, it contains its own bridge Management Information Base (MIB) information and can support its own implementation of spanning tree.

VLANs are often associated with IP subnetworks. For example, all the end stations in a particular IP subnet belong to the same VLAN. Interface VLAN membership on the device is assigned manually on an interface-by-interface basis. When you assign device interfaces to VLANs by using this method, it is known as interface-based, or static, VLAN membership.

The device can route traffic between VLANs by using device virtual interfaces (SVIs). An SVI must be explicitly configured and assigned an IP address to route traffic between VLANs.

Access Ports

An access port belongs to and carries the traffic of only one VLAN (unless it is configured as a voice VLAN port). Traffic is received and sent in native formats with no VLAN tagging. Traffic arriving on an access port is assumed to belong to the VLAN assigned to the port. If an access port receives a tagged packet IEEE 802.1Q tagged), the packet is dropped, and the source address is not learned.

Trunk Ports

A trunk port carries the traffic of multiple VLANs and by default is a member of all VLANs in the VLAN database. These trunk port types are supported:

- An IEEE 802.1Q trunk port supports simultaneous tagged and untagged traffic. An IEEE 802.1Q trunk port is assigned a default port VLAN ID (PVID), and all untagged traffic travels on the port default PVID. All untagged traffic and tagged traffic with a NULL VLAN ID are assumed to belong to the port default PVID. A packet with a VLAN ID equal to the outgoing port default PVID is sent untagged. All other traffic is sent with a VLAN tag.

Although by default, a trunk port is a member of every VLAN known to the VTP, you can limit VLAN membership by configuring an allowed list of VLANs for each trunk port. The list of allowed VLANs does not affect any other port but the associated trunk port. By default, all possible VLANs (VLAN ID 1 to 4094) are in the allowed list. A trunk port can become a member of a VLAN only if VTP knows of the VLAN and if the VLAN is in the enabled state. If VTP learns of a new, enabled VLAN and the VLAN is in the allowed list for a trunk port, the trunk port automatically becomes a member of that VLAN and traffic is forwarded to and from the trunk port for that VLAN. If VTP learns of a new, enabled VLAN that is not in the allowed list for a trunk port, the port does not become a member of the VLAN, and no traffic for the VLAN is forwarded to or from the port.

For more information on VLANs, see thehttps://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst9200/software/release/16-10/configuration_guide/vlan/b_1610_vlan_9200_cg/configuring_vlans.html

Creating a VLAN

Before you begin

With VTP version 1 and 2, if the device is in VTP transparent mode, you can assign VLAN IDs greater than 1006, but they are not added to the VLAN database.

To configure the Vlan, perform these steps. You can configure the Vlan in access or trunk mode. The procedure is same for the both the modes.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**vlan** *vlan-id* **Example:** `(config)# vlan 20`	Enters a VLAN ID, and enters VLAN configuration mode. Enter a new VLAN ID to create a VLAN, or enter an existing VLAN ID to modify that VLAN.

	Command or Action	Purpose
		Note The available VLAN ID range for this command is 1 to 4094.
Step 3	**name** *vlan-name* **Example:** `(config-vlan)# name test20`	(Optional) Enters a name for the VLAN. If no name is entered for the VLAN, the default is to append the *vlan-id* value with leading zeros to the word VLAN. For example, VLAN0004 is a default VLAN name for VLAN 4.
Step 4	**exit** **Example:** `(config-vlan)# exit`	Returns to configuration mode.
Step 5	**interface interface-id** **Example:** `router(config)# interface gigabitethernet1/0/1`	Specifies the physical port to be configured, and enter interface configuration mode.
Step 6	**switchport mode access** **Example:** `router(config-if)# switchport mode access`	Configures the interface as a VLAN access port.
Step 7	**switchport access vlan** *vlan id* **Example:** `router(config-if)# switchport access vlan 20`	Specifies the VLAN for which this access port will carry traffic. If you do not enter this command, the access port carries traffic on VLAN1 only; use this command to change the VLAN for which the access port carries traffic..
Step 8	**end** **Example:** `router(config-if)# end`	Returns to configuration mode.

Configuring LAN Ports for Layer 2 Switching

This section describes how configure all three types of ethernet LAN ports for Layer 2 switching on the Cisco 4000 series routers. The configuration tasks in this section apply to LAN ports on LAN switching modules.

Layer 2 LAN Port Modes

The following table lists the Layer 2 LAN port modes and describes how they function on LAN ports.

Table 4: Layer 2 LAN Port Modes

Mode	Function
switchport mode access	Puts the LAN port into permanent nontrunking mode and negotiates to convert the link into a nontrunk link. The LAN port becomes a nontrunk port even if the neighboring LAN port does not agree to the change.
switchport mode dynamic desirable	Makes the LAN port actively attempt to convert the link to a trunk link. The LAN port becomes a trunk port if the neighboring LAN port is set to **trunk**, **desirable**, or **auto** mode. This is the default mode for all LAN ports.
switchport mode dynamic auto	Makes the LAN port willing to convert the link to a trunk link. The LAN port becomes a trunk port if the neighboring LAN port is set to **trunk** or **desirable** mode.
switchport mode trunk	Puts the LAN port into permanent trunking mode and negotiates to convert the link into a trunk link. The LAN port becomes a trunk port even if the neighboring port does not agree to the change.
switchport nonegotiate	Puts the LAN port into permanent trunking mode but prevents the port from generating DTP frames. You must configure the neighboring port manually as a trunk port to establish a trunk link.

Note DTP is a point-to-point protocol. However, some internetworking devices might forward DTP frames improperly. To avoid this problem, ensure that LAN ports connected to devices that do not support DTP are configured with the **access** keyword if you do not intend to trunk across those links. To enable trunking to a device that does not support DTP, use the **nonegotiate** keyword to cause the LAN port to become a trunk but not generate DTP frames.

Default Layer 2 LAN Interface Configuration

The following table shows the Layer 2 LAN port default configuration.

Table 5: Layer 2 LAN Interface Default Configuration

Feature	Default
Interface mode:	
• Before entering the **switchport** command	
• After entering the **switchport** command	**switchport mode dynamic desirable**
Default access VLAN	VLAN 1
Native VLAN (for 802.1Q trunks)	VLAN 1

Configuring LAN Interfaces for Layer 2 Switching

These sections describe how to configure Layer 2 switching on the Cisco 4000 Series routers:

Note Use the **default interface** {**ethernet** | **fastethernet** | **gigabitethernet** | **tengigabitethernet**} *slot/subslot/port* command to revert an interface to its default configuration.

Spanning Tree Protocol Overview

Spanning Tree Protocol (STP) is a Layer 2 link management protocol that provides path redundancy while preventing loops in the network. For a Layer 2 Ethernet network to function properly, only one active path can exist between any two stations. Multiple active paths among end stations cause loops in the network. If a loop exists in the network, end stations might receive duplicate messages. Device might also learn end-station MAC addresses on multiple Layer 2 interfaces. These conditions result in an unstable network. Spanning-tree operation is transparent to end stations, which cannot detect whether they are connected to a single LAN segment or a switched LAN of multiple segments.

The STP uses a spanning-tree algorithm to select one device of a redundantly connected network as the root of the spanning tree. The algorithm calculates the best loop-free path through a switched Layer 2 network by assigning a role to each port based on the role of the port in the active topology:

- Root—A forwarding port elected for the spanning-tree topology
- Designated—A forwarding port elected for every switched LAN segment
- Alternate—A blocked port providing an alternate path to the root bridge in the spanning tree
- Backup—A blocked port in a loopback configuration

The device that has *all* of its ports as the designated role or as the backup role is the root device. The device that has at least *one* of its ports in the designated role is called the designated device.

Spanning tree forces redundant data paths into a standby (blocked) state. If a network segment in the spanning tree fails and a redundant path exists, the spanning-tree algorithm recalculates the spanning-tree topology and activates the standby path. Device send and receive spanning-tree frames, called bridge protocol data units (BPDUs), at regular intervals. The device do not forward these frames but use them to construct a loop-free path. BPDUs contain information about the sending device and its ports, including device and MAC addresses, device priority, port priority, and path cost. Spanning tree uses this information to elect the root device and root port for the switched network and the root port and designated port for each switched segment.

When two ports on a device are part of a loop, the spanning-tree and path cost settings control which port is put in the forwarding state and which is put in the blocking state. The spanning-tree port priority value represents the location of a port in the network topology and how well it is located to pass traffic. The path cost value represents the media speed.

Note By default, the device sends keepalive messages (to ensure the connection is up) only on interfaces that do not have small form-factor pluggable (SFP) modules. You can change the default for an interface by entering the [**no**] **keepalive** interface configuration command with no keywords.

Cisco SM-X-16G4M2X Layer 2 Gigabit EtherSwitch Service Module uses STP (the IEEE 802.1D bridge protocol) on all VLANs. By default, a single instance of STP runs on each configured VLAN (provided you do not manually disable STP). You can enable and disable STP on a per-VLAN basis.

For more information on STP, see https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst9200/software/release/16-10/configuration_guide/lyr2/b_1610_lyr2_9200_cg/configuring_spanning___tree_protocol.html

Default STP Configuration

The following table shows the default STP configuration.

Table 6: STP Default Configuration

Feature	Default Value
Disable state	STP disabled for all VLANs
Bridge priority	32768
STP port priority (configurable on a per-port basis—used on LAN ports configured as Layer 2 access ports)	128
STP port cost (configurable on a per-port basis—used on LAN ports configured as Layer 2 access ports)	Gigabit Ethernet: 4
STP VLAN port priority (configurable on a per-VLAN basis—used on LAN ports configured as Layer 2 trunk ports)	128
STP VLAN port cost (configurable on a per-VLAN basis—used on LAN ports configured as Layer 2 trunk ports)	Gigabit Ethernet:1000000000
Hello time	2 seconds
Forward delay time	15 seconds
Maximum aging time	20 seconds
Mode	PVST

Enabling STP

Note STP is disabled by default on all VLANs.

You can enable STP on a per-VLAN basis. The Cisco SM-X-16G4M2X or SM-X-40G8M2X Layer 2 Gigabit EtherSwitch Service Module maintain a separate instance of STP for each VLAN (except on VLANs on which you disable STP).

If you want to enable a mode that is different from the default mode, this procedure is required.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**spanning-tree mode** {**pvst** | **mst** | **rapid-pvst**}	Configures a spanning-tree mode. All stack members run the same version of spanning tree. • Select **pvst** to enable PVST+. • Select **mst** to enable MSTP. • Select **rapid-pvst** to enable rapid PVST+.
Step 3	**interface** *interface-id*	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports, VLANs, and port channels. The VLAN ID range is 1 to 4094. The port-channel range is 1 to 48.
Step 4	**spanning-tree link-type point-to-point** **Example:** `Device(config-if)# spanning-tree link-type point-to-point`	Specifies that the link type for this port is point-to-point. If you connect this port (local port) to a remote port through a point-to-point link and the local port becomes a designated port, the negotiates with the remote port and rapidly changes the local port to the forwarding state.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**clear spanning-tree detected-protocols** **Example:** `Device# clear spanning-tree detected-protocols`	If any port on the device is connected to a port on a legacy IEEE 802.1D device, this command restarts the protocol migration process on the entire device. This step is optional if the designated device detects that this device is running rapid PVST+.
Step 7	Device# **show spanning-tree vlan** *vlan_ID*	Verifies that STP is enabled.

What to do next

Caution Do not disable spanning tree on a VLAN unless all switches and bridges in the VLAN have spanning tree disabled. You cannot disable spanning tree on some switches and bridges in a VLAN and leave it enabled on other switches and bridges in the VLAN. This action can have unexpected results because switches and bridges with spanning tree enabled will have incomplete information regarding the physical topology of the network.

Caution We do not recommend disabling spanning tree, even in a topology that is free of physical loops. Spanning tree serves as a safeguard against misconfigurations and cabling errors. Do not disable spanning tree in a VLAN without ensuring that there are no physical loops present in the VLAN.

This example shows how to enable STP on VLAN 200:

```
Device# configure terminal
Device(config)# spanning-tree vlan 200

Device(config)# end

Device#
```

Note STP is disabled by default.

This example shows how to verify the configuration:

```
Device# show spanning-tree vlan 200

G0:VLAN0200
  Spanning tree enabled protocol ieee
  Root ID    Priority    32768
             Address     00d0.00b8.14c8
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
  Bridge ID  Priority    32768
             Address     00d0.00b8.14c8
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300
Interface        Role Sts Cost      Prio.Nbr Status
---------------- ---- --- --------- -------- --------------------------------
Gi1/4            Desg FWD 200000    128.196  P2p
Gi1/5            Back BLK 200000    128.197  P2p
Device#
```

Note You must have at least one interface that is active in VLAN 200 to create a VLAN 200 spanning tree. In this example, two interfaces are active in VLAN 200.

Configuring Optional STP Features

This section describes how to configure the following optional STP features:

Enabling PortFast

Caution Use PortFast *only* when connecting a single end station to a Layer 2 access port. Otherwise, you might create a network loop.

To enable PortFast on a Layer 2 access port, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **interface** {*type* [1] *slot/port* }	Selects a port to configure.
Step 2	Router(config-if)# **spanning-tree portfast**	Enables PortFast on a Layer 2 access port connected to a single workstation or server.
Step 3	Router(config-if)# **spanning-tree portfast default**	Enables PortFast.
Step 4	Router(config-if)# **end**	Exits configuration mode.
Step 5	Router# **show running interface** {*type* [2]*slot/port* }	Verifies the configuration.

