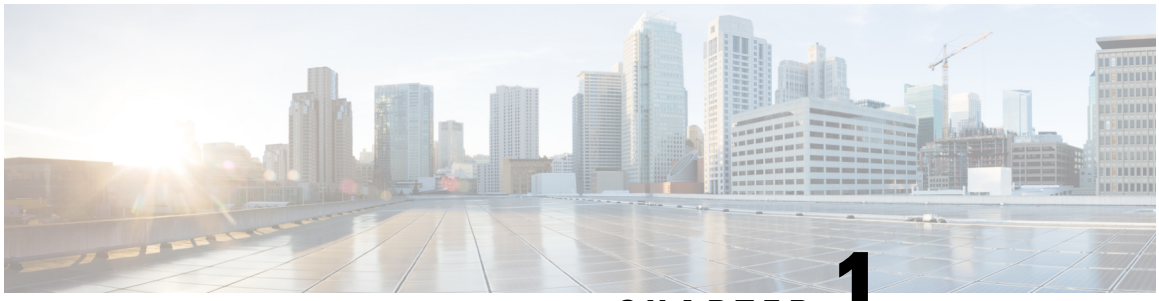
Carrier Ethernet Configuration Guide, Cisco IOS XE 17 (Cisco NCS 520 Series)

First Published: 2019-11-26

Last Modified: 2022-03-31

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

CONTENTS

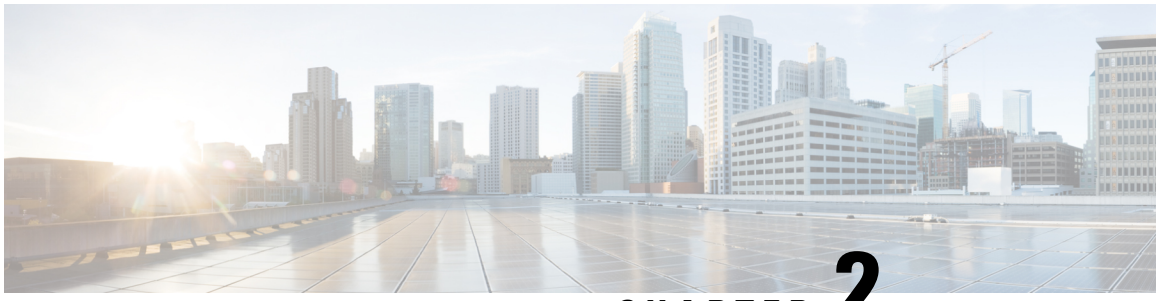


CHAPTER 1

Feature History

The following table lists the new and modified features that are supported in the Cisco NCS 520 Series Carrier Ethernet Configuration Guide in Cisco IOS XE 17 releases.

Feature	Description
Cisco IOS XE Bengaluru 17.6.1	
Unicast MAC for CCM Messages	Continuity Check Messages (CCM) use multicast destination MAC address by default. This feature enables you to unicast CCM messages to a specific remote MEP (RMEP) to avoid unnecessary traffic flood on the VLAN.



CHAPTER 2

Using Ethernet Operations Administration and Maintenance

Ethernet Operations, Administration, and Maintenance (OAM) is a protocol for installing, monitoring, and troubleshooting Ethernet metropolitan-area networks (MANs) and Ethernet WANs. It relies on a new, optional sublayer in the data link layer of the Open Systems Interconnection (OSI) model. The OAM features covered by this protocol are Discovery, Link Monitoring, Remote Fault Detection, Remote Loopback, and Cisco Proprietary Extensions.

The advent of Ethernet as a MAN and WAN technology has emphasized the necessity for integrated management for larger deployments. For Ethernet to extend into public MANs and WANs, it must be equipped with a new set of requirements on Ethernet's traditional operations, which had been centered on enterprise networks only. The expansion of Ethernet technology into the domain of service providers, where networks are substantially larger and more complex than enterprise networks and the user-base is wider, makes operational management of link uptime crucial.

Information About Using Ethernet Operations Administration and Maintenance

Ethernet OAM

Ethernet OAM is a protocol for installing, monitoring, and troubleshooting metro Ethernet networks and Ethernet WANs. It relies on a new, optional sublayer in the data link layer of the OSI model. Ethernet OAM can be implemented on any full-duplex point-to-point or emulated point-to-point Ethernet link. A system-wide implementation is not required; OAM can be deployed for part of a system; that is, on particular interfaces.

Normal link operation does not require Ethernet OAM. OAM frames, called OAM protocol data units (PDUs), use the slow protocol destination MAC address 0180.c200.0002. They are intercepted by the MAC sublayer and cannot propagate beyond a single hop within an Ethernet network.

Ethernet OAM is a relatively slow protocol with modest bandwidth requirements. The frame transmission rate is limited to a maximum of 10 frames per second; therefore, the impact of OAM on normal operations is

negligible. However, when link monitoring is enabled, the CPU must poll error counters frequently. In this case, the required CPU cycles will be proportional to the number of interfaces that have to be polled.

Two major components, the OAM client and the OAM sublayer, make up Ethernet OAM. The following two sections describe these components.

OAM Client

The OAM client is responsible for establishing and managing Ethernet OAM on a link. The OAM client also enables and configures the OAM sublayer. During the OAM discovery phase, the OAM client monitors OAM PDUs received from the remote peer and enables OAM functionality on the link based on local and remote state as well as configuration settings. Beyond the discovery phase (at steady state), the OAM client is responsible for managing the rules of response to OAM PDUs and managing the OAM remote loopback mode.

OAM Sublayer

The OAM sublayer presents two standard IEEE 802.3 MAC service interfaces: one facing toward the superior sublayers, which include the MAC client (or link aggregation), and the other interface facing toward the subordinate MAC control sublayer. The OAM sublayer provides a dedicated interface for passing OAM control information and OAM PDUs to and from a client.

The OAM sublayer is made up of three components: control block, multiplexer, and packet parser (p-parser). Each component is described in the following sections.

Control Block

The control block provides the interface between the OAM client and other blocks internal to the OAM sublayer. The control block incorporates the discovery process, which detects the existence and capabilities of remote OAM peers. It also includes the transmit process that governs the transmission of OAM PDUs to the multiplexer and a set of rules that govern the receipt of OAM PDUs from the p-parser.

Multiplexer

The multiplexer manages frames generated (or relayed) from the MAC client, control block, and p-parser. The multiplexer passes through frames generated by the MAC client untouched. It passes OAM PDUs generated by the control block to the subordinate sublayer; for example, the MAC sublayer. Similarly, the multiplexer passes loopback frames from the p-parser to the same subordinate sublayer when the interface is in OAM remote loopback mode.

P-Parser

The p-parser classifies frames as OAM PDUs, MAC client frames, or loopback frames and then dispatches each class to the appropriate entity. OAM PDUs are sent to the control block. MAC client frames are passed to the superior sublayer. Loopback frames are dispatched to the multiplexer.

Benefits of Ethernet OAM

Ethernet OAM provides the following benefits:

- Competitive advantage for service providers.
- Standardized mechanism to monitor the health of a link and perform diagnostics.

Note REP traps are prioritized when both Ethernet OAM and REP traps are configured on the same port. If you want to view Ethernet OAM logs, then you must disable REP configurations.

Cisco Implementation of Ethernet OAM

The Cisco implementation of Ethernet OAM consists of the Ethernet OAM shim and the Ethernet OAM module.

The Ethernet OAM shim is a thin layer that connects the Ethernet OAM module and the platform code. It is implemented in the platform code (driver). The shim also communicates port state and error conditions to the Ethernet OAM module via control signals.

The Ethernet OAM module, implemented within the control plane, handles the OAM client as well as control block functionality of the OAM sublayer. This module interacts with the CLI and Simple Network Management Protocol (SNMP)/programmatic interface via control signals. In addition, this module interacts with the Ethernet OAM shim through OAM PDU flows.

OAM Features

The OAM features as defined by IEEE 802.3ah, *Ethernet in the First Mile*, are discovery, Link Monitoring, Remote Fault Detection, Remote Loopback, and Cisco Proprietary Extensions.

Discovery

Discovery is the first phase of Ethernet OAM and it identifies the devices in the network and their OAM capabilities. Discovery uses information OAM PDUs. During the discovery phase, the following information is advertised within periodic information OAM PDUs:

- OAM mode—Conveyed to the remote OAM entity. The mode can be either active or passive and can be used to determine device functionality.
- OAM configuration (capabilities)—Advertises the capabilities of the local OAM entity. With this information a peer can determine what functions are supported and accessible; for example, loopback capability.
- OAM PDU configuration—Includes the maximum OAM PDU size for receipt and delivery. This information along with the rate limiting of 10 frames per second can be used to limit the bandwidth allocated to OAM traffic.
- Platform identity—A combination of an organization unique identifier (OUI) and 32-bits of vendor-specific information. OUI allocation, controlled by the IEEE, is typically the first three bytes of a MAC address.