Configuring PortFast BPDU Filtering

These sections describe how to configure PortFast BPDU filtering.

To enable PortFast BPDU filtering globally, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **spanning-tree portfast bpdufilter default**	Enables BPDU filtering globally on the router.
Step 2	Router# **show spanning-tree summary totals**	Verifies the configuration.

Enabling PortFast BPDU Filtering

BPDU filtering is set to default on each port. This example shows how to enable PortFast BPDU filtering on the port and verify the configuration in PVST+ mode:

```
Router(config)# spanning-tree portfast bpdufilter default

Router(config)# ^Z
Router# show spanning-tree summary totals

Switch is in pvst mode
Root bridge for: G0:VLAN0013, G0:VLAN0020, G1:VLAN0020
EtherChannel misconfig guard is enabled
Extended system ID           is enabled
Portfast Default             is disabled
PortFast BPDU Guard Default  is disabled
```

```
Portfast BPDU Filter Default is disabled
Loopguard Default            is disabled
UplinkFast                   is disabled
BackboneFast                 is disabled
Pathcost method used         is short
Name                   Blocking Listening Learning Forwarding STP Active
---------------------- -------- --------- -------- ---------- ----------
3 vlans                       0         0        0          3          3
```

To enable PortFast BPDU filtering on a nontrunking port, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **interface fastEthernet 4/4**	Selects the interface to configure.
Step 2	Router(config-if)# **spanning-tree bpdufilter enable**	Enables BPDU filtering.
Step 3	Router# **show spanning-tree interface fastEthernet 4/4**	Verifies the configuration.

What to do next

This example shows how to enable PortFast BPDU filtering on a nontrunking port:

```
Router(config)# interface fastEthernet 4/4
Router(config-if)# spanning-tree bpdufilter enable

Router(config-if)# ^Z
Router# show spanning-tree interface fastEthernet 4/4
Vlan             Role Sts Cost      Prio.Nbr Status
---------------- ---- --- --------- -------- --------------------------------
VLAN0010         Desg FWD 1000      160.196  Edge P2p
Router# show spanning-tree interface fastEthernet 4/4 detail

 Port 196 (FastEthernet4/4) of VLAN0010 is forwarding
   Port path cost 1000, Port priority 160, Port Identifier 160.196.
   Designated root has priority 32768, address 00d0.00b8.140a
   Designated bridge has priority 32768, address 00d0.00b8.140a
   Designated port id is 160.196, designated path cost 0
   Timers:message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state:1
   The port is in the portfast mode by portfast trunk configuration
   Link type is point-to-point by default
   Bpdu filter is enabled
   BPDU:sent 0, received 0
Router#
```

Enabling BPDU Guard

To enable BPDU Guard globally, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **spanning-tree portfast bpduguard default** **Example:** `Router(config)# no spanning-tree portfast bpduguard default`	Enables BPDU Guard globally. Disables BPDU Guard globally.
Step 2	Router(config)# **end**	Exits configuration mode.
Step 3	Router# **show spanning-tree summary totals**	Verifies the configuration.

What to do next

This example shows how to enable BPDU Guard:

```
Router# configure terminal
Router(config)# spanning-tree portfast bpduguard
Router(config)# end
Router#
```

This example shows how to verify the configuration:

```
Router# show spanning-tree summary totals
 default
Root bridge for:VLAN0010
EtherChannel misconfiguration guard is enabled
Extended system ID   is disabled
Portfast             is enabled by default
PortFast BPDU Guard  is disabled by default
Portfast BPDU Filter is enabled by default
Loopguard            is disabled by default
UplinkFast           is disabled
BackboneFast         is disabled
Pathcost method used is long
Name                   Blocking Listening Learning Forwarding STP Active
---------------------- -------- --------- -------- ---------- ----------
2 vlans                   0        0         0        3          3
Router#
```

Enabling UplinkFast

UplinkFast increases the bridge priority to 49152 and adds 3000 to the STP port cost of all Layer 2 LAN interfaces on the device, decreasing the probability that the router will become the root bridge. The *max_update_rate* value represents the number of multicast packets transmitted per second (the default is 150 packets per second). UplinkFast cannot be enabled on VLANs that have been configured for bridge priority. To enable UplinkFast on a VLAN with bridge priority configured, restore the bridge priority on the VLAN to the default value by entering a **no spanning-tree vlan** *vlan_ID* **priority** command in global configuration mode.

Note When you enable UplinkFast, it affects all VLANs on the device. You cannot configure UplinkFast on an individual VLAN.

To enable UplinkFast, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **spanning-tree uplinkfast** [**max-update-rate** *max_update_rate*]	Enables UplinkFast.
Step 2	Router(config)# **no spanning-tree uplinkfast max-update-rate**	Reverts to the default rate.
Step 3	Router(config)# **no spanning-tree uplinkfast**	Disables UplinkFast.
Step 4	Router(config)# **end**	Exits configuration mode.
Step 5	Router# **show spanning-tree vlan** *vlan_ID*	Verifies that UplinkFast is enabled.

What to do next

This example shows how to enable UplinkFast with an update rate of 400 packets per second:

```
Router# configure terminal

Router(config)# spanning-tree uplinkfast max-update-rate 400

Router(config)# exit

Router#
```

This example shows how to verify that UplinkFast is enabled:

```
Router# show spanning-tree uplinkfast

UplinkFast is enabled
Router#
```

Enabling BackboneFast

Note BackboneFast operates correctly only when enabled on all network devices in the network. BackboneFast is not supported on Token Ring VLANs. This feature is supported for use with third-party network devices.

To enable BackboneFast, perform this task:

Procedure

	Command or Action	Purpose
Step 1	Router(config)# **spanning-tree backbonefast**	Enables BackboneFast.
Step 2	Router(config)# **no spanning-tree backbonefast**	Disables BackboneFast.
Step 3	Router(config)# **end**	Exits configuration mode.
Step 4	Router# **show spanning-tree vlan** *vlan_ID*	Verifies that UplinkFast is enabled.

What to do next

This example shows how to enable BackboneFast:

```
Router# configure terminal

Router(config)# spanning-tree backbonefast

Router(config)# end

Router#
```

This example shows how to verify that BackboneFast is enabled:

```
Router# show spanning-tree backbonefast

BackboneFast is enabled
BackboneFast statistics
-----------------------
Number of transition via backboneFast (all VLANs) : 0
Number of inferior BPDUs received (all VLANs)     : 0
Number of RLQ request PDUs received (all VLANs)   : 0
Number of RLQ response PDUs received (all VLANs)  : 0
Number of RLQ request PDUs sent (all VLANs)       : 0
Number of RLQ response PDUs sent (all VLANs)      : 0
```

Router#

EtherChannel Overview

EtherChannel provides fault-tolerant high-speed links between switches, routers, and servers. You can use the EtherChannel to increase the bandwidth between the wiring closets and the data center, and you can deploy it anywhere in the network where bottlenecks are likely to occur. EtherChannel provides automatic recovery for the loss of a link by redistributing the load across the remaining links. If a link fails, EtherChannel redirects traffic from the failed link to the remaining links in the channel without intervention.

An EtherChannel consists of individual Ethernet links bundled into a single logical link

The EtherChannel provides full-duplex bandwidth up to 4 Gb/s (Gigabit EtherChannel) between your switch and another switch or host.

Each EtherChannel can consist of up to four compatibly configured Ethernet ports.

Channel Groups and Port-Channel Interfaces

An EtherChannel comprises a channel group and a port-channel interface. The channel group binds physical ports to the port-channel interface. Configuration changes applied to the port-channel interface apply to all the physical ports bound together in the channel group. The channel-group command binds the physical port and the port-channel interface together. Each EtherChannel has a port-channel logical interface numbered from 1 to 32. This port-channel interface number corresponds to the one specified with the channel-group interface configuration command.

Port Aggregation Protocol

The Port Aggregation Protocol (PAgP) is a Cisco-proprietary protocol that can be run only on Cisco devices and on those devices licensed by vendors to support PAgP. PAgP facilitates the automatic creation of EtherChannels by exchanging PAgP packets between Ethernet ports.

By using PAgP, the device learns the identity of partners capable of supporting PAgP and the capabilities of each port. It then dynamically groups similarly configured ports (on a single device in the stack) into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, PAgP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, PAgP adds the group to the spanning tree as a single device port.

Link Aggregation Control Protocol

The LACP is defined in IEEE 802.3ad and enables Cisco devices to manage Ethernet channels between devices that conform to the IEEE 802.3ad protocol. LACP facilitates the automatic creation of EtherChannels by exchanging LACP packets between Ethernet ports.

By using LACP, the switch learns the identity of partners capable of supporting LACP and the capabilities of each port. It then dynamically groups similarly configured ports into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, LACP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, LACP adds the group to the spanning tree as a single device port.

Auto-LAG

The auto-LAG feature provides the ability to auto create EtherChannels on ports connected to a switch. By default, auto-LAG is disabled globally and is enabled on all port interfaces. The auto-LAG applies to a switch only when it is enabled globally.

On enabling auto-LAG globally, the following scenarios are possible:

- All port interfaces participate in creation of auto EtherChannels provided the partner port interfaces have EtherChannel configured on them. For more information, see the "The supported auto-LAG configurations between the actor and partner devices" table below.
- Ports that are already part of manual EtherChannels cannot participate in creation of auto EtherChannels.
- When auto-LAG is disabled on a port interface that is already a part of an auto created EtherChannel, the port interface will unbundle from the auto EtherChannel.
- The following table shows the supported auto-LAG configurations between the actor and partner devices:

Table 7: The supported auto-LAG configurations between the actor and partner devices

Actor/Partner	Active	Passive	Auto
Active	Yes	Yes	Yes
Passive	Yes	No	Yes
Auto	Yes	Yes	Yes

On disabling auto-LAG globally, all auto created Etherchannels become manual EtherChannels.

You cannot add any configurations in an existing auto created EtherChannel. To add, you should first convert it into a manual EtherChannel by executing the **port-channel**<*channel-number*>**persistent**.

Configuring Layer 2 EtherChannels

Configure Layer 2 EtherChannels by assigning ports to a channel group with the **channel-group** command in interface configuration mode. This command automatically creates the port-channel logical interface.

Use the **show etherchannel swport xxx** command to view the Cisco SM-X-16G4M2X EtherChannels.

Procedure

	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 gigabitethernet 1/0/1`	Specifies a physical port, and enters interface configuration mode. Valid interfaces are physical ports. For a PAgP EtherChannel, you can configure up to four ports of the same type and speed for the same group. For a LACP EtherChannel, you can configure up to 8 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
Step 4	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks. If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.

	Command or Action	Purpose
Step 5	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 22`	(Optional) If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.
Step 6	**channel-group** *channel-group-number* **mode** {**auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive**} **Example:** `Device(config-if)# channel-group 5 mode auto`	Assigns the port to a channel group, and specifies the PAgP or the LACP mode. For **mode**, select one of these keywords: • **auto** –Enables PAgP only if a PAgP device is detected. It places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. • **desirable** –Unconditionally enables PAgP. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. • **on** –Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **non-silent** –(Optional) If your device is connected to a partner that is PAgP-capable, configures the device port for nonsilent operation when the port is in the **auto** or **desirable** mode. If you do not specify **non-silent**, silent is assumed. The silent setting is for connections to file servers or packet analyzers. This setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission. • **active**—Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** –Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring EtherChannel Load-Balancing

You can configure EtherChannel load-balancing to use one of several different forwarding methods.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**port-channel swport load-balance** { **dst-ip** | **dst-mac** | **dst-mixed-ip-port** | **dst-port** | **extended** [**dst-ip** | **dst-mac** | **dst-port** | **ipv6-label** | **l3-proto** | **src-ip** | **src-mac** | **src-port**] | **src-dst-ip** | **src-dst-mac** **src-dst-mixed-ip-port src-dst-portsrc-ip** | **src-mac** | **src-mixed-ip-port** | **src-port** } **Example:** `Device(config)# port-channel swport load-balance src-mac`	Configures an EtherChannel load-balancing method. Select one of these load-distribution methods: • **dst-ip**—Specifies destination-host IP address. • **dst-mac**—Specifies the destination-host MAC address of the incoming packet. • **dst-mixed-ip-port**—Specifies the host IP address and TCP/UDP port. • **dst-port**—Specifies the destination TCP/UDP port. • **extended**—Specifies extended load balance methods--combinations of source and destination methods beyond those available with the standard command. • **ipv6-label**—Specifies the IPv6 flow label. • **l3-proto**—Specifies the Layer 3 protocol. • **src-dst-ip**—Specifies the source and destination host IP address. • **src-dst-mac**—Specifies the source and destination host MAC address. • **src-dst-mixed-ip-port**—Specifies the source and destination host IP address and TCP/UDP port.

	Command or Action	Purpose
		• **src-dst-port**—Specifies the source and destination TCP/UDP port. • **src-ip**—Specifies the source host IP address. • **src-mac**—Specifies the source MAC address of the incoming packet. • **src-mixed-ip-port**—Specifies the source host IP address and TCP/UDP port. • **src-port**—Specifies the source TCP/UDP port.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the PAgP Learn Method and Priority

This task is optional.

Procedure

	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 gigabitethernet 1/0/2`	Specifies the port for transmission, and enters interface configuration mode.
Step 4	**pagp learn-method physical-port** **Example:** `Device(config-if)# pagp learn-method physical port`	Selects the PAgP learning method. By default, **aggregation-port learning** is selected, which means the device sends packets to the source by using any of the ports in the EtherChannel. With aggregate-port learning, it is not important on which physical port the packet arrives.