Discovery includes an optional phase in which the local station can accept or reject the configuration of the peer OAM entity. For example, a node may require that its partner support loopback capability to be accepted into the management network. These policy decisions may be implemented as vendor-specific extensions.

Link Monitoring

Link monitoring in Ethernet OAM detects and indicates link faults under a variety of conditions. Link monitoring uses the event notification OAM PDU and sends events to the remote OAM entity when there are problems detected on the link. The error events include the following:

- Error Symbol Period (error symbols per second)—The number of symbol errors that occurred during a specified period exceeded a threshold. These errors are coding symbol errors.
- Error Frame (error frames per second)—The number of frame errors detected during a specified period exceeded a threshold.
- Error Frame Period (error frames per *n* frames)—The number of frame errors within the last n frames has exceeded a threshold.
- Error Frame Seconds Summary (error seconds per *m* seconds)—The number of error seconds (1-second intervals with at least one frame error) within the last m seconds has exceeded a threshold.

Since IEEE 802.3ah OAM does not provide a guaranteed delivery of any OAM PDU, the event notification OAM PDU may be sent multiple times to reduce the probability of a lost notification. A sequence number is used to recognize duplicate events.

Remote Failure Indication

Faults in Ethernet connectivity that are caused by slowly deteriorating quality are difficult to detect. Ethernet OAM provides a mechanism for an OAM entity to convey these failure conditions to its peer via specific flags in the OAM PDU. The following failure conditions can be communicated:

- Link Fault—Loss of signal is detected by the receiver; for instance, the peer's laser is malfunctioning. A link fault is sent once per second in the information OAM PDU. Link fault applies only when the physical sublayer is capable of independently transmitting and receiving signals.
- Dying Gasp—An unrecoverable condition has occurred; for example, when an interface is shut down. This type of condition is vendor specific. A notification about the condition may be sent immediately and continuously.

Note Dying Gasp is only supported on interface down events. It is not supported in System down scenarios.

For more information on Dying Gasp, see the Dying Gasp Support for Loss of Power Supply Through SNMP, Syslog and Ethernet OAM chapter in the Cisco NCS 520 Series Router Configuration Guide.

- Critical Event—An unspecified critical event has occurred. This type of event is vendor specific. A critical event may be sent immediately and continuously.

Remote Loopback

An OAM entity can put its remote peer into loopback mode using the loopback control OAM PDU. Loopback mode helps an administrator ensure the quality of links during installation or when troubleshooting. In loopback mode, every frame received is transmitted back on the same port except for OAM PDUs and pause frames. The periodic exchange of OAM PDUs must continue during the loopback state to maintain the OAM session.

The loopback command is acknowledged by responding with an information OAM PDU with the loopback state indicated in the state field. This acknowledgement allows an administrator, for example, to estimate if a network segment can satisfy a service-level agreement. Acknowledgement makes it possible to test delay, jitter, and throughput.

When an interface is set to the remote loopback mode the interface no longer participates in any other Layer 2 or Layer 3 protocols; for example Spanning Tree Protocol (STP) or Open Shortest Path First (OSPF). The reason is that when two connected ports are in a loopback session, no frames other than the OAM PDUs are sent to the CPU for software processing. The non-OAM PDU frames are either looped back at the MAC level or discarded at the MAC level.

From a user's perspective, an interface in loopback mode is in a link-up state.

Note Remote loopback is *not* supported on the RSP3 module.

Cisco Vendor-Specific Extensions

Ethernet OAM allows vendors to extend the protocol by allowing them to create their own type-length-value (TLV) fields.

OAM Messages

Ethernet OAM messages or OAM PDUs are standard length, untagged Ethernet frames within the normal frame length bounds of 64 to 1518 bytes. The maximum OAM PDU frame size exchanged between two peers is negotiated during the discovery phase.

OAM PDUs always have the destination address of slow protocols (0180.c200.0002) and an Ethertype of 8809. OAM PDUs do not go beyond a single hop and have a hard-set maximum transmission rate of 10 OAM PDUs per second. Some OAM PDU types may be transmitted multiple times to increase the likelihood that they will be successfully received on a deteriorating link.

Four types of OAM messages are supported:

- Information OAM PDU--A variable-length OAM PDU that is used for discovery. This OAM PDU includes local, remote, and organization-specific information.
- Event notification OAM PDU--A variable-length OAM PDU that is used for link monitoring. This type of OAM PDU may be transmitted multiple times to increase the chance of a successful receipt; for example, in the case of high-bit errors. Event notification OAM PDUs also may include a time stamp when generated.
- Loopback control OAM PDU--An OAM PDU fixed at 64 bytes in length that is used to enable or disable the remote loopback command.
- Vendor-specific OAM PDU--A variable-length OAM PDU that allows the addition of vendor-specific extensions to OAM.

IEEE 802.3ah Link Fault RFI Support

The IEEE 802.3ah Link Fault RFI Support feature provides a per-port configurable option that moves a port into a blocking state when an OAM PDU control request packet is received with the Link Fault Status flag

set. In the blocking state, the port can continue to receive OAM PDUs, detect remote link status, and automatically recover when the remote link becomes operational. When an OAM PDU is received with the Link Fault Status flag set to zero or FALSE, the port is enabled and all VLANs configured on the port are set to “forwarding.”

Note If you configure the Ethernet OAM timeout period to be the minimum allowable value of 2 seconds, the Ethernet OAM session may be dropped briefly when the port transitions from blocked to unblocked. This action will not occur by default; the default timeout value is 5 seconds.

Before the release of the IEEE 802.3ah Link Fault RFI Support feature, when an OAM PDU control request packet was received with the Link Fault Status flag set, one of three actions was taken:

- The port was put in the error-disable state, meaning that the port did not send or receive packets, including Bridge Protocol Data Units (BPDU) packets. In the error-disable state, a link can automatically recover after the error-disable timeout period but cannot recover automatically when the remote link becomes operational.
- A warning message was displayed or logged, and the port remained operational.
- The Link Fault Status flag was ignored.

Ethernet Connectivity Fault Management

Ethernet connectivity fault management (CFM) is an end-to-end per-service-instance Ethernet layer OAM protocol that includes proactive connectivity monitoring, fault verification, and fault isolation. End to end can be provider edge (PE) to PE or customer edge (CE) to CE. Per service instance means per VLAN.

For more information about Ethernet CFM, see Ethernet Connectivity Fault Management .

Understanding E-LMI and Interactions with CFM

Ethernet Local Management Interface (E-LMI) is a protocol between the customer edge (CE) device and the provider edge (PE) device. It runs only on the PE-CE UNI link and notifies the CE of connectivity status and configuration parameters of Ethernet services available on the CE port. E-LMI interoperates with an OAM protocol, such as CFM, that runs within the provider network to collect OAM status. CFM runs at the provider maintenance level (UPE to UPE with inward-facing MEPs at the UNI). E-LMI relies on the OAM Ethernet Infrastructure (EI) to interwork with CFM for end-to-end status of Ethernet virtual connections (EVCs) across CFM domains.

OAM manager streamlines interaction between OAM protocols, and handles the interaction between CFM and E-LMI. E-LMI interaction with OAM manager is unidirectional, running only from OAM manager to E-LMI on the UPE side of the switch. Information is exchanged either as a result of a request from E-LMI or triggered by OAM when it received notification of a change from the OAM protocol. This type of information is relayed:

- EVC name and availability status
- Remote UNI name and status
- Remote UNI counts

You can configure Ethernet virtual connections (EVCs), service VLANs, UNI ids (for each CE-to-PE link), and UNI count and attributes. You need to configure CFM to notify the OAM manager of any change to the number of active UNIs and or the remote UNI ID for a given S-VLAN domain.

E-LMI implementation on the Cisco ME 3400 switch includes only PE-side support.

How to Set Up and Configure Ethernet Operations Administration and Maintenance

Enabling Ethernet OAM on an Interface

Ethernet OAM is by default disabled on an interface.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet 0/0/1
```

Specifies an interface and enters interface configuration mode.

Step 4 **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]

Example:

```
Device(config-if)# ethernet oam
```

Enables Ethernet OAM.

Step 5 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Disabling and Enabling a Link Monitoring Session

Link monitoring is enabled by default when you enable Ethernet OAM. Perform these tasks to disable and enable link monitoring sessions:

Disabling a Link Monitoring Session

Perform this task to disable a link monitoring session.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitEthernet 0/0/2
```

Specifies an interface and enters interface configuration mode.

Step 4 **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]

Example:

```
Device(config-if)# ethernet oam
```

Enables Ethernet OAM.

Step 5 **no ethernet oam link-monitor supported**

Example:

```
Device(config-if)# no ethernet oam link-monitor supported
```

Disables link monitoring on the interface.