	Command or Action	Purpose
		Selects **physical-port** to connect with another device that `is` a physical learner. Make sure to configure the **port-channel load-balance** global configuration command to **src-mac**. The learning method must be configured the same at both ends of the link.
Step 5	**pagp port-priority** *priority* **Example:** `Device(config-if)# pagp port-priority 200`	Assigns a priority so that the selected port is chosen for packet transmission. For *priority*, the range is 0 to 255. The default is 128. The higher the priority, the more likely that the port will be used for PAgP transmission.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring the LACP Port Channel Min-Links Feature

You can specify the minimum number of active ports that must be in the link-up state and bundled in an EtherChannel for the port channel interface to transition to the link-up state. Using EtherChannel min-links, you can prevent low-bandwidth LACP EtherChannels from becoming active. Port channel min-links also cause LACP EtherChannels to become inactive if they have too few active member ports to supply the required minimum bandwidth.

To configure the minimum number of links that are required for a port channel. Perform the following tasks.

Procedure

	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 port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 2`	Enters interface configuration mode for a port-channel. For *channel-number*, the range is 1 to 63.
Step 4	**port-channel min-links** *min-links-number* **Example:**	Specifies the minimum number of member ports that must be in the link-up state and bundled in

	Command or Action	Purpose
	`Device(config-if)# port-channel min-links 3`	the EtherChannel for the port channel interface to transition to the link-up state. For *min-links-number* , the range is 2 to 8.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring LACP Fast Rate Timer

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate** command to set the rate at which LACP control packets are received by an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

Procedure

	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** {**fastethernet** | **gigabitethernet** | **tengigabitethernet**} *slot/port* **Example:** `Device(config)# interface gigabitEthernet 2/1`	Configures an interface and enters interface configuration mode.
Step 4	**lacp rate** {**normal** | **fast**} **Example:** `Device(config-if)# lacp rate fast`	Configures the rate at which LACP control packets are received by an LACP-supported interface. To reset the timeout rate to its default, use the **no lacp rate** command.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show lacp internal** **Example:**	Verifies your configuration.

	Command or Action	Purpose
	`Device# show lacp internal` `Device# show lacp counters`	

Configuring Auto-LAG Globally

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **port-channel swport auto** **Example:** `Device(config)# port-channel swport auto`	Enables the auto-LAG feature on a switch globally. Use the no form of this command to disable the auto-LAG feature on the switch globally. **Note** By default, the auto-LAG feature is enabled on the port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show etherchannel swport auto** **Example:** `Device# show etherchannel swport auto`	Displays that EtherChannel is created automatically.

Modular Quality of Service Command-Line Interface

The MQC (Modular Quality of Service (QoS) Command-Line Interface (CLI)) enables you to set packet classification and marking based on a QoS group value. ith the device, QoS features are enabled through the Modular QoS command-line interface (MQC). The MQC is a command-line interface (CLI) structure that allows you to create traffic policies and attach these policies to interfaces. A traffic policy contains a traffic class and one or more QoS features. A traffic class is used to classify traffic, while the QoS features in the traffic policy determine how to treat the classified traffic. One of the main goals of MQC is to provide a platform-independent interface for configuring QoS across Cisco platforms. For more infomraton on the Modular Quality of Service, see the Quality of Service Configuration Guide, Cisco IOS XE Fuji 16.9.x.

Creating a Traffic Class

To create a traffic class containing match criteria, use the **class-map** command to specify the traffic class name, and then use the following **match** commands in class-map configuration mode, as needed.

Before you begin

All match commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 2	**class-map** *class-map name* { **match-any** }
	Example:
	`Device(config)# class-map type ngsw-qos test_1000`
	`Device(config-cmap)#`
		Enters class map configuration mode.
		• Creates a class map to be used for matching packets to the class whose name you specify.
		• **match-any:** Any one of the match criteria must be met for traffic entering the traffic class to be classified as part of it.

Step 3	**match access-group** { *index number* | *name* }
	Example:
	`Device(config-cmap)# match access-group 100`
	`Device(config-cmap)#`
		The following parameters are available for this command:
		• access-group
		• cos
		• dscp
		• group-object
		• ip
		• mpls
		• precedence
		• protocol
		• qos-group
		• vlan
		• wlan
		(Optional) For this example, enter the access-group ID:

	Command or Action	Purpose
		• Access list index (value from 1 to 2799) • Named access list

Step 4	**match cos** *cos value*
	Example:
	`Device(config-cmap)# match cos 2 3 4 5`
	`Device(config-cmap)#`
		(Optional) Matches IEEE 802.1Q or ISL class of service (user) priority values.
		• Enters up to 4 CoS values separated by spaces (0 to 7).

Step 5	**match dscp** *dscp value*
	Example:
	`Device(config-cmap)# match dscp af11 af12`
	`Device(config-cmap)#`
		(Optional) Matches the DSCP values in IPv4 and IPv6 packets.

Step 6	**match ip** {**dscp** *dscp value* | **precedence** *precedence value* }
	Example:
	`Device(config-cmap)# match ip dscp af11 af12`
	`Device(config-cmap)#`
		(Optional) Matches IP values including the following:
		• **dscp**—Matches IP DSCP (DiffServ codepoints).
		• **precedence**—Matches IP precedence (0 to 7).

Step 7	**match qos-group** *qos group value*
	Example:
	`Device(config-cmap)# match qos-group 10`
	`Device(config-cmap)#`
		(Optional) Matches QoS group value (from 0 to 31).

Step 8	**match vlan** *vlan value*
	Example:
	`Device(config-cmap)# match vlan 210`
	`Device(config-cmap)#`
		(Optional) Matches a VLAN ID (from 1 to 4095).

Step 9	**end** **Example:** `Device(config-cmap)# end`	Saves the configuration changes.

What to do next

Configure the policy map.

Creating a Traffic Policy

To create a traffic policy, use the **policy-map** global configuration command to specify the traffic policy name.

The traffic class is associated with the traffic policy when the **class** command is used. The **class** command must be entered after you enter the policy map configuration mode. After entering the **class** command, the device is automatically in policy map class configuration mode, which is where the QoS policies for the traffic policy are defined.

The following policy map class-actions are supported:

- bandwidth—Bandwidth configuration options.
- exit—Exits from the QoS class action configuration mode.
- no—Negates or sets default values for the command.
- police—Policer configuration options.
- priority—Strict scheduling priority configuration options for this class.
- queue-buffers—Queue buffer configuration options.
- queue-limit—Queue maximum threshold for Weighted Tail Drop (WTD) configuration options.
- service-policy—Configures the QoS service policy.
- set—Sets QoS values using the following options:
  - CoS values
  - DSCP values
  - Precedence values
  - QoS group values
- shape—Traffic-shaping configuration options.

Before you begin

You should have first created a class map.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map type***policy-map name* **Example:**	Enters policy map configuration mode.

	Command or Action	Purpose
	`Device(config)# policy-map type ngsw-qos test_1000`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class test_1000`	Specifies the name of the class whose policy you want to create or change. You can also create a system default class for unclassified packets.
Step 4	**bandwidth** {**kb/s** *kb/s value* | **percent** *percentage* | **remaining** {*percent* | *ratio*}} **Example:** `Device(config-pmap-c)# bandwidth 50`	(Optional) Sets the bandwidth using one of the following: • **kb/s**—Kilobits per second, enter a value between 20000 and 10000000 for Kb/s. • **percent**—Enter the percentage of the total bandwidth to be used for this policy map. • **remaining**—Enter the percentage ratio of the remaining bandwidth.
Step 5	**exit** **Example:** `Device(config-pmap-c)# exit`	(Optional) Exits from QoS class action configuration mode.
Step 6	**no** **Example:** `Device(config-pmap-c)# no`	(Optional) Negates the command.
Step 7	**police** {*target_bit_rate* | **cir** | **rate**} **Example:** `Device(config-pmap-c)# police 100000`	(Optional) Configures the policer: • *target_bit_rate*—Enter the bit rate per second, enter a value between 8000 and 10000000000. • **cir**—Committed Information Rate • **rate**—Specify police rate, PCR for hierarchical policies or SCR for single-level ATM 4.0 policer policies.
Step 8	**Example:** `Device(config-pmap-c)#`	(Optional) Sets the strict scheduling priority for this class. Command options include: • **level**—Establishes a multi-level priority queue. Enter a value (1 or 2).

	Command or Action	Purpose
Step 9	**queue-buffers ratio** *ratio limit* **Example:** `Device(config-pmap-c)# queue-buffers ratio 10`	(Optional) Configures the queue buffer for the class. Enter the queue buffers ratio limit (0 to 100).
Step 10	**queue-limit** {*packets* | **cos** | **dscp** | **percent**} **Example:** `Device(config-pmap-c)# queue-limit cos 7 percent 50`	(Optional) Specifies the queue maximum threshold for the tail drop: • *packets*—Packets by default, enter a value between 1 to 2000000. • **cos**—Enter the parameters for each COS value. • **dscp**—Enter the parameters for each DSCP value. • **percent**—Enter the percentage for the threshold.
Step 11	**service-policy** *policy-map name* **Example:** `Device(config-pmap-c)# service-policy test_2000`	(Optional) Configures the QoS service policy.
Step 12	**set** {**cos** | **dscp** | **ip** | **precedence** | **qos-group** | **wlan**} **Example:** `Device(config-pmap-c)# set cos 7`	(Optional) Sets the QoS values. Possible QoS configuration values include: • **cos**—Sets the IEEE 802.1Q/ISL class of service/user priority. • **dscp**—Sets DSCP in IP(v4) and IPv6 packets. • **ip**—Sets IP specific values. • **precedence**—Sets precedence in IP(v4) and IPv6 packet. • **qos-group**—Sets the QoS Group.
Step 13	**shape average** {*target _bit_rate* | **percent**} **Example:** `Device(config-pmap-c) #shape average percent 50`	(Optional) Sets the traffic shaping. Command parameters include: • *target_bit_rate*—Target bit rate. • **percent**—Percentage of interface bandwidth for Committed Information Rate.

	Command or Action	Purpose
Step 14	**end** **Example:** `Device(config-pmap-c) #end`	Saves the configuration changes.

What to do next

Configure the interface.

Configuring Class-Based Packet Marking

This is an important procedure that explains how to configure the following class-based packet marking features on your device:

- CoS value
- DSCP value
- IP value
- Precedence value
- QoS group value
- WLAN value

Before you begin

You should have created a class map and a policy map before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 2	**policy-maptype***policy name*
	Example:
	`Device(config)# policy-map type ngsw-qos policy1`
	`Device(config-pmap)#`
		Enters policy map configuration mode.
		Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.

	Command or Action	Purpose

Step 3	**class** *class name*
	Example:
	`Device(config-pmap)# class class1`
		Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change.
		Command options for policy class map configuration mode include the following:
		• **bandwidth**—Bandwidth configuration options.
		• **exit**—Exits from the QoS class action configuration mode.
		• **no**—Negates or sets default values for the command.
		• **police**—Policer configuration options.
		• **priority**—Strict scheduling priority configuration options for this class.
		• **queue-buffers**—Queue buffer configuration options.
		• **queue-limit**—Queue maximum threshold for Weighted Tail Drop (WTD) configuration options.
		• **service-policy**—Configures the QoS service policy.
		• **set**—Sets QoS values using the following options:
		  • CoS values
		  • DSCP values
		  • Precedence values
		  • QoS group values
		  • WLAN values
		• **shape**—Traffic-shaping configuration options.

	Command or Action	Purpose
		Note This procedure describes the available configurations using **set** command options. The other command options (**bandwidth**) are described in other sections of this guide. Although this task lists all of the possible **set** commands, only one **set** command is supported per class.
Step 4	**set cos** {*cos value* | **cos table** *table-map name* | **dscp table** *table-map name* | **precedence table** *table-map name* | **qos-group table** *table-map name* | **wlan user-priority table** *table-map name*} **Example:** `Device(config-pmap)# set cos 5`	(Optional) Sets the specific IEEE 802.1Q Layer 2 CoS value of an outgoing packet. Values are from 0 to7. You can also set the following values using the **set cos** command: • **cos table**—Sets the CoS value based on a table map. • **dscp table**—Sets the code point value based on a table map. • **precedence table**—Sets the code point value based on a table map. • **qos-group table**—Sets the CoS value from QoS group based on a table map. • **wlan user-priority table**—Sets the CoS value from the WLAN user priority based on a table map.
Step 5	**set dscp** {*dscp value* | **default** | **dscp table** *table-map name* | **ef** | **precedence table** *table-map name* | **qos-group table** *table-map name* | **wlan user-priority table** *table-map name*} **Example:** `Device(config-pmap)# set dscp af11`	(Optional) Sets the DSCP value. In addition to setting specific DSCP values, you can also set the following using the **set dscp** command: • **default**—Matches packets with default DSCP value (000000). • **dscp table**—Sets the packet DSCP value from DSCP based on a table map. • **ef**—Matches packets with EF DSCP value (101110). • **precedence table**—Sets the packet DSCP value from precedence based on a table map.

	Command or Action	Purpose
		• **qos-group table**—Sets the packet DSCP value from a QoS group based upon a table map. • **wlan user-priority table**—Sets the packet DSCP value based upon a WLAN user-priority based upon a table map.
Step 6	**set ip** {**dscp** | **precedence**} **Example:** `Device(config-pmap)# set ip dscp c3`	(Optional) Sets IP specific values. These values are either IP DSCP or IP precedence values. You can set the following values using the **set ip dscp** command: • *dscp value*—Sets a specific DSCP value. • **default**—Matches packets with default DSCP value (000000). • **dscp table**—Sets the packet DSCP value from DSCP based on a table map. • **ef**—Matches packets with EF DSCP value (101110). • **precedence table**—Sets the packet DSCP value from precedence based on a table map. • **qos-group table**—Sets the packet DSCP value from a QoS group based upon a table map. • **wlan user-priority table**—Sets the packet DSCP value based upon a WLAN user-priority based upon a table map. You can set the following values using the **set ip precedence** command: • *precedence value*—Sets the precedence value (from 0 to 7) . • **cos table**—Sets the packet precedence value from Layer 2 CoS based on a table map. • **dscp table**—Sets the packet precedence from DSCP value based on a table map. • **precedence table**—Sets the precedence value from precedence based on a table map

	Command or Action	Purpose
		• **qos-group table**—Sets the precedence value from a QoS group based upon a table map.
Step 7	**set precedence** {*precedence value* | **cos table** *table-map name* | **dscp table** *table-map name* | **precedence table** *table-map name* | **qos-group table** *table-map name*} **Example:** `Device(config-pmap)# set precedence 5`	(Optional) Sets precedence values in IPv4 and IPv6 packets. You can set the following values using the **set precedence** command: • *precedence value*—Sets the precedence value (from 0 to 7) . • **cos table**—Sets the packet precedence value from Layer 2 CoS on a table map. • **dscp table**—Sets the packet precedence from DSCP value on a table map. • **precedence table**—Sets the precedence value from precedence based on a table map. • **qos-group table**—Sets the precedence value from a QoS group based upon a table map.
Step 8	**set qos-group** {*qos-group value* | **dscp table** *table-map name* | **precedence table** *table-map name*} **Example:** `Device(config-pmap)# set qos-group 10`	(Optional) Sets QoS group values. You can set the following values using this command: • *qos-group value*—A number from 1 to 31. • **dscp table**—Sets the code point value from DSCP based on a table map. • **precedence table**—Sets the code point value from precedence based on a table map.
Step 9	**set wlan user-priority** {*wlan user-priority value* | **cos table** *table-map name* | **dscp table** *table-map name* | **qos-group table** *table-map name* | **wlan table** *table-map name*} **Example:** `Device(config-pmap)# set wlan user-priority 1`	(Optional) Sets the WLAN user priority value. You can set the following values using this command: • *wlan user-priority value*—A value between 0 to 7. • **cos table**—Sets the WLAN user priority value from CoS based on a table map. • **dscp table**—Sets the WLAN user priority value from DSCP based on a table map.

	Command or Action	Purpose
		• **qos-group table**—Sets the WLAN user priority value from QoS group based on a table map. • **wlan table**—Sets the WLAN user priority value from the WLAN user priority based on a table map.
Step 10	**end** **Example:** `Device(config-pmap)# end`	Saves configuration changes.
Step 11	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Attach the traffic policy to an interface using the **service-policy** command.

Attaching a Traffic Policy to an Interface

After the traffic class and traffic policy are created, you must use the **service-policy** interface configuration command to attach a traffic policy to an interface, and to specify the direction in which the policy should be applied (either on packets coming into the interface or packets leaving the interface).

Before you begin

A traffic class and traffic policy must be created before attaching a traffic policy to an interface.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type* **Example:**	
Step 3	**service-policy** {**input** *policy-map* | **output** *policy-map*} **Example:**	Attaches a policy map to an input or output interface. This policy map is then used as the service policy for that interface.

	Command or Action	Purpose
	`Device(config-if)# service-policy output policy_map_01`	In this example, the traffic policy evaluates all traffic leaving that interface.
Step 4	**end** **Example:** `Device(config-if)# end`	Saves configuration changes.
Step 5	**show policy map** **Example:** `Device# show policy map`	(Optional) Displays statistics for the policy on the specified interface.

What to do next

Proceed to attach any other traffic policy to an interface, and to specify the direction in which the policy should be applied.

Classifying, Policing, and Marking Traffic on Physical Ports by Using Policy Maps

You can configure a nonhierarchical policy map on a physical port that specifies which traffic class to act on. Actions supported are remarking and policing.

Before you begin

You should have already decided upon the classification, policing, and marking of your network traffic by policy maps prior to beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 2	**class-map** {*class-map name* | **match-any**}
	Example:
	`Device(config)# class-map ipclass1`
	`Device(config-cmap)# exit`
		Enters class map configuration mode.
		• Creates a class map to be used for matching packets to the class whose name you specify.
		• If you specify **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified

	Command or Action	Purpose
		as part of the traffic class. This is the default.

Step 3	**match access-group** { *access list index* | *access list name* }
	Example:
	`Device(config-cmap)# match access-group 1000`
	`Device(config-cmap)# exit`
		The following parameters are available for this command:
		• access-group
		• cos
		• dscp
		• group-object
		• ip
		• mpls
		• precedence
		• protocol
		• qos-group
		• vlan
		• wlan
		(Optional) For this example, enter the access-group ID:
		• Access list index (value from 1 to 2799)
		• Named access list

Step 4	**policy-map** *policy-map-name* **Example:** `Device(config)# olicy-map type ngsw-qos flowit`	Creates a policy map by entering the policy map name, and enters policy-map configuration mode. By default, no policy maps are defined.
Step 5	**class** {*class-map-name* | **class-default**} **Example:** `Device(config-pmap)# class ipclass1`	Defines a traffic classification, and enter policy-map class configuration mode. By default, no policy map class-maps are defined. If a traffic class has already been defined by using the **class-map** global configuration command, specify its name for *class-map-name* in this command. A **class-default** traffic class is predefined and can be added to any policy. It is always placed at the end of a policy map. With an implied

	Command or Action	Purpose
		match any included in the **class-default** class, all packets that have not already matched the other traffic classes will match **class-default**.
Step 6	**set** {**cos** | **dscp** | **ip** | **precedence** | **qos-group** | **wlan user-priority**} **Example:** `Device(config-pmap-c)# set dscp 45`	(Optional) Sets the QoS values. Possible QoS configuration values include: • **cos**—Sets the IEEE 802.1Q/ISL class of service/user priority. • **dscp**—Sets DSCP in IP(v4) and IPv6 packets. • **ip**—Sets IP specific values. • **precedence**—Sets precedence in IP(v4) and IPv6 packet. • **qos-group**—Sets QoS group. • **wlan user-priority**—Sets WLAN user priority. In this example, the **set dscp** command classifies the IP traffic by setting a new DSCP value in the packet.
Step 7	**police** {*target_bit_rate* | **cir** | **rate** } **Example:** `Device(config-pmap-c)# police 100000 conform-action transmit exceed-action drop`	(Optional) Configures the policer: • *target_bit_rate*—Specifies the bit rate per second, enter a value between 8000 and 10000000000. • **cir**—Committed Information Rate. • **rate**—Specifies the police rate PCR for hierarchical policies. In this example, the **police** command adds a policer to the class where any traffic beyond the 100000 set target bit rate is dropped.
Step 8	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy map configuration mode.
Step 9	**exit** **Example:** `Device(config-pmap)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface** *interface-id* **Example:** `Device(config)# interface HundredGigabitEthernet 1/0/2`	Specifies the port to attach to the policy map, and enters interface configuration mode. Valid interfaces include physical ports.
Step 11	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input flowit`	Specifies the policy-map name, and applies it to an ingress port. Only one policy map per ingress port is supported.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 13	**show policy-map** [*policy-map-name* [**class** *class-map-name*]] **Example:** `Device# show policy-map`	(Optional) Verifies your entries.
Step 14	**copy running-config startup-config** **Example:** `Device# copy-running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

If applicable to your QoS configuration, configure classification, policing, and marking of traffic on SVIs by using policy maps.

MACsec Encryption

This section describes how to configure MACsec encryption on Cisco SM-X-16G4M2X or SM-X-40G8M2X.

Prerequisites for MACsec Encryption

- Ensure that you have configured Cisco Identity Services Engine (ISE) Release 2.0.
- Ensure that 802.1x authentication and AAA are configured on your device.

Restrictions for MACsec Encryption

- MACsec configuration is not supported on EtherChannel ports.
- HSEC license is required to configure MACsec encryption.
- Only MKA pre-shared key approach is supported for switch-to-switch MACsec. CTS/SAP (NDAC) and certificated-based MKA is not supported.
- Extended Packet Numbering (XPN) is not supported.
- VLAN Tag in clear is not supported.

Information About MACsec Encryption

Recommendations for MACsec Encryption

This section list the recommendations for configuring MACsec encryption:

- Use the confidentiality (encryption) offset as 0 in switch-to-host connections.
- Execute the **shutdown** command, and then the **no shutdown** command on a port, after changing any MKA policy or MACsec configuration for active sessions, so that the changes are applied to active sessions.
- Set the connectivity association key (CAK) rekey overlap timer to 30 seconds or more.

MACsec Encryption Overview

MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Cisco SM-X-16G4M2X or SM-X-40G8M2X supports 802.1AE encryption with MACsec Key Agreement (MKA) on switch-to-host links for encryption between the switch and host device. The switch also supports MACsec encryption for switch-to-switch (inter-network device) security using MKA-based key exchange protocol.

Link layer security can include both packet authentication between switches and MACsec encryption between switches (encryption is optional).

Table 8: MACsec Support on Switch Ports

Connections	MACsec support
Switch-to-host	MACsec MKA encryption
Switch-to-switch	MACsec MKA encryption

MKA is supported on switch-to-host facing links. Host-facing links typically use flexible authentication ordering for handling heterogeneous devices with or without IEEE 802.1x, and can optionally use MKA-based MACsec encryption.

Media Access Control Security and MACsec Key Agreement

MACsec, defined in 802.1AE, provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session

keys and manages the required encryption keys. MKA and MACsec are implemented after successful authentication using certificate-based MACsec or Pre Shared Key (PSK) framework.

A device using MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the MKA peer. MACsec frames are encrypted and protected with an integrity check value (ICV). When the device receives frames from the MKA peer, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The device compares that ICV to the ICV within the frame. If they are not identical, the frame is dropped. The device also encrypts and adds an ICV to any frames sent over the secured port (the access point used to provide the secure MAC service to a MKA peer) using the current session key.

The MKA Protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1x-REV. The MKA Protocol extends 802.1x to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by the peers.

The EAP framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). The device acts as the key server for both uplink and downlink; and acts as the authenticator for downlink. It generates a random secure association key (SAK), which is sent to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the device sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes with no MKPDU received from a participant. For example, if a MKA peer disconnects, the participant on the device continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the MKA peer.

Note Integrity check value (ICV) indicator in MKPDU is optional. ICV is not optional when the traffic is encrypted.

EAPoL Announcements indicate the use of the type of keying material. The announcements can be used to announce the capability of the supplicant as well as the authenticator. Based on the capability of each side, the largest common denominator of the keying material could be used.

MKA Policies

To enable MKA on an interface, a defined MKA policy should be applied to the interface. You can configure these options:

- Policy name, not to exceed 16 ASCII characters.
- Confidentiality (encryption) offset of 0, 30, or 50 bytes for each physical interface

Definition of Policy-Map Actions

This section describes the policy-map actions and its definition:

- Activate: Applies a service template to the session.
- Authenticate: Starts authentication of the session.
- Authorize: Explicitly authorizes a session.

- Set-domain: Explicitly sets the domain of a client.
- Terminate: Terminates the method that is running, and deletes all the method details associated with the session.
- Deactivate: Removes the service-template applied to the session. If not applied, no action is taken.
- Set-timer: Starts a timer and gets associated with the session. When the timer expires, any action that needs to be started can be processed.
- Authentication-restart: Restarts authentication.
- Clear-session: Deletes a session.
- Pause: Pauses authentication.

Rest of the actions as self-explanatory and are associated with authentication.

Virtual Ports

Use virtual ports for multiple secured connectivity associations on a single physical port. Each connectivity association (pair) represents a virtual port. In uplink, you can have only one virtual port per physical port. You cannot simultaneously host secured and unsecured sessions in the same VLAN on the same port. Because of this limitation, 802.1x multiple authentication mode is not supported.

The exception to this limitation is in multiple-host mode when the first MACsec supplicant is successfully authenticated and connected to a hub that is connected to the device. A non-MACsec host connected to the hub can send traffic without authentication because it is in multiple-host mode. We do not recommend using multi-host mode because after the first successful client, authentication is not required for other clients.

Virtual ports represent an arbitrary identifier for a connectivity association and have no meaning outside the MKA Protocol. A virtual port corresponds to a separate logical port ID. Valid port IDs for a virtual port are 0x0002 to 0xFFFF. Each virtual port receives a unique secure channel identifier (SCI) based on the MAC address of the physical interface concatenated with a 16-bit port ID.

MKA Statistics

Some MKA counters are aggregated globally, while others are updated both globally and per session.

Key Lifetime and Hitless Key Rollover

A MACsec key chain can have multiple pre-shared keys (PSK) each configured with a key id and an optional lifetime. A key lifetime specifies at which time the key expires. In the absence of a lifetime configuration, the default lifetime is unlimited. When a lifetime is configured, MKA rolls over to the next configured pre-shared key in the key chain after the lifetime is expired. Time zone of the key can be local or UTC. Default time zone is UTC.

You can Key rolls over to the next key within the same key chain by configuring a second key in the key chain and configuring a lifetime for the first key. When the lifetime of the first key expires, it automatically rolls over to the next key in the list. If the same key is configured on both sides of the link at the same time, then the key rollover is hitless, that is, key rolls over without traffic interruption.