Step 6 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Enabling a Link Monitoring Session

Perform this task to reenable a link monitoring session after it was previously disabled.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitEthernet 0/0/1
```

Specifies an interface and enters interface configuration mode.

Step 4 **ethernet oam link-monitor supported**

Example:

```
Device(config-if)# ethernet oam link-monitor supported
```

Enables link monitoring on the interface.

Step 5 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Stopping and Starting Link Monitoring Operations

Link monitoring operations start automatically when Ethernet OAM is enabled on an interface. When link monitoring operations are stopped, the interface does not actively send or receive event notification OAM PDUs. The tasks in this section describe how to stop and start link monitoring operations.

Stopping Link Monitoring Operations

Perform this task to stop link monitoring operations.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet 0/0/2
```

Specifies an interface and enters interface configuration mode.

Step 4 **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]

Example:

```
Device(config-if)# ethernet oam
```

Enables Ethernet OAM.

Step 5 **no ethernet oam link-monitor on**

Example:

```
Device(config-if)# no ethernet oam link-monitor on
```

Stops link monitoring operations.

Step 6 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Starting Link Monitoring Operations

Perform this task to start link monitoring operations.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet 0/0/2
```

Specifies an interface and enters interface configuration mode.

Step 4 **ethernet oam link-monitor on**

Example:

```
Device(config-if)# ethernet oam link-monitor on
```

Starts link monitoring operations.

Step 5 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Configuring Link Monitoring Options

Perform this optional task to specify link monitoring options. Steps 4 through 10 can be performed in any sequence.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitEthernet 0/0/3
```

Identifies the interface and enters interface configuration mode.

Step 4 **ethernet oam** [**max-rate** *oampdus* | **min-rate** *num-seconds*| **mode** {**active** | **passive**} | **timeout** *seconds*]

Example:

```
Device(config-if)# ethernet oam
```

Enables Ethernet OAM.

Step 5 **ethernet oam link-monitor high-threshold action error-disable-interface**

Example:

```
Device(config-if)# ethernet oam link-monitor high-threshold action error-disable-interface
```

Configures an error-disable function on an Ethernet OAM interface when a high threshold for an error is exceeded.

Step 6 **ethernet oam link-monitor frame** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-if)# ethernet oam link-monitor frame window 399
```

Configures a number for error frames that when reached triggers an action.

Step 7 **ethernet oam link-monitor frame-period** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *frames*}

Example:

```
Device(config-if)# ethernet oam link-monitor frame-period threshold high 599
```

Configures a number of frames to be polled.

Frame period is a user-defined parameter.

Step 8 **ethernet oam link-monitor frame-seconds** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-if)# ethernet oam link-monitor frame-seconds window 699
```

Configures a period of time in which error frames are counted.

Step 9 **ethernet oam link-monitor receive-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-if)# ethernet oam link-monitor receive-crc window 99
```

Configures an Ethernet OAM interface to monitor ingress frames with cyclic redundancy check (CRC) errors for a period of time.

Step 10 **ethernet oam link-monitor symbol-period** {**threshold** {**high** {**none** | *high-symbols*} | **low** *low-symbols*} | **window** *symbols*}

Example:

```
Device(config-if)# ethernet oam link-monitor symbol-period threshold high 299
```

Configures a threshold or window for error symbols, in number of symbols.

Step 11 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Example

Configuring Global Ethernet OAM Options Using a Template

Perform this task to create a template to use for configuring a common set of options on multiple Ethernet OAM interfaces. Steps 4 through 10 are optional and can be performed in any sequence. These steps may also be repeated to configure different options.

Procedure

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** *template-name*

Example:

```
Device(config)# template oam-temp
```

Configures a template and enters template configuration mode.

Step 4 **ethernet oam link-monitor receive-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-template)# ethernet oam link-monitor receive-crc window 99
```

Configures an Ethernet OAM interface to monitor ingress frames with CRC errors for a period of time.

Step 5 **ethernet oam link-monitor transmit-crc** {**threshold** {**high** {*high-frames* | **none**} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-template)# ethernet oam link-monitor transmit-crc threshold low 199
```

Configures an Ethernet OAM interface to monitor egress frames with CRC errors for a period of time.

Step 6 **ethernet oam link-monitor symbol-period** {**threshold** {**high** {**none** | *high-symbols*} | **low** *low-symbols*} | **window** *symbols*}

Example:

```
Device(config-template)# ethernet oam link-monitor symbol-period threshold high 299
```

Configures a threshold or window for error symbols, in number of symbols.

Step 7 **ethernet oam link-monitor high-threshold action error-disable-interface**

Example:

```
Device(config-template)# ethernet oam link-monitor high-threshold action
error-disable-interface
```

Configures an error-disable function on an Ethernet OAM interface when a high threshold for an error is exceeded.

Step 8 **ethernet oam link-monitor frame** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-template)# ethernet oam link-monitor frame window 399
```

Configures a number for error frames that when reached triggers an action.

Step 9 **ethernet oam link-monitor frame-period** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *frames*}

Example:

```
Device(config-template)# ethernet oam link-monitor frame-period threshold high 599
```

Configures a number of frames to be polled.

Frame period is a user-defined parameter.

Step 10 **ethernet oam link-monitor frame-seconds** {**threshold** {**high** {**none** | *high-frames*} | **low** *low-frames*} | **window** *milliseconds*}

Example:

```
Device(config-template)# ethernet oam link-monitor frame-seconds window 699
```

Configures a period of time in which error frames are counted.

Step 11 **exit**

Example:

```
Device(config-template)# exit
```

Returns to global configuration mode.

Step 12 **interface** *type number*

Example:

```
Device(config)# interface gigabitEthernet 0/0/2
```

Identifies the interface on which to use the template and enters interface configuration mode.

Step 13 **source template** *template-name*

Example:

```
Device(config-if)# source template oam-temp
```

Applies to the interface the options configured in the template.

Step 14 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Step 15 **exit**

Example:

```
Device(config)# exit
```

Returns to privileged EXEC mode.

Step 16 **show running-config**

Example:

```
Device# show running-config
```

Displays the updated running configuration.

Configuring a Port for Link Fault RFI Support

Perform this task to put a port into a blocking state when an OAM PDU control request packet is received with the Link Fault Status flag set.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet 0/0/1
```

Enters interface configuration mode.

Step 4 **ethernet oam remote-failure** {**critical-event** | **dying-gasp** | **link-fault**} **action** { **error-disable-interface**}

Example:

```
Device(config-if)# ethernet oam remote-failure dying-gasp action error-disable-interface
```

Sets the interface to the blocking state when a critical event occurs.

Step 5 **exit**

Example:

```
Device(config-if)# exit
```

Returns to global configuration mode.

Configuring E-LMI for Interaction with CFM

For E-LMI to work with CFM, you configure Ethernet virtual connections (EVCs), Ethernet service instances (EFPs), and E-LMI customer VLAN mapping. Most of the configuration occurs on the PE switch on the interfaces connected to the CE. On the CE switch, you only need to enable E-LMI on the connecting interface. Note that you must configure some OAM parameters, for example, EVC definitions, on PE devices on both sides of a metro network.

Default E-LMI and OAM Configuration

Ethernet LMI is globally disabled by default.

When you globally enable E-LMI by entering the **ethernet lmi global** global configuration command, it is automatically enabled on all interfaces. You can also enable or disable E-LMI per interface to override the global configuration. The command given last is the command that has precedence.

There are no EVCs, EFP service instances, or UNIs defined.

UNI bundling service is bundling with multiplexing.

Configuration Guidelines

OAM manager is an infrastructural element and requires two interworking OAM protocols, in this case CFM and E-LMI. For OAM to operate, the PE side of the connection must be running CFM and E-LMI.

- E-LMI is supported only when the metro IP access or metro access image is running on the switch.

- You cannot configure E-LMI on VLAN interfaces.

Enabling Ethernet OAM Remote Loopback

You must enable Ethernet OAM remote loopback on an interface for the local OAM client to initiate OAM remote loopback operations. Changing this setting causes the local OAM client to exchange configuration information with its remote peer. Remote loopback is disabled by default. Remote loopback has the following limitation:

- If dynamic ARP inspection is enabled, ARP or reverse ARP packets are not looped or dropped.

Use the **no ethernet oam remote-loopback {supported | timeout}** interface configuration command to disable remote loopback support or remove the timeout setting.

Beginning in privileged EXEC mode, follow these steps to enable Ethernet OAM remote loopback on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**interface** *interface-id*	Define an interface to configure as an EOM interface, and enter interface configuration mode.
Step 3	**ethernet oam remote-loopback** {**supported** | **timeout** *seconds*}	Enable Ethernet remote loopback on the interface or set a loopback timeout period. • Enter supported to enable remote loopback. • Enter timeout seconds to set a remote loopback timeout period. The range is from 1 to 10 seconds.
Step 4	end	Return to privileged EXEC mode.
Step 5	**ethernet oam remote-loopback** {**start** | **stop**} {**interface** *interface-id*}	Turn on or turn off Ethernet OAM remote loopback on an interface.
Step 6	**showethernet oam status** [**interface** *interface-id*]	Verify the configuration.
Step 7	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring Ethernet OAM Link Monitoring

You can configure high and low thresholds for link-monitoring features. If no high threshold is configured, the default is none —no high threshold is set. If you do not set a low threshold, it defaults to a value lower than the high threshold.

The **ethernet oam link-monitor receive-crc** {**threshold**{**high** {*high-frames* | **none**} | **low** {*low-frames*}} | **window** *milliseconds*} command is visible on the router and you are allowed to enter it, but it is not supported. Enter the no form of the commands to disable the configuration. Use the no form of each command to disable the threshold setting.

Beginning in privileged EXEC mode, follow these steps to configure Ethernet OAM link monitoring on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**interface** *interface-id*	Define an interface, and enter interface configuration mode.
Step 3	**ethernet oam link-monitor supported**	Enable the interface to support link monitoring. This is the default. You need to enter this command only if it has been disabled by previously entering the no ethernet oam link-monitor supported command.
Step 4	**ethernet oam link-monitor symbol-period** {**threshold**{**high** {*high-symbols*	**none**}	**low** {*low-symbols*}}	**window** *symbols*}	(Optional) Configure high and low thresholds for an error-symbol period that trigger an error-symbol period link event. • Enter threshold high high-symbols to set a high threshold in number of symbols. The range is 1 to 65535. The default is none. • Enter threshold high none to disable the high threshold if it was set. This is the default. • Enter threshold low low-symbols to set a low threshold in number of symbols. The range is 0 to 65535. It must be lower than the high threshold. • Enter window symbols to set the window size (in number of symbols) of the polling period. The range is 1 to 65535 symbols. **Note** Repeat this step to configure both high and low thresholds.

	Command or Action	Purpose
Step 5	**ethernet oam link-monitor frame** {**threshold**{**high** {*high-frames* | **none**} | **low** {*low-frames*}} | **window** *milliseconds*}	(Optional) Configure high and low thresholds for error frames that trigger an error-frame link event. • Enter threshold high high-frames to set a high threshold in number of frames. The range is 1 to 65535. The default is none. • Enter threshold high none to disable the high threshold if it was set. This is the default. • Enter threshold low low-frames to set a low threshold in number of frames. The range is 0 to 65535. The default is 1. • Enter window milliseconds to set the a window and period of time during which error frames are counted. The range is 10 to 600 and represents the number of milliseconds in multiples of 100. The default is 100. **Note** Repeat this step to configure both high and low thresholds.
Step 6	**ethernet oam link-monitor frame-period** {**threshold**{**high** {*high-frames* | **none**} | **low** {*low-frames*}} | **window** *frames*}	(Optional) Configure high and low thresholds for the error-frame period that triggers an error-frame-period link event. • Enter threshold high high-frames to set a high threshold in number of frames. The range is 1 to 65535. The default is none. • Enter threshold high none to disable the high threshold if it was set. This is the default. • Enter threshold low low-frames to set a low threshold in number of frames. The range is 0 to 65535. The default is 1. • Enter window frames to set the a polling window size in number of frames. The range is 1 to 65535; each value is a multiple of 10000 frames. The default is 1000. **Note** Repeat this step to configure both high and low thresholds.

	Command or Action	Purpose
Step 7	**ethernet oam link-monitor frame-seconds** {**threshold**{**high** {*high-frames* | **none**} | **low** {*low-frames*}} | **window** *milliseconds*}	(Optional) Configure high and low thresholds for the frame-seconds error that triggers an error-frame-seconds link event. • Enter threshold high high-frames to set a high error frame-seconds threshold in number of seconds. The range is 1 to 900. The default is none. • Enter threshold high none to disable the high threshold if it was set. This is the default. • Enter threshold low low-frames to set a low threshold in number of frames. The range is 1 to 900. The default is 1. • Enter window frames to set the a polling window size in number of milliseconds. The range is 100 to 9000; each value is a multiple of 100 milliseconds. The default is 1000. **Note** Repeat this step to configure both high and low thresholds.
Step 8	**ethernet oam link-monitor receive-crc** {**threshold**{**high** {*high-frames* | **none**} | **low** {*low-frames*}} | **window** *milliseconds*}	(Optional) Configure thresholds for monitoring ingress frames received with cyclic redundancy code (CRC) errors for a period of time. • Enter threshold high high-frames to set a high threshold for the number of frames received with CRC errors. The range is 1 to 65535 frames. • Enter threshold high none to disable the high threshold. • Enter threshold low low-frames to set a low threshold in number of frames. The range is 0 to 65535. The default is 1. • Enter window milliseconds to set the a window and period of time during which frames with CRC errors are counted. The range is 10 to 1800 and represents the number of milliseconds in multiples of 100. The default is 100. **Note** Repeat this step to configure both high and low thresholds.

	Command or Action	Purpose
Step 9	[**no**] **ethernet oam link-monitor on**	(Optional) Start or stop (when the no keyword is entered) link-monitoring operations on the interface. Link monitoring operations start automatically when support is enabled.
Step 10	end	Return to privileged EXEC mode.
Step 11	**showethernet oam status** [**interface** *interface-id*]	Verify the configuration.
Step 12	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuration Examples for Ethernet Operations Administration and Maintenance

The following example shows how to configure Ethernet OAM options using a template and overriding that configuration by configuring an interface. In this example, the network supports a Gigabit Ethernet interface between the customer edge device and provider edge device.

```
! Configure a global OAM template for both PE and CE configuration.
!
Device(config)# template oam
Device(config-template)# ethernet oam link-monitor symbol-period threshold low 10
Device(config-template)# ethernet oam link-monitor symbol-period threshold high 100
Device(config-template)# ethernet oam link-monitor frame window 100
Device(config-template)# ethernet oam link-monitor frame threshold low 10
Device(config-template)# ethernet oam link-monitor frame threshold high 100
Device(config-template)# ethernet oam link-monitor frame-period window 100
Device(config-template)# ethernet oam link-monitor frame-period threshold low 10
Device(config-template)# ethernet oam link-monitor frame-period threshold high 100
Device(config-template)# ethernet oam link-monitor frame-seconds window 1000
Device(config-template)# ethernet oam link-monitor frame-seconds threshold low 10
Device(config-template)# ethernet oam link-monitor frame-seconds threshold high 100
Device(config-template)# ethernet oam link-monitor receive-crc window 100
Device(config-template)# ethernet oam link-monitor receive-crc threshold high 100
Device(config-template)# ethernet oam link-monitor transmit-crc window 100
Device(config-template)# ethernet oam link-monitor transmit-crc threshold high 100
Device(config-template)# ethernet oam remote-failure dying-gasp action error-disable-interface
Device(config-template)# exit
!
! Enable Ethernet OAM on the CE interface
!
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ethernet oam
!
! Apply the global OAM template named "oam" to the interface.
!
Device(config-if)# source template oam
!
! Configure any interface-specific link monitoring commands to override the template
configuration. The following example disables the high threshold link monitoring for receive
 CRC errors.
```

```
!
Device(config-if)# ethernet oam link-monitor receive-crc threshold high none
!
! Enable Ethernet OAM on the PE interface
!
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ethernet oam
!
! Apply the global OAM template named "oam" to the interface.
!
Device(config-if)# source template oam
```

The following examples show how to verify various Ethernet OAM configurations and activities.

Verifying an OAM Session

The following example shows that the local OAM client, Gigabit Ethernet interface Gi0/0/1, is in session with a remote client with MAC address 0012.7fa6.a700 and OUI 00000C, which is the OUI for Cisco. The remote client is in active mode and has established capabilities for link monitoring and remote loopback for the OAM session.

```
Device# show ethernet oam summary
Symbols:          * - Master Loopback State,  # - Slave Loopback State
Capability codes: L - Link Monitor,  R - Remote Loopback
                  U - Unidirection,  V - Variable Retrieval
  Local                         Remote
Interface       MAC Address    OUI    Mode    Capability
 Gi6/1/1       0012.7fa6.a700 00000C active      L R
```

Verifying OAM Discovery Status

The following example shows how to verify OAM discovery status of a local client and a remote peer:

```
Device# show ethernet oam discovery interface gigabitethernet0/0/1
GigabitEthernet0/0/1show ethernet oam discovery
Local client
------------
  Administrative configurations:
    Mode:              active
    Unidirection:      not supported
    Link monitor:      supported (on)
    Remote loopback:   not supported
    MIB retrieval:     not supported
    Mtu size:          1500
  Operational status:
Port status:       operational
    Loopback status:   no loopback
    PDU permission:    any
    PDU revision:      1
Remote client
-------------
  MAC address: 0030.96fd.6bfa
  Vendor(oui): 0x00 0x00 0x0C (cisco)
  Administrative configurations:
   PDU revision:      2
   Mode:              active
   Unidirection:      not supported
   Link monitor:      supported
   Remote loopback:   not supported
   MIB retrieval:     not supported
   Mtu size:          1500
```

Verifying Information OAMPDU and Fault Statistics

The following example shows how to verify statistics for information OAM PDUs and local and remote faults:

```
Device# show ethernet oam statistics interface gigabitethernet0/0/1
GigabitEthernet0/0/1show ethernet oam statistics
Counters:
---------
Information OAMPDU Tx                   : 588806
Information OAMPDU Rx                   : 988
Unique Event Notification OAMPDU Tx     : 0
Unique Event Notification OAMPDU Rx     : 0
Duplicate Event Notification OAMPDU TX  : 0
Duplicate Event Notification OAMPDU RX  : 0
Loopback Control OAMPDU Tx              : 1
Loopback Control OAMPDU Rx              : 0
Variable Request OAMPDU Tx              : 0
Variable Request OAMPDU Rx              : 0
Variable Response OAMPDU Tx             : 0
Variable Response OAMPDU Rx             : 0
Cisco OAMPDU Tx                         : 4
Cisco OAMPDU Rx                         : 0
Unsupported OAMPDU Tx                   : 0
Unsupported OAMPDU Rx                   : 0
Frames Lost due to OAM                  : 0
Local Faults:
-------------
0 Link Fault records
2 Dying Gasp records
Total dying gasps       : 4
Time stamp              : 00:30:39
Total dying gasps       : 3
Time stamp              : 00:32:39
0 Critical Event records
Remote Faults:
--------------
0 Link Fault records
0 Dying Gasp records
0 Critical Event records
Local event logs:
-----------------
0 Errored Symbol Period records
0 Errored Frame records
0 Errored Frame Period records
0 Errored Frame Second records
Remote event logs:
------------------
0 Errored Symbol Period records
0 Errored Frame records
0 Errored Frame Period records
0 Errored Frame Second records
```

Verifying Link Monitoring Configuration and Status

The following example shows how to verify link monitoring configuration and status on the local client. The highlighted Status field in the example shows that link monitoring status is supported and enabled (on).

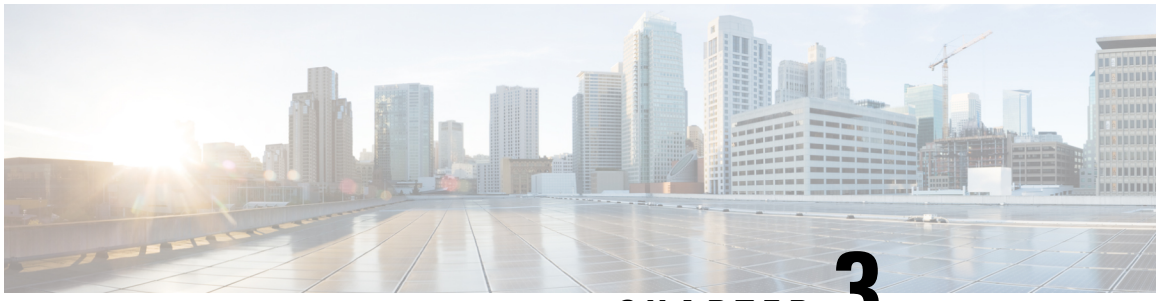
```
Device# show ethernet oam status interface gigabitethernet0/0/1
GigabitEthernet0/0/1show ethernet oam discovery
General
-------
  Mode:                 active
```

```
  PDU max rate:           10 packets per second
  PDU min rate:           1 packet per 1 second
  Link timeout:           5 seconds
  High threshold action: no action
Link Monitoring
---------------
  Status: supported (on)
  Symbol Period Error
    Window:               1 million symbols
    Low threshold:        1 error symbol(s)
    High threshold:       none
  Frame Error
    Window:               10 x 100 milliseconds
    Low threshold:        1 error frame(s)
    High threshold:       none
Frame Period Error
    Window:               1 x 100,000 frames
    Low threshold:        1 error frame(s)
    High threshold:       none
  Frame Seconds Error
    Window:               600 x 100 milliseconds
    Low threshold:        1 error second(s)
    High threshold:       none
```

Verifying Status of a Remote OAM Client

The following example shows that the local client interface Gi6/1/1 is connected to a remote client. Note the values in the Mode and Capability fields.

```
Device# show ethernet oam summary
Symbols:          * - Master Loopback State,  # - Slave Loopback State
Capability codes: L - Link Monitor,  R - Remote Loopback
                  U - Unidirection,  V - Variable Retrieval
  Local                        Remote
Interface       MAC Address    OUI    Mode    Capability
 Gi6/1/1       0012.7fa6.a700 00000C active      L R
```



CHAPTER 3

Trunk EFP Support

The Trunk EFP Support feature provides support for Ethernet flow points (EFPs) on trunk ports. A trunk port allows a range of VLANs to be forwarded on a given interface while still maintaining data-plane segmentation between the VLANs.

Finding Feature Information

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

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

Restrictions for Trunk EFP Support

- The **rewrite ingress tag pop 1 symmetric** command is the only **rewrite** command that is supported for trunk EFP configurations. The **rewrite ingress tag pop 1 symmetric** command must be included in the configuration when the Trunk EFP Support feature is enabled.
- A bridge-domain number that is part of a trunk EFP configuration cannot be shared by other EFPs under the same port or interface.
- Only one trunk EFP can be configured under one port or interface.
- All features configured on a trunk EFP (other than encapsulations and bridge-domain assignments) are applied uniformly to all VLANs and bridge domains. If a feature requires VLAN-specific or

bridge-domain-specific configuration values, the feature cannot be applied on the trunk EFP. Those special VLANs or bridge domains must be removed from the EFP trunk to form individual EFPs.

- Trunk EFP supports a maximum of 1000 VLANs.

RSP3 Module:

- L2 port will start dropping untagged traffic when untagged/default/ptag EFP is not configured and it may start impacting any control plane protocol which requires untagged traffic to be processed. If that happens, you need to explicitly configure the untagged EFP. For example, LACP.

Restrictions for Trunk EFP with Encapsulation from Bridge Domain

- When an EFP is created on an interface followed by a TEFP with encapsulation from bridge domain (BD), all the BDs in the switch gets added to the TEFP with encapsulation from BD except the ones present in the EFP configured .
- You cannot create an EFP or TEFP after configuring TEFP with encapsulation from BD. It is recommneded that TEFP with encapsulation from BD should be the last EFP created on an interface.
- You cannot make changes to EFP after you have configured TEFP with encapsulation from BD. If you need to edit the EFP, you must first remove the TEFP with encapsulation from BD and then edit the TEFP.
- You cannot convert a TEFP into a TEFP with encapsulation from BD or vice versa.
- It is recommended to have a service instance ID of the TEFP with encapsulation from BD greater than the ID of any other EFP configured on that interface.
- You must maintain some delay when detaching and attaching the scaled TEFP with encapsulation from BD configurations.
- On an access interface having both EFP and TEFP or TEFP with encapsulation from BD configured, any data traffic with VLAN ID equal to bridge domain of EFP is flooded if the VLAN ID present in the data traffic does not match the encapsulation values present in the EFP and TEFP with encapsulation from BD.

Information About Trunk EFP Support

Benefits of Trunk EFP Support

The Carrier Ethernet infrastructure supports the following types of Ethernet flow points (EFPs):

- Static EFPs that are user-configurable.
- Dynamic EFPs that are created and maintained during a Cisco Intelligent Services Gateway (ISG) session.

With this feature, a new EFP type has been added that is intended for use on a trunk port.

A trunk port allows a range of VLANs to be forwarded on a given interface while maintaining data-plane segmentation between the VLANs.

Note Trunk EFP (with or without port channel) supports encapsulation of up to 1000 VLANs.

Like a static EFP, this new type of EFP is user-configurable via the **service instance trunk** command, the **encapsulation** command, and the **bridge-domain from-encapsulation** command when the Trunk EFP Support feature is enabled.

Ethernet Flow Points

An Ethernet flow point (EFP) is a forwarding decision point in the provider edge (PE) router, which gives network designers flexibility to make many Layer 2 flow decisions within the interface. Many EFPs can be configured on a single physical port. (The number varies from one device to another.) EFPs are the logical demarcation points of an Ethernet virtual connection (EVC) on an interface. An EVC that uses two or more user network interfaces (UNIs) requires an EFP on the associated ingress and egress interfaces of every device that the EVC passes through.

EFPs can be configured on any Layer 2 traffic port; however, they are usually configured on UNI ports. The following parameters (matching criteria) can be configured on the EFP:

- Frames of a specific VLAN, a VLAN range, or a list of VLANs (100-150 or 100,103,110)
- Frames with no tags (untagged)
- Frames with identical double-tags (VLAN tags) as specified
- Frames with identical Class of Service (CoS) values

A frame passes each configured match criterion until the correct matching point is found. If a frame does not fit any of the matching criteria, it is dropped. Default criteria can be configured to avoid dropping frames.

You can configure a new type of TEFP called TEFP with encapsulation from bridge domain (BD). All the BDs configured on the switch are part of the VLAN list of the encapsulated TEFP. The TEFP is encapsulated using the **encapsulation dot1q from-bd** command. The feature brings about the following interaction between the Ethernet-EFP and Layer2-bridge domain components:

- If BDs exist in the system and a TEFP with encapsulation from bridge domain is created, then all the BDs get added to the VLAN list of TEFP with encapsulation from bridge domain.
- If TEFP with encapsulation from bridge domain exists in the system and a new BD is created, then the BD is added to the VLAN list of all the TEFP with encapsulation from bridge domain in the system.
- If TEFP with encapsulation from bridge domain exists in the system and a BD gets deleted, and if the deleted BD is not part of an existing TEFP or EFP then it gets deleted from all the TEFP with encapsulation from bridge domain in the system.

The following types of commands can be used in an EFP:

- Rewrite commands—In each EFP, VLAN tag management can be specified with the following actions:
  - Pop—1) pops out a tag; 2) pops out two tags

- Feature commands—In each EFP, the QoS features or parameters can be changed and the ACL can be updated.

How to Enable Trunk EFP Support

Enabling Trunk EFP Support

To enable Ethernet flow point (EFP) support on a trunk port or trunk interface, complete the following steps.

Note TEFP is supported on a PC interface and on a Gigabit interface. The procedure listed below is for TEFP configuration on a PC interface. Similar procedure is used for TEFP configuration on a gigabit interface.

Note When configuring TEFP on a port-channel interface, ensure that the port interface is always up.

Procedure

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 number*

Example:

```
Device(config)# interface port-channel 1
```

Configures the interface and enters interface configuration mode.

Step 4 **service instance trunk** *id* **ethernet**

Example:

```
Device(config-if)# service instance trunk 1 ethernet
```

Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.

Step 5 **encapsulation dot1q** {**from-bd** |*vlan-id* [**,** *vlan-id* [**-** *vlan-d*]]}

Example:

```
Device(config-if-srv)# encapsulation dot1q 1-5, 7, 9-12
```

```
Device(config-if-srv)# encapsulation dot1q from-bd
```

Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance.

Step 6 **rewrite ingress tag pop 1 symmetric**

Example:

```
Device(config-if-srv)# rewrite ingress tag pop 1 symmetric
```

Specifies the encapsulation adjustment to be performed on a frame that is entering a service instance.

Step 7 **bridge-domain from-encapsulation**

Example:

```
Device(config-if-srv)# bridge-domain from-encapsulation
```

Creates a list of bridge domains for an EFP trunk port using the bridge-domain IDs derived from the encapsulation VLAN numbers.

Step 8 **no shutdown**

Example:

```
Device(config-if-srv)# no shutdown
```

Disables shutdown and keeps the interface or port active.

Step 9 **end**

Example:

```
Device(config-if-srv)# end
```

Returns to privileged EXEC mode.

Verifying the Trunk EFP Support Configuration

Use one or more of the commands listed below to verify the Trunk EFP Support feature configuration.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show ethernet service instance**

Example:

```
Device# show ethernet service instance
```

Displays information about Ethernet service instances.

Step 3 **show ethernet service instance interface port-channel** [*number*]

Example:

```
Device# show ethernet service instance interface port-channel 1
```

Displays interface-only information about Ethernet service instances for all port-channel interfaces or for a specified port-channel interface.

Step 4 **show bridge-domain**

Example:

```
Device# show bridge-domain
```

Displays bridge-domain information.

Step 5 **exit**

Example:

```
Device# exit
```

Exits privileged EXEC mode.

Configuration Examples

Example: Configuring Trunk EFP Support

In the following example, EFP support has been configured on a trunk interface.

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 1
Device(config-if)# service instance trunk 1 ethernet
Device(config-if-srv)# encapsulation dot1q 1 - 5, 7, 9 - 12
Device(config-if-srv)# rewrite ingress tag pop 1 symmetric
Device(config-if-srv)# bridge-domain from-encapsulation
```

```
Device(config-if-srv)# no shutdown
Device(config-if-srv)# end
```

Example: Configure the Trunk EFP with Encapsulation from Bridge Domain

```
Device> enable
Device# configure terminal
Device(config)#interface gigabitEthernet 0/0/0
Device(config-if)#service instance trunk 4000 eth
Device(config-if-srv)#encapsulation dot1q from-bd
Device(config-if-srv)#rewrite ingress tag pop 1 symmetric
Device(config-if-srv)#bridge-domain from-encapsulation
Device(config-if-srv)#end
```

Example: Verifying the Trunk EFP Support Configuration

The following is sample output from the **show ethernet service instance** command. The output displays trunk as the service instance type and indicates that a bridge domain for VLANs in the range of 12 to 1900 (as specified by the encapsulation parameters) has been created for service instance 4000 on a trunk port (interface).

```
Device# show ethernet service instance id 4000 interface port-channel 1

Service Instance ID: 4000
Service Instance Type: Trunk
Associated Interface Port-channel: 1
Associated EVC:
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 12-1900 vlan protocol type 0x8100
Rewrite: ingress tag pop 1 symmetric
Interface Port-channel Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
168729725 10798985220  160246675 10255787200
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 12-1900
```

Example: Verify the Trunk EFP with Encapsulation from Bridge Domain

```
Device#show ethernet service  instance id 4000 int GigabitEthernet 0/0/0 detail
Service Instance ID: 4000
Service Instance Type: Trunk
Associated Interface: GigabitEthernet0/0/0
Associated EVC:
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 2-21 vlan protocol type 0x8100
Rewrite: ingress tag pop 1 symmetric
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
2810511074 191114753032          0          0
EFP Microblocks:
****************
```

```
Microblock type: Bridge-domain
Bridge-domain: 2-21

Microblock type: L2Mcast
L2 Multicast GID: 9

Microblock type: dhcp_snoop
L2 Multicast GID: 9

Microblock type: PPPoE IA UBLOCK
PPPoE IA info
Enable: 0
Format Type: 0
cricuit id:
remote id:
```

Additional References

Related Documents

Related Topic	Document Title
Ethernet CFM	Configuring Ethernet Connectivity Fault Management in a Service Provider Network
IEEE 802.3ah	*IEEE 802.3ah Ethernet in the First Mile*
ITU-T Y.1731 fault management functions	*Configuring ITU-T Y.1731 Fault Management Functions*
Delivering and filtering syslog messages	*Reliable Delivery and Filtering for Syslog*
Cisco IOS commands: master list of commands with complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS Master Command List, All Releases
Cisco IOS Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*

Standards

Standard	Title
IEEE P802.1ag/D1.0	*Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks - Amendment 5: Connectivity Fault Management*
IETF VPLS OAM	*L2VPN OAM Requirements and Framework*
ITU-T	ITU-T Y.1731 OAM Mechanisms for Ethernet-Based Networks

MIBs

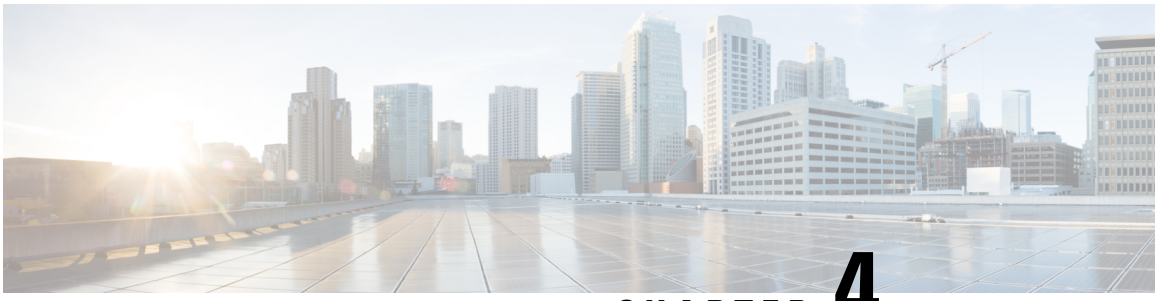
MIB	MIBs Link
• CISCO-ETHER-CFM-MIB • CISCO-IEEE-CFM-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 3164	*The BSD syslog Protocol*

Technical Assistance

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



CHAPTER 4

Ethernet Virtual Connections Configuration

An Ethernet Virtual Connection (EVC) is defined by the Metro-Ethernet Forum (MEF) as an association between two or more user network interfaces that identifies a point-to-point or multipoint-to-multipoint path within the service provider network. An EVC is a conceptual *service pipe* within the service provider network. A *bridge domain* is a local broadcast domain that is VLAN-ID-agnostic. An Ethernet flow point (EFP) service instance is a logical interface that connects a bridge domain to a physical port or to an EtherChannel group.

An EVC broadcast domain is determined by a bridge domain and the EFPs that are connected to it. You can connect multiple EFPs to the same bridge domain on the same physical interface, and each EFP can have its own matching criteria and rewrite operation. An incoming frame is matched against EFP matching criteria on the interface, learned on the matching EFP, and forwarded to one or more EFPs in the bridge domain. If there are no matching EFPs, the frame is dropped.