Note The lifetime of the keys need to be overlapped in order to achieve hitless key rollover.

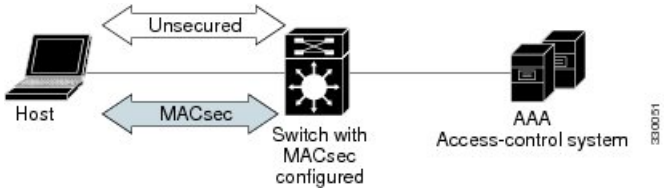
MACsec, MKA and 802.1x Host Modes

You can use MACsec and the MKA Protocol with 802.1x single-host mode, multi-host mode, or Multi Domain Authentication (MDA) mode. Multiple authentication mode is not supported.

Single-Host Mode

The figure shows how a single EAP authenticated session is secured by MACsec by using MKA

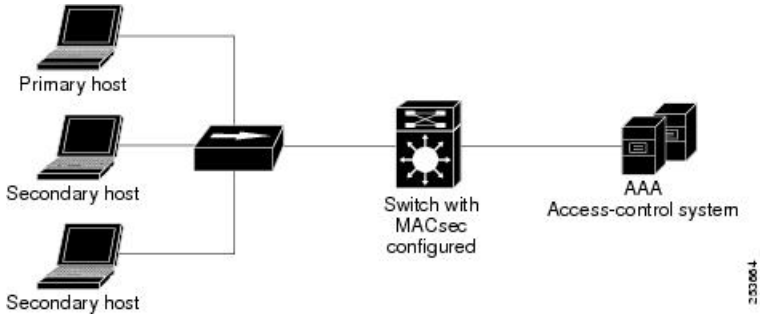
Figure 1: MACsec in Single-Host Mode with a Secured Data Session



Multiple Host Mode

In standard (not 802.1x REV) 802.1x multiple-host mode, a port is open or closed based on a single authentication. If one user, the primary secured client services client host, is authenticated, the same level of network access is provided to any host connected to the same port. If a secondary host is a MACsec supplicant, it cannot be authenticated and traffic would not flow. A secondary host that is a non-MACsec host can send traffic to the network without authentication because it is in multiple-host mode. The figure shows MACsec in Standard Multiple-Host Unsecure Mode.

Figure 2: MACsec in Multiple-Host Mode - Unsecured



Note Multi-host mode is not recommended because after the first successful client, authentication is not required for other clients, which is not secure.

In standard (not 802.1x REV) 802.1x multiple-domain mode, a port is open or closed based on a single authentication. If the primary user, a PC on data domain, is authenticated, the same level of network access is provided to any domain connected to the same port. If a secondary user is a MACsec supplicant, it cannot be authenticated and traffic would no flow. A secondary user, an IP phone on voice domain, that is a non-MACsec host, can send traffic to the network without authentication because it is in multiple-domain mode.

Multiple-Domain Mode

In standard (not 802.1x REV) 802.1x multiple-domain mode, a port is open or closed based on a single authentication. If the primary user, a PC on data domain, is authenticated, the same level of network access is provided to any domain connected to the same port. If a secondary user is a MACsec supplicant, it cannot be authenticated and traffic would no flow. A secondary user, an IP phone on voice domain, that is a non-MACsec host, can send traffic to the network without authentication because it is in multiple-domain mode.

MKA/MACsec for Port Channel

MKA/MACsec can be configured on the port members of a port channel. MKA/MACsec is agnostic to the port channel since the MKA session is established between the port members of a port channel.

Note Etherchannel links that are formed as part of the port channel can either be congruent or disparate i.e. the links can either be MACsec-secured or non-MACsec-secured. MKA session between the port members is established even if a port member on one side of the port channel is not configured with MACsec.

It is recommended that you enable MKA/MACsec on all the member ports for better security of the port channel.

How to Configure MACsec Encryption

Configuring MKA and MACsec

MACsec is disabled by default. No MKA policies are configured.

Configuring an MKA Policy

Procedure

	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	**mka policy** *policy name* **Example:** `Device(config)# mka policy mka_policy`	Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters. **Note** The default MACsec cipher suite in the MKA policy will always be GCM-AES-128.

	Command or Action	Purpose
Step 4	**key-server** *priority* **Example:** `Device(config-mka-policy)# key-server priority 200`	Configures MKA key server options and set priority (between 0-255). **Note** When value of key server priority is set to 255, the peer can not become the key server.
Step 5	**include-icv-indicator** **Example:** `Device(config-mka-policy)# include-icv-indicator`	Enables the ICV indicator in MKPDU. Use the **no** form of this command to disable the ICV indicator — **no include-icv-indicator**.
Step 6	**macsec-cipher-suite** *gcm-aes-128* **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128`	Configures cipher suite for deriving SAK with 128-bit encryption.
Step 7	**confidentiality-offset** *Offset value* **Example:** `Device(config-mka-policy)# confidentiality-offset 0`	Set the Confidentiality (encryption) offset for each physical interface **Note** Offset Value can be 0, 30 or 50. If you are using Anyconnect on the client, it is recommended to use Offset 0.
Step 8	**end** **Example:** `Device(config-mka-policy)# end`	Exit enters MKA policy configuration mode and returns to privileged EXEC mode.
Step 9	**show mka policy** **Example:** `Device# show mka policy`	Displays MKA policy configuration information.

Example

This example configures the MKA policy:

```
Switch(config)# mka policy mka_policy
Switch(config-mka-policy)# key-server priority 200
Switch(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Switch(config-mka-policy)# confidentiality-offset 30
Switch(config-mka-policy)# end
```

Configuring MACsec MKA using PSK

Procedure

	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	**key chain** *key-chain-name* **macsec** **Example:** `Device(config)# key chain keychain1 macsec`	Configures a key chain and enters the key chain configuration mode.
Step 4	**key** *hex-string* **Example:** `Device(config-key-chain)# key 1000`	Configures a unique identifier for each key in the keychain and enters the keychain's key configuration mode. **Note** For 128-bit encryption, use any value between 1 and 32 hex digit key-string. For 256-bit encryption, use 64 hex digit key-string.
Step 5	**key-string** *{ [0|6|7] pwd-string | pwd-string}* **Example:** `Device(config-key-chain)# key-string 12345678901234567890123456789012`	Sets the password for a key string. Only hex characters must be entered.
Step 6	**lifetime local** [*start timestamp {hh::mm::ss | day | month | year}*] [**duration** *seconds* | *end timestamp {hh::mm::ss | day | month | year}*] **Example:** `Device(config-key-chain)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016`	Sets the lifetime of the pre shared key.
Step 7	**end** **Example:** `Device(config-key-chain)# end`	Exits key chain configuration mode and returns to privileged EXEC mode.

Configuring MACsec MKA on an Interface using PSK

Procedure

	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-if)# interface GigabitEthernet 1/0/0`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.
Step 6	**mka pre-shared-key key-chain** *key-chain name* **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Configures an MKA pre-shared-key key-chain name. **Note** The MKA pre-shared key can be configured on either physical interface or sub-interfaces and not on both.
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 10`	Sets the MACsec window size for replay protection.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to do next

It is not recommended to change the MKA policy on an interface with MKA PSK configured when the session is running. However, if a change is required, you must reconfigure the policy as follows:

1. Disable the existing session by removing macsec network-link configuration on each of the participating node using the **no macsec network-link** command
2. Configure the MKA policy on the interface on each of the participating node using the **mka policy policy-name** command.
3. Enable the new session on each of the participating node by using the **macsec network-link** command.

Configuring MKA MACsec on the Switch-to-host Mode

To configure the MKA MACsec on Switch-to-host mode, perform these steps:

- Configure dot1x with the SANet including identity control policy.
- (Optionally) Configure identity control policy with linksec policy.
- (Optionally) Configure a MKA policy.
- Apply the macsec on the interface.
- (Optionally) Apply the configured mka policy on the interface
- Apply the configured identity control policy on the interface.

Enabling 802.1x Authentication and Configuring AAA

Procedure

	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	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication dot1x default group** *group-name* **Example:** `Device(config)# aaa authentication dot1x default group macsec-ise`	Sets the default authentication server group for IEEE 802.1x.
Step 5	**aaa authorization network default group** *group-name* **Example:**	Sets the network authorization default group.

	Command or Action	Purpose
	`Device(config)# aaa authentication dot1x default group macsec-ise`	
Step 6	**dot1x system-auth-control** **Example:** `Device(config)# dot1x system-auth-control`	Enables 802.1X on your device.
Step 7	**aaa group server** {**radius** | **tacacs+***group-name* **Example:** `Device(config)# aaa group server radius macsec-ise`	Specifies the name of the RADIUS server configuration for Protected Access Credential (PAC) provisioning and enters RADIUS server configuration mode.
Step 8	**server** *name* **Example:** `Device(config)# server name macsec`	Specifies the name of the server configuration for Protected Access Credential (PAC) provisioning and enters RADIUS server configuration mode.
Step 9	**address** *ip-address* **auth-port** *port-number* **acct-port** *port-number* **Example:** `Device(config)# address ipv4 <ise.ip> auth-port 1812 acct-port 1813`	Configures the IPv4 address for the RADIUS server accounting and authentication parameters.
Step 10	**key** *string* **Example:** `Device(config)# key cisco123`	Configures the authentication and encryption key for all RADIUS communications between the device and the RADIUS server.
Step 11	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 12	**event** *event name* [**match-all** | **match-first**] **Example:** `Device(config-event-control-policymap)# event session-started match-all`	Specifies the type of event that triggers actions in a control policy if conditions are met. • **match-all** is the default behavior. • To display the available event types, use the question mark (**?**) online help function. For a complete description of event types, see the **event** command.
Step 13	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:**	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.

	Command or Action	Purpose
	`Device(config-class-control-policymap)# 10 class always do-until-failure`	
Step 14	*action-number* **authenticate using** {**dot1x** | **mab** | **webauth**} [**aaa** {**authc-list** *authc-list-name* | **authz-list** *authz-list-name*]} [**merge**] [**parameter-map** *map-name*] [**priority** *priority-number*] [**replace** | **replace-all**] [**retries** *number* {**retry-time** *seconds*}] **Example:** `Device(config-action-control-policymap)# 10 authenticate using dot1x priority 10`	(Optional) Initiates the authentication of a subscriber session using the specified method.
Step 15	**exit**	Returns to global configuration mode.
Step 16	**interface** {*type* | *slot* | *port*} **Example:** `Device(config)# interface 1/10`	Specifies an interface to configure, and enters interface configuration mode.
Step 17	**switchport mode access vlan** *vlan id* **Example:** `Device(config-if)# switchport access vlan 17`	Specifies the VLAN for which this access port will carry traffic. If you do not enter this command, the access port carries traffic on VLAN1 only; use this command to change the VLAN for which the access port carries traffic..
Step 18	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Sets the interface as a nontrunking nontagged single-VLAN Ethernet interface. An access port can carry traffic in one VLAN only. By default, an access port carries traffic for VLAN1.
Step 19	**access-session closed** **Example:** `Device(config-if)# access-session closed`	Closes access to a port, preventing clients or devices from gaining network access before authentication is performed.
Step 20	**access-session port-control** {**auto** | **force-authorized** | **force-unauthorized**} **Example:** `Device(config-if)# access-session port-control auto`	Enables port-based authentication on the interface.
Step 21	**dot1x pae** [**supplicant** | **authenticator**] **Example:** `Device(config-if)# dot1x pae authenticator`	Enables port-based authentication on the interface. • **supplicant**—The interface acts only as a supplicant and does not respond to

	Command or Action	Purpose
		messages that are meant for an authenticator. • **authenticator**-—The interface acts only as an authenticator and does not respond to any messages meant for a supplicant. • **both**—The interface behaves both as a supplicant and as an authenticator and thus does respond to all dot1x messages.
Step 22	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 23	**exit**	Returns to global configuration mode.

Configuring Identity Control Policy with linksec Policy

Procedure

	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	**service-template** *template-name* **Example:** `Device(config)# service-template dot1x-macsec-policy`	Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode.
Step 4	**linksec policy** {**must-not-secure** | **must-secure** | **should-secure**} **Example:** `Device(config-service-template)# linksec policy must-secure`	Sets the link security policy as must-secure. • Must-secure policy authorizes the eEdge device port only if a secure MACsec session is established.
Step 5	**exit** **Example:** `Device(config-service-template)# exit`	Exits service template configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 6	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 7	**event authentication-success** [**match-all** | **match-any**] **Example:** `Device(config-event-control-policymap)# event authentication-success match-all`	Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode.
Step 8	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class always do-until-failure`	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.
Step 9	*action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]} **Example:** `Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy`	Activates a control policy on a subscriber session.
Step 10	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Configuring MACsec on Switch-to-switch Mode

To configure MACsec on Switch-to-switch mode, perform the following task:

- Configure a MACsec Pre-Shared Key.
- (Optionally) configure a MKA policy.
- Apply the MACsec on the interface.
- (Optionally) apply the configured MKA policy on the interface.
- Apply the configured MACsec Pre-Shared Key on the interface.

Configuring MKA Pre-shared Key

Perform the following task to configure MACsec Key Agreement (MKA) pre-shared key.

Procedure

	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	**key chain** *key-chain-name* [**macsec**] **Example:** `Device(config)# Key chain keychain1 macsec`	Configures a key chain and enters keychain configuration mode
Step 4	**key** *hex-string* **Example:** `Device(config-keychain)# key 9ABCD`	Configures a key and enters keychain key configuration mode. **Note** From Cisco IOS XE Everest Release 16.6.1 onwards, the Connectivity Association Key name (CKN) uses exactly the same string, which is configured as the hex-string for the key. For more information about this behavior change, see the section titled "MKA-PSK: CKN Behavior Change" after this task.
Step 5	**cryptographic-algorithm** {**gcm-aes-128** | **gcm-aes-256**} **Example:** `Device(config-keychain-key)# cryptographic-algorithm gcm-aes-128`	Set cryptographic authentication algorithm.
Step 6	**key-string** {[**0** | **6**] *pwd-string* | **7** | *pwd-string*} **Example:** `Device(config-keychain-key)# key-string 0 pwd`	Sets the password for a key string.
Step 7	**end** **Example:** `Device(config-keychain-key)# end`	Returns to privileged EXEC mode.

Configuring MKA

The MACsec Key Agreement (MKA) enables configuration and control of keying parameters. Perform the following task to configure MKA.

Procedure

	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	**mka policy** *policy-name* **Example:** `Device(config)# mka policy MKAPolicy`	Configures an MKA policy.
Step 4	**key-server priority** *key-server-priority* **Example:** `Device(config-mka-policy)# key-server priority 200`	(Optional) Configures MKA key server priority.
Step 5	**macsec-cipher-suite** {**gcm-aes-128** | **gcm-aes-256** | **gcm-aes-xpn-128** | **gcm-aes-xpn-256**} **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128 gcm-aes-256`	(Optional) Configures cipher suite(s) for secure association key (SAK) derivation. Each of the cipher suite options can be repeated only once, but they can be used in any order.
Step 6	**confidentiality-offset 30** **Example:** `Device(config-mka-policy)# confidentiality-offset 30`	(Optional) Configures confidentiality offset for MACsec operation.
Step 7	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-mka-policy)# end`	**Note** The MKA policy does not process confidentiality offset for XPN ciphers. Therefore when both XPN and non-XPN ciphers are configured in an MKA policy alongwith confidentiality offset, the confidentiality offset is ignored for XPN ciphers. It is therefore strongly recommended to use your discretion while using configuring a MKA policy with XPN or non-XPN ciphers.

Configuring MACsec and MKA on Interfaces

Perform the following task configure MACsec and MKA on an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface TenGigabitEthernet 1/0/0`	Enters interface configuration mode.
Step 4	**switchport mode { access | trunk }** **Example:** `Device(config-if)# switchport mode trunk }`	Sets the switchport mode to trunk.
Step 5	**macsec network-link** **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Enables MKA MACsec on the network link.
Step 6	**mka policy** *policy-name* **Example:**	Configures an MKA policy.

	Command or Action	Purpose
	`Device(config)# mka policy MKAPolicy`	
Step 7	**mka pre-shared-key key-chain** *key-chain-name* **Example:** `Device(config)# mka pre-shared-key key-chain k10`	Configures an MKA pre-shared-key key-chain 10.
Step 8	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring MKA/MACsec for Port Channel using PSK

Procedure

	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-if)# interface gigabitethernet 1/0/3`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface. Supports layer 2 and layer 3 port channels.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.
Step 6	**mka pre-shared-key key-chain** *key-chain-name* **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Configures an MKA pre-shared-key key-chain name. **Note** The MKA pre-shared key can be configured on either physical interface or sub-interfaces and not on both.

	Command or Action	Purpose
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 0`	Sets the MACsec window size for replay protection.
Step 8	**channel-group** *channel-group-number* **mode** {**auto** | **desirable**} | {**active** | **passive**} | {**on**} **Example:** `Device(config-if)# channel-group 3 mode auto active on`	Configures the port in a channel group and sets the mode. **Note** You cannot configure ports in a channel group without configuring MACsec on the interface. You must configure the commands in Step 3, 4, 5 and 6 before this step. The channel-number range is from 1 to 4096. The port channel associated with this channel group is automatically created if the port channel does not already exist.For mode, select one of the following keywords: • **auto** — Enables PAgP only if a PAgP device is detected. This places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. **Note** The **auto** keyword is not supported when EtherChannel members are from different switches in the switch stack. • **desirable** — Unconditionally enables PAgP. This places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. **Note** The **desirable** keyword is not supported when EtherChannel members are from different switches in the switch stack. • **on** — Forces the port to channel without PAgP or LACP. In the on mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **active** — Enables LACP only if a LACP device is detected. It places the port into

	Command or Action	Purpose
		an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** — Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 9	**end** **Example:** `Device(config-if)# cend`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel Logical Interfaces for Layer 2 EtherChannels

To create a port channel interface for a Layer 2 EtherChannel, perform this task:

Procedure

	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 port-channel** *channel-group-number* **Example:** `Device(config)# interface port-channel 1`	Creates the port channel interface. **Note** Use the **no** form of this command to delete the port channel interface.
Step 4	**switchport** **Example:** `Device(config-if)# switchport`	Switches an interface that is in Layer 3 mode into Layer 2 mode for Layer 2 configuration.
Step 5	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Examples for MACsec Encryption

Example: Configuring MKA and MACsec

This example shows how to create an MKA policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server priority 200
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 30
Device(config-mka-policy)# ssci-based-on-sci
Device(config-mka-policy)#end
```

This example shows how to configure downlink MACsec on an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# switchport access vlan 17
Device(config-if)# switchport mode access
Device(config-if)# macsec
Device(config-if)# access-session host-mode single-host
Device(config-if)# access-session closed
Device(config-if)# access-session port-control auto
Device(config-if)#mka policy mka_policy
Device(config-if)# dot1x pae authenticator
Device(config-if)#service-policy type control subscriber POLICY_SHOULDSECURE
Device(config-if)#end
```

Examples: Configuring MACsec MKA using PSK

This example shows how to configure MACsec MKA using PSK.

```
Device> enable
Device# configure terminal
Device(config)# Key chain keychain1 macsec
Device(config-key-chain)# key 1000
Device(config-keychain-key)# cryptographic-algorithm gcm-aes-128
Device(config-keychain-key)# key-string 12345678901234567890123456789012
Device(config-keychain-key)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016
Device(config-keychain-key)# end
```

This example shows how to configure uplink MACsec MKA on an interface using PSK.

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# mka policy mka_policy
Device(config-if)# mka pre-shared-key key-chain key-chain-name
Device(config-if)# macsec replay-protection window-size 10
Device(config-if)# end
```

Example: Configuring MACsec MKA for Port Channel using PSK

Etherchannel Mode — Static/On

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode on:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

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

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
```

```
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)           -        Te1/0/1(P)  Te1/0/2(P)
```

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

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)           -        Te1/0/1(P)  Te1/0/2(P)
```

Etherchannel Mode — LACP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as LACP.

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
```

```
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

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

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1
```

```
------+------------+-----------+-----------------------------------------------

 2      Po2(SU)          LACP      Te1/1/1(P)  Te1/1/2(P)
```

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

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+------------+-----------+-----------------------------------------------

 2      Po2(RU)          LACP      Te1/1/1(P)  Te1/1/2(P)
```

Etherchannel Mode — PAgP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as PAgP:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
```

```
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following shows a sample output from the **show etherchannel summary** command.

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1


------+-------------+-----------+-----------------------------------------------

2      Po2(SU)          PAgP      Te1/1/1(P)  Te1/1/2(P)
```

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

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
```

```
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          PAgP      Te1/1/1(P)   Te1/1/2(P)
```

Displaying Active MKA Sessions

The following shows all the active MKA sessions.

```
Device# show mka sessions interface Te1/0/1

=======================================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status             CKN
=======================================================================================
Te1/0/1        00a3.d144.3364/0025 POLICY           NO                  NO

37             701f.539b.b0c6/0032 1                Secured
1000
```

Example: Displaying MKA Information

The following is sample output from the **show mka sessions** command.

```
Device# show mka sessions


Total MKA Sessions..... 1
      Secured Sessions... 1
      Pending Sessions... 0

=======================================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status             CKN
=======================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                  YES
43             c800.8459.e764/002a 1                Secured
01000000000000000000000000000000000000000000000000000000000000000000
```

The following is sample output from the **show mka sessions interface** *interface-name* command.

```
Device# show mka sessions interface GigabitEthernet 1/0/1

Summary of All Currently Active MKA Sessions on Interface
GigabitEthernet1/0/1...

===============================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status             CKN
===============================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                 YES
43             c800.8459.e764/002a 1                Secured
01000000000000000000000000000000000000000000000000000000000000000000
```

The following is sample output from the **show mka sessions interface** *interface-name* **detail**command.

```
Device# show mka sessions interface GigabitEthernet 1/0/1 detail


MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI..... 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
01000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
```

```
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output from the **show mka sessions details** command:

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
```

```
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is sample output from the **show mka policy** command:

```
Device# show mka policy


MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher
 Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)
 Applied
================================================================================
*DEFAULT POLICY*  0        FALSE   TRUE    0      0      GCM-AES-128

p1                1        FALSE   TRUE    0      0      GCM-AES-128

p2                2        FALSE   TRUE    0      0      GCM-AES-128
 Gi1/0/1
```

The following is sample output from the **show mka policy** *policy-name* command:

```
Device# show mka policy p2


MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher
```

```
 Interfaces
Name               Priority Protect Protect Size   Offset Suite(s)
 Applied
================================================================================
p2                 2        FALSE   TRUE    0      0      GCM-AES-128
 Gi1/0/1
```

The following is sample output from the **show mka policy** *policy-name* **detail** command:

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
=======================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128

Applied Interfaces...
  GigabitEthernet1/0/1
```

The following is sample output from the **show mka statistics interface** *interface-name* command:

```
Device# show mka statistics interface GigabitEthernet 1/0/1


MKA Statistics for Session
==========================
Reauthentication Attempts.. 0

CA Statistics
   Pairwise CAKs Derived... 0
   Pairwise CAK Rekeys..... 0
   Group CAKs Generated.... 0
   Group CAKs Received..... 0

SA Statistics
   SAKs Generated.......... 1
   SAKs Rekeyed............ 0
   SAKs Received........... 0
   SAK Responses Received.. 1

MKPDU Statistics
   MKPDUs Validated & Rx... 89585
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Transmitted...... 89596
      "Distributed SAK".. 1
      "Distributed CAK".. 0
```

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

```
Device# show mka summary

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

================================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status             CKN
================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                 YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000



MKA Global Statistics
=====================
MKA Session Totals
   Secured..................... 1
   Reauthentication Attempts.. 0

   Deleted (Secured).......... 0
   Keepalive Timeouts......... 0

CA Statistics
   Pairwise CAKs Derived...... 0
   Pairwise CAK Rekeys........ 0
   Group CAKs Generated....... 0
   Group CAKs Received........ 0

SA Statistics
   SAKs Generated............. 1
   SAKs Rekeyed............... 0
   SAKs Received.............. 0
   SAK Responses Received..... 1

MKPDU Statistics
   MKPDUs Validated & Rx...... 89589
      "Distributed SAK"..... 0
      "Distributed CAK"..... 0
   MKPDUs Transmitted......... 89600
      "Distributed SAK"..... 1
      "Distributed CAK"..... 0

MKA Error Counter Totals
========================
Session Failures
   Bring-up Failures.............. 0
   Reauthentication Failures........ 0
   Duplicate Auth-Mgr Handle........ 0
```

```
SAK Failures
   SAK Generation..................... 0
   Hash Key Generation................ 0
   SAK Encryption/Wrap................ 0
   SAK Decryption/Unwrap.............. 0
   SAK Cipher Mismatch................ 0

CA Failures
   Group CAK Generation.............. 0
   Group CAK Encryption/Wrap........ 0
   Group CAK Decryption/Unwrap...... 0
   Pairwise CAK Derivation.......... 0
   CKN Derivation..................... 0
   ICK Derivation..................... 0
   KEK Derivation..................... 0
   Invalid Peer MACsec Capability... 0
MACsec Failures
   Rx SC Creation..................... 0
   Tx SC Creation..................... 0
   Rx SA Installation................. 0
   Tx SA Installation................. 0