You can use EFPs to configure VLAN translation. For example, if there are two EFPs egressing the same interface, each EFP can have a different VLAN rewrite operation, which is more flexible than the traditional switchport VLAN translation model.

QoS policies on EFPs are supported with ingress rewrite type as push. In the ingress direction with one VLAN tag is pushed and in the egress direction one VLAN tag is popped.

This document describes how to configure EVC features.

For detailed information about the commands, see:

- The Cisco IOS XE Carrier Ethernet Command Reference: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/17_xe/command/command-references.html
- Master Command Index for Cisco IOS XE Release: http://www.cisco.com/en/US/docs/ios/mcl/allreleasemcl/all_book.html

Supported EVC Features

- Service instance—you create, delete, and modify EFP service instances on Ethernet interfaces.

- Encapsulation—you can map traffic to EFPs based on:
  - 802.1Q VLANs (a single VLAN or a list or range of VLANs)
  - 802.1Q tunneling (QinQ) VLANs (a single outer VLAN and a list or range of inner VLANs)
  - Double-tagged frames mapped to EVC based on C-tags (wildcard S-Tags)
- Bridge domains—you can configure EFPs as members of a bridge domain (up to 64 EFPs per bridge domain for bridge domain with BDIs.).
- Rewrite (VLAN translation)
  - Pop symmetric

    pop 1 removes the outermost tag

    pop 2 removes the two outermost tags

    pop symmetric adds a tag (or 2 tags for **pop 2 symmetric**) on egress for a *push* operation
  - QinQ with rewrite
- EVC forwarding
- MAC address learning and aging
- EVCs on EtherChannels
- Split horizon
- Layer 2 protocol tunneling and QinQ
- Bridging between EFPs
- MSTP (MST on EVC bridge domain)
- EFP statistics (packets and bytes)
- QoS aware EVC/EFP per service instance
- Static MAC Addresses

These Layer 2 port-based features can run with EVC configured on the port:

- LACP
- CDP
- MSTP
- EVC egress filtering

Restrictions for Ethernet Virtual Connections Configuration

- Translate operations are not supported.
- You can create a maximum of 128 EFPs per bridge-domain.

- Only dot1q encapsulation is supported on trunk EFPs.
- Ingress mapping of Differentiated Services Code Point (DSCP) or Class of Service (CoS) to the C-CoS or S-CoS is supported.
- Egress classification and queuing is based on DSCP or CoS.
- 1024 EFPs per port are supported, which fall into the category of second or higher EFP configured under any BDI on that port.
- 64 EFPs per BD is supported for BDs with BDI.
- Egress filtering is not supported for EFP with double tagged encapsulation.
- Frame forwarding of double tagged frames with both outermost and innermost tag as 0x88a8 is not supported.
- Frame forwarding of double tagged frames with outermost tag as 0x8100 and innermost tag as 0x88a8 is not supported.
- Static MAC entry does not get cleared on deleting the encapsulation dot1q VLAN.
- Classification based on multiple encapsulation types in single EFP is not supported.
- MAC ageing happens randomly, it takes around 700 seconds to age out.
- Show MAC address table output takes a maximum of 60 seconds to syncup between the hardware entries.
- Custom Ethertype is not supported.
- Hair pin switching is not supported.
- Encapsulation criteria as Ethertype is not supported.

Configuring EFPs

Default EVC Configuration

No EFPs are configured. No service instances or bridge domains are configured.

Configuration Guidelines

The following guidelines apply when you configure EVCs on the router.

- To configure a service instance on an interface, these commands are prerequisites:

```
Router (config)# interface gigabitethernet0/0/1
Router (config-if)# service instance 22 Ethernet ether
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 10
```

- You must configure encapsulation on a service instance before configuring bridge domain.
- ISL trunk encapsulation is not supported.

- The router does not support overlapping configurations on the same interface and same bridge domain. If you have configured a VLAN range encapsulation, or encapsulation default, or encapsulation any on service instance 1, you cannot configure any other encapsulations that also match previous encapsulations in the same interface and bridge domain.
- QinQ is not supported on Trunk EFP interfaces.
- Trunk EFPs should be configured with the **rewrite ingress tag pop 1 symmetric** command.
- In MST instance, when you add or remove VLAN from Trunk EFP, the BDI interface goes down which results in loss of packets.
- On an access interface configured with EFP untagged and TEFP, when a tagged packet with encapsulation equal to bridge-domain of untagged EFP passes through the access interface, the packet passes through TEFP and returns to the source through EFP untagged configuration and the packet is not dropped.

Creating Service Instances

Beginning in privileged EXEC mode, follow these steps to create an EFP service instance:

Procedure

Step 1 **configure terminal**

Enter global configuration mode.

Step 2 **interface** *interface-id*

Specify the port to attach to the policy map, and enter interface configuration mode. Valid interfaces are physical ports.

Step 3 **service instance** *number* **ethernet** [*name*]

Configure an EFP (service instance) and enter service instance configuration) mode.

- The number is the EFP identifier, an integer from 1 to 4000.
- (Optional) **ethernet** name is the name of a previously configured EVC. You do not need to use an EVC name in a service instance.

Step 4 **encapsulation** {**default** | **dot1q** | **priority-tagged** | **untagged**}

Configure encapsulation type for the service instance.

- **default**—Configure to match all unmatched packets.
- **dot1q**—Configure 802.1Q encapsulation. See for details about options for this keyword.
- **priority-tagged**—Specify priority-tagged frames, VLAN-ID 0 and CoS value of 0 to 7.
- **untagged**—Map to untagged VLANs. Only one EFP per port can have untagged encapsulation.

Step 5 **rewrite ingress tag pop** {**1** | **2**} **symmetric**

(Optional) Specify that encapsulation modification to occur on packets at ingress.

- **pop 1**—Pop (remove) the outermost tag.
- **pop 2**—Pop (remove) the two outermost tags.
- **symmetric—**Configure the packet to undergo the reverse of the ingress action at egress. If a tag is popped at ingress, it is pushed (added) at egress. This keyword is required for **rewrite** to function properly.

Step 6 **bridge-domain** *bridge-id* [**split-horizon group** *group-id*]

Configure the bridge domain ID. The range is from 1 to 4000.

You can use the **split-horizon** keyword to configure the port as a member of a split horizon group. The *group-id* range is from 0 to 2.

Step 7 **end**

Return to privileged EXEC mode.

Step 8 **show ethernet service instance show bridge-domain** [*n* | **split-horizon]**

Verify your entries.

Step 9 **copy running-config startup-config**

(Optional) Save your entries in the configuration file.

Use the **no** forms of the commands to remove the service instance, encapsulation type, or bridge domain or to disable the rewrite operation.

Creating a Trunk EFP

Beginning in privileged EXEC mode, follow these steps to create an EFP service instance:

Note Use the no forms of the commands to remove the service instance, encapsulation type, or bridge domain or to disable the rewrite operation.

Note Trunk EFPs on port-channel interfaces is supported. Traffic may *not* flow to the TEFP when the port-channel or its member links are in down state.

Procedure

Step 1 **configure terminal**

Enter global configuration mode.

Step 2 **interface** *interface-id*

Specify the port to attach to the policy map, and enter interface configuration mode. Valid interfaces are physical ports.

Step 3 **service instance** [**trunk**] *number* **ethernet**

Configure an EFP (service instance) and enter service instance configuration) mode.

- The number is the EFP identifier, an integer from 1 to 4000.
- The trunk keyword identifies the trunk ID to which the service instance is assigned.

Note Trunk EFP (without port channel) supports encapsulation of up to 1000 VLANS.

Step 4 **encapsulation** {**default** | **dot1q** | **priority-tagged** | **untagged**}

Note Only dot1q encapsulation is supported on trunk EFPs.

Configure encapsulation type for the service instance.

- **default** —Configure to match all unmatched packets.
- **dot1q** —Configure 802.1Q encapsulation. See Table 1 for details about options for this keyword.
- **priority-tagged** —Specify priority-tagged frames, VLAN-ID 0 and CoS value of 0 to 7.
- **untagged** —Map to untagged VLANs. Only one EFP per port can have untagged encapsulation.

Step 5 **rewrite ingress tag pop** {**1** | **2**} **symmetric**

(Optional) Specify that encapsulation modification to occur on packets at ingress.

- **pop 1** —Pop (remove) the outermost tag.
- **pop 2** —Pop (remove) the two outermost tags.

  Caution The **pop2** option is not currently supported on Trunk EFPs.

- **symmetric**—Configure the packet to undergo the reverse of the ingress action at egress. If a tag is popped at ingress, it is pushed (added) at egress. This keyword is required for rewrite to function properly.

Step 6 **bridge-domain** *bridge-id*

Configures the router to derive bridge domains from the encapsulation VLAN list.

Step 7 **end**

Return to privileged EXEC mode.

Step 8 Use one of the following commands

- **show ethernetservice instance**
- **show bridge-domain** *[n* | **split-horizon**]

Verify your entries.