MKPDU Failures
   MKPDU Tx........................... 0
   MKPDU Rx Validation................ 0
   MKPDU Rx Bad Peer MN............... 0
   MKPDU Rx Non-recent Peerlist MN.. 0
```

IPv6 First Hop Security Overview

First Hop Security in IPv6 (FHS IPv6) is a set of IPv6 security features, whose policies can be attached to a physical interface, an EtherChannel interface, or a VLAN. An IPv6 software policy database service stores and accesses these policies. When a policy is configured or modified, the attributes of the policy are stored or updated in the software policy database, and applied as specified. The following IPv6 policies are currently supported:

- Manual IPv6 Binding—Creates static IPv6 binding for secure network.
- IPv6 Address Glean/Inspect/Guard—Allows to build dynamic binding table by NDP and DHCPv6 glean. Also, inspects control packets to prevent unauthorized messages by rogue host, and guard unauthorized RA and DHCP server messages.
- IPv6 Device Tracking—IPv6 Device Tracking allows to track the presence, location, and movement of end-nodes in the network. SISF snoops traffic received by the switch ports, extracts device identity (MAC and IP address), and stores them in a binding table. Many features, such as, Cisco TrustSec, IEEE 802.1X, LISP, and web authentication depend on the accuracy of this information to operate properly.
- IPv6 FHS Binding Recory—IPv6 binding address recovery allows to recover binding table from a complete failure of the router. When the traffic is received from a an unknow source that is not in the binding table, IPv6 FHS Binding Recory feature helps to rebuild binding table based on IPv6 address glean by NDP or DHCPv6 recovery.

- IPv6 Source Guard—Like IPv4 Source Guard, IPv6 Source Guard validates the source address or prefix to prevent source address spoofing.

  A source guard programs the hardware to allow or deny traffic based on source or destination addresses. It deals exclusively with data packet traffic.

  The IPv6 source guard feature provides the ability to store entries in the hardware TCAM table to prevent a host from sending packets with an invalid IPv6 source address.

  To debug source-guard packets, use the **debug device-tracking source-guard** privileged EXEC command.

  Note The IPv6 Source Guard feature is supported only in the ingress direction and not supported in the egress direction. The IPv6 Prefix Guard is not supported.

- IPv6 DHCP Guard—The IPv6 DHCP Guard feature blocks reply and advertisement messages that come from unauthorized DHCPv6 servers and relay agents. IPv6 DHCP guard can prevent forged messages from being entered in the binding table and block DHCPv6 server messages when they are received on ports that are not explicitly configured as facing a DHCPv6 server or DHCP relay. To use this feature, configure a policy and attach it to an interface or a VLAN.
- IPv6 Router Advertisement Guard—The IPv6 Router Advertisement (RA) guard feature enables the network administrator to block or reject unwanted or rogue RA guard messages that arrive at the network device platform. RAs are used by devices to announce themselves on the link. The RA Guard feature analyzes the RAs and filters out bogus RAs sent by unauthorized devices. In host mode, all router advertisement and router redirect messages are disallowed on the port. The RA guard feature compares configuration information on the Layer 2 device with the information found in the received RA frame. Once the Layer 2 device has validated the content of the RA frame and router redirect frame against the configuration, it forwards the RA to its unicast or multicast destination. If the RA frame content is not validated, the RA is dropped.

Configuring the Manual IPv6 Binding

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Binding Table Content :

Procedure

	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	**device-tracking binding vlan** *vlan-id* {*ipv6-address* **interface** interface {*mac_address*} [**tracking**{ [default | disable] [**reachable-lifetimevalue** [*seconds* | **default** | **infinite**] | [**enable** [**reachable-lifetimevalue** [*seconds* | **default** | **infinite**] } **Example:** `Device(config)# decive-tracking binding`	Adds a static entry to the binding table database.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show device-tracking binding** **Example:** `Device# show device-tracking binding`	Displays contents of a binding table.

Configuring the IPv6 Binding Recovery

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Binding Recovery:

Procedure

	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-tracking policy** *policy-name* **Example:** `Device(config)# device-tracking policy example_policy`	Creates a device tracking policy and enters IPv6 device-tracking policy configuration mode.
Step 4	**data-glean recovery {dhcp | ndp [dhcp] }** **Example:** `Device(config-device-tracking)# data-glean recovery dhcp`	Enables data address gleaning, validates messages against various criteria, specifies the security level for messages.

	Command or Action	Purpose
Step 5	**data-glean log-only** **Example:** `Device(config-device-tracking)# data-glean log-only`	Enables IPv6 first-hop security binding table recovery using source (or "data") address gleaning.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring an IPv6 Neighbor Discovery Inspection Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 ND Inspection Policy:

Procedure

	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-tracking policy** *policy-name* **Example:** `Device(config)# device-tracking policy example_policy`	Creates the policy and enters the device-tracking configuration mode.
Step 4	**security-level inspect** **Example:** `Device(config-device-tracking)# security-level inspect`	Specifies the level of security enforced by the feature.
Step 5	**device-role** {**host** | **switch**} **Example:** `Device(config-device-tracking)# device-role switch`	Specifies the role of the device attached to the port. The default is **host**.
Step 6	**limit address-count** *value* **Example:** `Device(config-device-tracking)# limit address-count 1000`	Limits the number of IPv6 addresses allowed to be used on the port.

	Command or Action	Purpose
Step 7	**trusted-port** **Example:** `Device(config-device-tracking)# trusted-port`	Configures a port to become a trusted port.
Step 8	**end** **Example:** `Device(config-device-tracking)# end`	Exits ND Inspection Policy configuration mode and returns to privileged EXEC mode.
Step 9	**show device-tracking policy** *example_policy* **Example:** `Device# show device-tracking policy example_policy`	Verifies the device-tracking inspection configuration.

Configuring an IPv6 Device Tracking Policy

Note The IPv6 Snooping Policy feature has been deprecated. Although the commands are visible on the CLI and you can configure them, we recommend that you use the Switch Integrated Security Feature (SISF)-based Device Tracking feature instead.

Beginning in privileged EXEC mode, follow these steps to configure device tracking policy :

Procedure

	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-tracking policy** *policy-name* **Example:** `Device(config)# device-tracking policy example_policy`	Creates a device tracking policy and enters IPv4 or IPv6 device-tracking policy configuration mode.

	Command or Action	Purpose

Step 4	{[**default**] | [**device-role** {**node** | **switch**}] | [**limit address-count** *value*] | [**no**] | [**protocol** {**dhcp** | **dhcp 6** | **arp ndp**}] | [**security-level** {**glean** | **guard** | **inspect**}] | [**tracking** {**disable** [**stale-lifetime** [*seconds* | **infinite**] | **enable** [**reachable-lifetime** [*seconds* | **infinite**] }] | [**trusted-port**] }
	Example:
	`Device(config-device-tracking)# security-level inspect`
	Example:
	`Device(config-device-tracking policy )# trusted-port`
		Enables data address gleaning, validates messages against various criteria, specifies the security level for messages.
		• (Optional) **default**—Sets all to default options.
		• (Optional) **device-role**{**node**] | **switch**}—Specifies the role of the device attached to the port. Default is **node**.
		• (Optional) **limit address-count** *value*—Limits the number of addresses allowed per target.
		• (Optional) **no**—Negates a command or sets it to defaults.
		• (Optional) **protocol**{**dhcp** | **ndp**}—Specifies which protocol should be redirected to the snooping feature for analysis. The default, is **dhcp** and **ndp**. To change the default, use the **no protocol** command.
		• (Optional) **security-level**{**glean**|**guard**|**inspect**}—Specifies the level of security enforced by the feature. Default is **guard.**
		glean—Gleans addresses from messages and populates the binding table without any verification.
		guard—Gleans addresses and inspects messages. In addition, it rejects RA and DHCP server messages. This is the default option.
		inspect—Gleans addresses, validates messages for consistency and conformance, and enforces address ownership.
		• (Optional) **tracking** {**disable** | **enable**}—Overrides the default tracking behavior and specifies a tracking option.
		• (Optional) **trusted-port**—Sets up a trusted port. It disables the guard on applicable targets. Bindings learned through a trusted port have preference over bindings learned through any other port. A trusted port is given preference in case of a collision while making an entry in the table.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-device-tracking policy )# end`	Exits IPv6 snooping policy configuration mode and returns to privileged EXEC mode.
Step 6	**show device-tracking policy** *policy-name* **Example:** `Device#show device-tracking policy example_policy`	Displays the device-tracking policy policy configuration.

What to do next

Attach an IPv6 device-tracking policy to interfaces or VLANs.

Attaching an IPv6 Device Tracking Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 device tracking policy on an interface or VLAN:

Procedure

	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_type stack/module/port* **Example:** `Device(config)# interface gigabitethernet 1/1/4`	Specifies an interface type and identifier and enters the interface configuration mode.
Step 4	**device-tracking** [**attach-policy** *policy_name* [**vlan** {*vlan_id* | **add** *vlan_ids* | **except** *vlan_ids* | **none** | **remove** *vlan_ids*}] | **vlan** {*vlan_id* | **add** *vlan_ids* | **except** *vlan_ids* | **none** | **remove** *vlan_ids* | **all**}] **Example:** `Device(config-if)# device-traking attach-policy example_policy`	Attaches a custom IPv6 snooping policy to the interface or the specified VLANs on the interface. To attach the default policy to the interface, use the **device-traking** command without the **attach-policy** keyword. To attach the default policy to VLANs on the interface, use the **device-traking vlan** command. The default policy is, security-level **guard**, device-role **node**, protocol **ndp** and **dhcp.**

	Command or Action	Purpose

	`Device(config-if)# device-traking vlan 111,112`
	
	`Device(config-if)# device-traking attach-policy example_policy vlan 111,112`	

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies that the policy is attached to the specified interface without exiting the interface configuration mode.

Attaching an IPv6 Device Tracking Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 device-tracing policy to VLANs across multiple interfaces:

Procedure

	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	**vlan configuration** *vlan_list* **Example:** `Device(config)# vlan configuration 333`	Specifies the VLANs to which the IPv6 Snooping policy will be attached, and enters the VLAN interface configuration mode.
Step 4	**device-traking** [**attach-policy** *policy_name*] **Example:** `Device(config-vlan-config)#device-tracking attach-policy example_policy`	Attaches the IPv6 Snooping policy to the specified VLANs across all device interfaces. The default policy is attached if the **attach-policy** option is not used. The default policy is, security-level **guard**, device-role **node**, protocol **ndp** and **dhcp.**
Step 5	**end** **Example:**	Exits VLAN interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-vlan-config)# end`	

Configuring IPv6 Source Guard

Procedure

	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	**ipv6 source-guard policy** *policy_name* **Example:** `Device(config)# ipv6 source-guard policy example_policy`	Specifies the IPv6 Source Guard policy name and enters IPv6 Source Guard policy configuration mode.
Step 4	**validate address** **Example:** `Device(config-sisf-sourceguard)# validate address`	Enables the validate address feature. This feature does not support the validate prefix and no validate options.
Step 5	**end** **Example:** `Device(config-sisf-sourceguard)# end`	Exits of IPv6 Source Guard policy configuration mode and returns to privileged EXEC mode.
Step 6	**show ipv6 source-guard policy** *policy_name* **Example:** `Device# show ipv6 source-guard policy example_policy`	Shows the policy configuration and all the interfaces where the policy is applied.

What to do next

Apply the IPv6 Source Guard policy to an interface.

Attaching an IPv6 Source Guard Policy to an Interface

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1/4`	Specifies an interface type and identifier; enters interface configuration mode.
Step 4	**ipv6 source-guard** [**attach-policy** <*policy_name*>] **Example:** `Device(config-if)# ipv6 source-guard attach-policy example_policy`	Attaches the IPv6 Source Guard policy to the interface. The default policy is attached if the **attach-policy** option is not used.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show ipv6 source-guard policy** *policy_name* **Example:** `Device#(config)# show ipv6 source-guard policy example_policy`	Shows the policy configuration and all the interfaces where the policy is applied.

Configuring an IPv6 DHCP Guard Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 DHCP (DHCPv6) Guard policy:

Procedure

	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	**ipv6 dhcp guard policy** *policy-name* **Example:** `Device(config)# ipv6 dhcp guard policy example_policy`	Specifies the DHCPv6 Guard policy name and enters DHCPv6 Guard Policy configuration mode.
Step 4	**device-role** {**client** | **monitor** |**server**} **Example:** `Device(config-dhcp-guard)# device-role server`	(Optional) Filters out DHCPv6 replies and DHCPv6 advertisements on the port that are not from a device of the specified role. Default is **client**. • **client**—Default value, specifies that the attached device is a client. Server messages are dropped on this port. • **server**—Specifies that the attached device is a DHCPv6 server. Server messages are allowed on this port.
Step 5	**trusted-port** **Example:** `Device(config-dhcp-guard)# trusted-port`	(Optional) **trusted-port**—Sets the port to a trusted mode. No further policing takes place on the port. **Note** If you configure a trusted port then the device-role option is not available.
Step 6	**end** **Example:** `Device(config-dhcp-guard)# end`	Exits DHCPv6 Guard Policy configuration mode and returns to privileged EXEC mode.
Step 7	**show ipv6 dhcp guard policy** *policy_name* **Example:** `Device# show ipv6 dhcp guard policy example_policy`	(Optional) Displays the configuration of the IPv6 DHCP guard policy. Omitting the *policy_name* variable displays all DHCPv6 policies.

Attaching an IPv6 DHCP Guard Policy to an Interface or a VLAN on an Interface

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Binding Table Content :

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1/4`	Specifies an interface type and identifier, and enters interface configuration mode.

Step 4	**ipv6 dhcp guard** [**attach-policy** *policy_name* [**vlan** {*vlan_ids* | **add** *vlan_ids* | **except** *vlan_ids* | **none** | **remove** *vlan_ids* | **all**}] | **vlan** [{*vlan_ids* | **add** *vlan_ids* | **except***vlan_ids* | **none** | **remove** *vlan_ids* | **all**}]
	Example:
	`Device(config-if)# ipv6 dhcp guard attach-policy example_policy`
	`Device(config-if)# ipv6 dhcp guard attach-policy example_policy vlan 222,223,224`
	`Device(config-if)# ipv6 dhcp guard vlan 222, 223,224`
		Attaches the DHCP Guard policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Attaching an IPv6 DHCP Guard Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 DHCP Guard policy to VLANs across multiple interfaces:

Procedure

	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	**vlan configuration** *vlan_list* **Example:** `Device(config)# vlan configuration 334`	Specifies the VLANs to which the IPv6 Snooping policy will be attached, and enters VLAN interface configuration mode.
Step 4	**ipv6 dhcp guard** [**attach-policy** *policy_name*] **Example:** `Device(config-vlan-config)#ipv6 dhcp guard attach-policy example_policy`	Attaches the IPv6 Neighbor Discovery policy to the specified VLANs across all switch and stack interfaces. The default policy is attached if the **attach-policy** option is not used. The default policy is, device-role **client**, **no** trusted-port.
Step 5	**end** **Example:** `Device(config-vlan-config)# end`	Exits VLAN interface configuration mode and returns to privileged EXEC mode.

Configuring an IPv6 Router Advertisement Guard Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 Router Advertisement policy :

Procedure

	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	**ipv6 nd raguard policy** *policy-name* **Example:** `Device(config)# ipv6 nd raguard policy example_policy`	Specifies the RA guard policy name and enters RA guard policy configuration mode.
Step 4	[**no**]**device-role** {**host** | **monitor** | **router** | **switch**} **Example:**	Specifies the role of the device attached to the port. The default is **host**.

	Command or Action	Purpose
	`Device(config-nd-raguard)# device-role switch`	**Note** For a network with both host-facing ports and router-facing ports, along with a RA guard policy configured with **device-role host** on host-facing ports or vlan, it is mandatory to configure a RA guard policy with **device-role router** on router-facing ports to allow the RA Guard feature to work properly.
Step 5	**hop-limit** {**maximum** | **minimum**} *value* **Example:** `Device(config-nd-raguard)# hop-limit maximum 33`	Enables filtering of Router Advertisement messages by the Hop Limit value. A rogue RA message may have a low Hop Limit value (equivalent to the IPv4 Time to Live) that when accepted by the host, prevents the host from generating traffic to destinations beyond the rogue RA message generator. An RA message with an unspecified Hop Limit value is blocked. (1–255) Range for Maximum and Minimum Hop Limit values. If not configured, this filter is disabled. Configure **minimum** to block RA messages with Hop Limit values lower than the value you specify. Configure **maximum**to block RA messages with Hop Limit values greater than the value you specify.
Step 6	**managed-config-flag** {**off** | **on**} **Example:** `Device(config-nd-raguard)# managed-config-flag on`	Enables filtering of Router Advertisement messages by the managed address configuration, or "M" flag field. A rouge RA message with an M field of 1 can cause a host to use a rogue DHCPv6 server. If not configured, this filter is disabled. **On**—Accepts and forwards RA messages with an M value of 1, blocks those with 0. **Off**—Accepts and forwards RA messages with an M value of 0, blocks those with 1.
Step 7	**match** {**ipv6 access-list** *list* | **ra prefix-list** *list*} **Example:** `Device(config-nd-raguard)# match ipv6 access-list example_list`	Matches a specified prefix list or access list.

	Command or Action	Purpose
Step 8	**router-preference maximum** {**high** | **medium** | **low**} **Example:** `Device(config-nd-raguard)# router-preference maximum high`	Enables filtering of Router Advertisement messages by the router preference flag. If not configured, this filter is disabled. • **high**—Accepts RA messages with the router preference set to high, medium, or low. • **medium**—Blocks RA messages with the router preference set to high. • **low**—Blocks RA messages with the router preference set to medium and high.
Step 9	**trusted-port** **Example:** `Device(config-nd-raguard)# trusted-port`	When configured as a trusted port, all attached devices are trusted, and no further message verification is performed.
Step 10	**end** **Example:** `Device(config-nd-raguard)# end`	Exits RA Guard policy configuration mode and returns to privileged EXEC mode.
Step 11	**show ipv6 nd raguard policy** *policy_name* **Example:** `Device# show ipv6 nd raguard policy example_policy`	(Optional)—Displays the ND guard policy configuration.

Attaching an IPv6 Router Advertisement Guard Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Router Advertisement policy to an interface or to VLANs on the interface :

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Specifies an interface type and identifier; enters the interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1/4`	

Step 4	**ipv6 nd raguard** [**attach-policy** *policy_name* [**vlan** {*vlan_ids* | **add** *vlan_ids* | **except** *vlan_ids* | **none** | **remove** *vlan_ids* | **all**}] | **vlan** [{*vlan_ids* | **add** *vlan_ids* | **except***vlan_ids* | **none** | **remove** *vlan_ids* | **all**}]
	Example:
	`Device(config-if)# ipv6 nd raguard attach-policy example_policy`
	`Device(config-if)# ipv6 nd raguard attach-policy example_policy vlan 222,223,224`
	`Device(config-if)# ipv6 nd raguard vlan 222, 223,224`
		Attaches the Neighbor Discovery Inspection policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Attaching an IPv6 Router Advertisement Guard Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Router Advertisement policy to VLANs regardless of interface:

Procedure

	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	**vlan configuration** *vlan_list* **Example:** `Device(config)# vlan configuration 335`	Specifies the VLANs to which the IPv6 RA Guard policy will be attached, and enters VLAN interface configuration mode.
Step 4	**ipv6 dhcp guard** [**attach-policy** *policy_name*] **Example:**	Attaches the IPv6 RA Guard policy to the specified VLANs across all switch and stack interfaces. The default policy is attached if the **attach-policy** option is not used.

	Command or Action	Purpose
	`Device(config-vlan-config)#ipv6 nd raguard attach-policy example_policy`	
Step 5	**end** **Example:** `Device(config-vlan-config)# end`	Exits VLAN interface configuration mode and returns to privileged EXEC mode.

Information About Dynamic ARP Inspection

Dynamic ARP Inspection (DAI) is a security feature that validates Address Resolution Protocol (ARP) packets in a network. DAI allows a network administrator to intercept, log, and discard ARP packets with invalid MAC address to IP address bindings. This capability protects the network from certain “man-in-the-middle” attacks.

To prevent ARP poisoning attacks such as the one described in the previous section, a device must ensure that only valid ARP requests and responses are relayed. DAI prevents these attacks by intercepting all ARP requests and responses. Each of these intercepted packets is verified for valid MAC address to IP address bindings before the local ARP cache is updated or the packet is forwarded to the appropriate destination. Invalid ARP packets are dropped.

DAI determines the validity of an ARP packet based on valid MAC address to IP address bindings stored in a trusted database. This database is built at runtime by DHCP snooping, provided that it is enabled on the VLANs and on the device in question. In addition, DAI can also validate ARP packets against user-configured ARP ACLs in order to handle hosts that use statically configured IP addresses.

DAI can also be configured to drop ARP packets when the IP addresses in the packet are invalid or when the MAC addresses in the body of the ARP packet do not match the addresses specified in the Ethernet header.

Configuring Dynamic ARP Inspection

Dynamic ARP inspection intercepts, logs, and discards ARP packets with invalid IP-to-MAC address bindings. You can configure the devic to perform additional checks on the destination MAC address, the sender and target IP addresses, and the source MAC address.

Follow these steps to configure dynamic ARP inspection.

Procedure

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

	Command or Action	Purpose
Step 3	**ip arp inspection vlan {vlan_ID | vlan_range}** **Example:** `Device(config)# ip arp inspection vlan 1`	Enables DAI on VLANs (disabled by default).
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface fastEthernet 3/3`	Specify the Switch A interface that is connected to Switch B, and enter interface configuration mode. For untrusted interfaces, the device intercepts all ARP requests and responses. It verifies that the intercepted packets have valid IP-to-MAC address bindings before updating the local cache and before forwarding the packet to the appropriate destination. The device drops invalid packets and logs them in the log buffer according to the logging configuration specified with the **ip arp inspection vlan logging** global configuration command.
Step 5	**ip arp inspection trust** **Example:** `Device(config-if)# ip arp inspection trust`	Configures the connection between switches.
Step 6	**ip arp inspection filter arp_acl_name vlan {vlan_ID | vlan_range} [static]** **Example:** `Device(config-if)# ip arp inspection filter test vlan 1`	Applies the ARP ACL to a VLAN Apply the ARP ACL to the VLAN. By default, no defined ARP ACLs are applied to any VLAN. • For arp-acl-name, specify the name of the ACL. • For vlan-range, specify the VLAN that the switches and hosts are in. You can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • (Optional) Specify **static** to treat implicit denies in the ARP ACL as explicit denies and to drop packets that do not match any previous clauses in the ACL. DHCP bindings are not used. If you do not specify this keyword, it means that there is no explicit deny in the ACL that denies the packet, and DHCP bindings determine

	Command or Action	Purpose
		whether a packet is permitted or denied if the packet does not match any clauses in the ACL.
Step 7	**ip arp inspection limit {rate pps [burst interval seconds] | none}** **Example:** `Device(config-if)# ip arp inspection limit rate pps 1`	Limits the rate of incoming ARP requests and responses on the interface. The default rate is 15 pps on untrusted interfaces and unlimited on trusted interfaces. The burst interval is 1 second.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 9	**errdisable recovery cause arp-inspection**	(Optional) Enables error recovery from the dynamic ARP inspection error-disabled state, and configure the dynamic ARPinspection recover mechanism variables.
Step 10	**ip arp inspection validate {[src-mac] [dst-mac] [ip]}** **Example:** `Device(config)# ip inspection validate ip`	Performs a specific check on incoming ARP packets. By default, no checks are performed. The keywords have these meanings: • For **src-mac**, check the source MAC address in the Ethernet header against the sender MAC address in the ARP body. This check is performed on both ARP requests and responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped. • For **dst-mac**, check the destination MAC address in the Ethernet header against the target MAC address in ARP body. This check is performed for ARP responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped. • For **ip**, check the ARP body for invalid and unexpected IP addresses. Addresses include 0.0.0.0, 255.255.255.255, and all IP multicast addresses. Sender IP addresses are checked in all ARP requests and responses, and target IP addresses are checked only in ARP responses. You must specify at least one of the keywords. Each command overrides the configuration of the previous command; that is, if a command

	Command or Action	Purpose
		enables src and dst mac validations, and a second command enables IP validation only, the src and dst mac validations are disabled as a result of the second command.
Step 11	**ip arp inspection log-buffer entries number**	Configures the DAI logging buffer size (range is 0 to 1024).
Step 12	**ip arp inspection log-buffer logs number_of_messages interval length_in_seconds**	Configures the DAI logging buffer.
Step 13	**ip arp inspection vlan vlan_range logging {acl-match {matchlog | none} | dhcp-bindings {all | none | permit}}**	Configures log filtering for each VLAN.
Step 14	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 15	**show ip arp inspection vlan** *vlan-range* **Example:** `Device# show ip arp insepction vlan 1-2`	Displays the statistics for the selected range of VLANs.

Information about InterfaceTemplate

An interface template provides a mechanism to configure multiple commands at the same time and associate it with a target such as an interface. An interface template is a container of configurations or policies that can be applied to specific ports.

Interface Templates provide an efficient way to apply ACLs along with other commands on interfaces. ACLs can be applied on an interface by first configuring an ACL inside an interface template, and then applying the template to any number of desired interfaces. A single template having an ACL can be applied to any number of physical or virtual interfaces.

Note Interface Template is not supported on SVI or EtherChannel.

Configuring Interface Template

To configure an interface template, follow these steps.

Procedure

	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	**template <name>** **Example:** `Device(config)# template test`	Pls provide the inputs.
Step 4	**ip access-group <acl> in | out** **Example:** `Device(config-template)# ip access-group <acl> in | out`	Applies the specified IPv6 access list to the template.
Step 5	**ipv6 traffic-filter <acl> in | out** **Example:** `Device(config-template)# ip access-group <acl> in | out`	Applies the specified IPv6 access list to the interface specified in the previous step.
Step 6	**source template***template name* **Example:** `Device(config-if)# source template test`	Pls provide the inputs.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Information about Time Domain Reflectometer

Time Domain Reflectometry is a technique used to analyze a conductor by transmitting into it a pulsed signal and then by examining the polarity, amplitude and round trip time of the reflected waveform.

By estimating the speed of propagation of the signal in the specific transmission medium and by measuring the time it takes for its reflection to travel back to the source it is possible to measure the distance of the reflecting point from the cable tester. Also, by comparing the polarity and amplitude of the original pulse with its reflection it is possible to distinguish between different types of faults, for example open or shorted pairs.

Configuring Time Domain Reflectometer

To configure an interface template, follow these steps.

Procedure

	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	**test cable-diagnostics tdr {interface { Starts the TDR test. interface-number }}** **Example:** `Device(config)# test cable-diagnostics tdr {interface { Starts the TDR test. interface-number }}`	Starts the TDR test.
Step 4	**show cable-diagnostics tdr {interfaces}** **Example:** `Device(config)# show cable-diagnostics tdr {interfaces}`	Displays the TDR test counter information. interface-number
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Cisco C-SM-16P4M2X or C-SM-40P4M2X Service Module

To troubleshoot and collect debug logs, use the following commands:

- Check the status of the module by using the **show platform** command.
- To check if the related vlan is created, use the **show vlan id** <*id_number*> command.
- Ensure the port is not blocked by Spanning Tree Protocol, or error-disabled by UDLD, port-security, and so on.
- When both the Cisco C-SM-16P4M2X or C-SM-40P4M2X are inserted in the same router, the Cisco 16-Port service module takes the priority. The router reboots and work in ‘next-gen switching mode’ instead of ‘legacy switching mode’. After the reload, Cisco 4-Port and 8-Port goes out of service’, the Cisco 16-Port is active.

Related Documents

Related Topic	Document Title
I Installing the Cisco C-SM-16P4M2X or C-SM-40P8M2X EtherSwitch Service Module	Installing the Cisco C-SM-16P4M2X or C-SM-40P8M2X EtherSwitch Service Module

Conventions

This document uses the following conventions.

Conventions	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
courier font	Terminal sessions and information the system displays appear in courier font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see What's New in Cisco Product Documentation at: http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html .

Subscribe to What's New in Cisco Product Documentation, which lists all new and revised Cisco technical documentation as an RSS feed and delivers content directly to your desktop using a reader application. The RSS feeds are a free service.