Step 9 **copy running-config startup-config**

(Optional) Save your entries in the configuration file.

Configuration Examples

Example for Configuring a Service Instance

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 22 Ethernet ether
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 10
```

Example for Encapsulation Using a VLAN Range

Configuration Example for Larger String VLAN in Encapsulation

Configuration Example

```
  show running config

ethernet service multi-line
  !
  interface GigabitEthernet0/0/0
   service instance 1 ethernet
    encapsulation dot1q 10,13,19-21,24,29,32-36,41,46-48,55,61,63-66
    encapsulation dot1q add 69-73,78,80,83-86
   !
   service instance 2 ethernet
    encapsulation dot1q 1 second-dot1q 10,13,19-21,24,29,32-36,41
    encapsulation dot1q add outer 2-5,7
    encapsulation dot1q add inner 46-48,55,61,63-66,69-73,78,80,83-86
    encapsulation dot1q add inner 91,95-99,101
   !
   interface GigabitEthernet0/0/0
    ethernet dot1ad nni
    service instance 3 ethernet
    encapsulation dot1ad 10,13,19-21,24,29,32-36,41,46-48,55,61,63-66
    encapsulation dot1ad add 69-73,78,80,83-86
   !
   service instance 4 ethernet
    encapsulation dot1ad 1 dot1q 10,13,19-21,24,29,32-36,41,46-48,55
    encapsulation dot1ad add inner 61,63-66,69-73,78,80,83-86
   !
  !
```

Example for Two Service Instances Joining the Same Bridge Domain

In this example, service instance 1 on interfaces Gigabit Ethernet 0/0/1 and 0/0/2 can bridge between each other.

```
Router (Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 10

Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 10
```

Example for Bridge Domains and VLAN Encapsulation

Unlike VLANs, the bridge-domain number does not need to match the VLAN encapsulation number.

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 3000

Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 20
Router (config-if-srv)# bridge-domain 3000
```

However, when encapsulations do not match in the same bridge domain, traffic cannot be forwarded. In this example, the service instances on Gigabit Ethernet 0/0/1 and 0/0/2 can not forward between each other, since the encapsulations don't match (filtering criteria). However, you can use the **rewrite** command to allow communication between these two.

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# bridge-domain 3000

Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 99
Router (config-if-srv)# bridge-domain 3000
```

Example for Rewrite

In this example, a packet that matches the encapsulation will have one tag removed (popped off). The **symmetric** keyword allows the reverse direction to have the inverse action: a packet that egresses out this service instance will have the encapsulation (VLAN 10) added (pushed on).

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# rewrite ingress tag pop 1 symmetric
Router (config-if-srv)# bridge-domain 3000
```

Example for Split Horizon

In this example, service instances 1 and 2 cannot forward and receive packets from each other. Service instance 3 can forward traffic to any service instance in bridge domain 3000 since no other service instance in bridge domain 3000 is in split-horizon group 2. Service instance 4 can forward traffic to any service instance in bridge domain 3000 since it has not joined any split-horizon groups.

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# rewrite ingress pop 1 symmetric
Router (config-if-srv)# bridge-domain 3000 split-horizon group 1
Router (config-if-srv)# exit
Router (config-if)# service instance 2 Ethernet
Router (config-if-srv)# encapsulation dot1q 99
Router (config-if-srv)# rewrite ingress pop 1 symmetric
Router (config-if-srv)# bridge-domain 3000 split-horizon group 1
```

```
Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 3 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# rewrite ingress pop 1 symmetric
Router (config-if-srv)# bridge-domain 3000 split-horizon group 2
Router (config-if-srv)# exit
Router (config-if)# service instance 4 Ethernet
Router (config-if-srv)# encapsulation dot1q 99
Router (config-if-srv)# rewrite ingress pop 1 symmetric
Router (config-if-srv)# bridge-domain 3000
```

Example for Egress Filtering

In EVC switching, egress filtering is performed before the frame is sent on the egress EFP. Egress filtering ensures that when a frame is sent, it conforms to the matching criteria of the service instance applied on the ingress direction. EFP does not require egress filtering if the number of pops is the same as the number of VLANs specified in the **encapsulation** command.

Egress Filtering is not supported on the RSP3 module.

Note Specifying the **cos** keyword in the encapsulation command is relevant only in the ingress direction. For egress filtering, **cos** is ignored.

For example, consider the following configuration.

```
Router (config)# interface gigabitethernet0/1
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 20
Router (config-if-srv)# bridge-domain 19

Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 2 Ethernet
Router (config-if-srv)# encapsulation dot1q 30
Router (config-if-srv)# bridge-domain 19

Router (config)# interface gigabitethernet0/3
Router (config-if)# service instance 3 Ethernet
Router (config-if-srv)# encapsulation dot1q 10 second-dot1q 20
Router (config-if-srv)# rewrite ingress pop 1 symmetric
Router (config-if-srv)# bridge-domain 19
```

If a packet with VLAN tag 10 or 20 is received on Gigabit Ethernet 0/0/3, the ingress logical port would be service instance 3. For the frame to be forwarded on a service instance, the egress frame must match the encapsulation defined on that service instance after the rewrite is done. Service instance 1 checks for outermost VLAN 20; service instance 2 checks for VLAN 30. In this example, the frame with VLAN tags 10 and 20 can be sent to service instance 1 but not to service instance 2.

Configuring Other Features on EFPs

EFPs and EtherChannels

You can configure EFP service instances on EtherChannel port channels, but EtherChannels are not supported on ports configured with service instances. Load-balancing on port channels is based on the MAC address or IP address of the traffic flow on the EtherChannel interface.

This example configures a service instance on an EtherChannel port channel. Configuration on the ports in the port channel are independent from the service instance configuration.

```
Router (config)# interface port-channel 4
Router (config-if)# service instance 1 ethernet
Router (config-if-srv)# encapsulation untagged
Router (config-if-srv)# bridge-domain {any vlan}
Router (config-if-srv)# l2protocol peer {lacp | pagp}
```

Layer 2 Protocol Peering

For Layer 2 protocols (CDP, UDLD, LLDP, MSTP, LACP,) to peer with a neighbor on a port that has an EFP service instance configured, you need to enter the **l2 protocol peer** *protocol* service-instance configuration command on the service instance.

This example shows how to configure CDP to peer with a neighbor on a service instance:

Layer 2 Protocol Software Forwarding

Layer 2 protocol forwarding is based on the bridge domain ID and the destination MAC address.

Selecting the l2protocol forward option causes the router to flood interfaces in the same VLAN or bridge-domain with untagged or tagged BPDU packets. You can apply the l2protocol forward command to CDP, LACP, LLDP, PAGP, STP, UDLD, and VTP traffic. This is an example how to configure the l2protocol forward option:

Configuring IEEE 802.1Q Tunneling and Layer 2 Protocol Tunneling Using EFPs

Tunneling is a feature used by service providers whose networks carry traffic of multiple customers and who are required to maintain the VLAN and Layer 2 protocol configurations of each customer without impacting the traffic of other customers. The router uses EFPs to support QinQ and Layer 2 protocol tunneling.

802.1Q Tunneling (QinQ)

Service provider customers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit (4096) of the 802.1Q specification.

Using the EVCs, service providers can encapsulate packets that enter the service-provider network with multiple customer VLAN IDs (C-VLANs) and a single 0x8100 Ethertype VLAN tag with a service provider

VLAN (S-VLAN). Within the service provider network, packets are switched based on the S-VLAN. When the packets egress the service provider network onto the customer network, the S-VLAN tag is decapsulated and the original customer packet is restored.

Figure below shows the tag structures of the double-tagged packets.

In figure below, Customer A was assigned VLAN 30, and Customer B was assigned VLAN 40. Packets entering the edge switches with 802.1Q tags are double-tagged when they enter the service-provider network, with the outer tag containing VLAN ID 30 or 40, appropriately, and the inner tag containing the original VLAN number, for example, VLAN 100. Even if both Customers A and B have VLAN 100 in their networks, the traffic remains segregated within the service-provider network because the outer tag is different. Each customer controls its own VLAN numbering space, which is independent of the VLAN numbering space used by other customers and the VLAN numbering space used by the service-provider network. At the outbound port, the original VLAN numbers on the customer's network are recovered.

Method 1

In this example, for Customer A, interface is the customer-facing port, and is a trunk port facing the service provider network. For Customer B, is the customer-facing port, and is the trunk port facing the service provider network.

Customer A

For Customer A, service instance 1 on is configured with the VLAN encapsulations used by the customer: C-VLANs 1–100. These are forwarded on bridge-domain 4000. The service provider facing port is configured with a service instance on the same bridge-domain and with an **encapsulation dot1q** command matching the S-VLAN. The **rewrite ingress pop 1 symmetric** command also implies a push of the configured encapsulation on egress packets. Therefore, the original packets with VLAN tags between 1 and 100 are encapsulated with another S-VLAN (VLAN 30) tag when exiting Gigabit Ethernet port 0/0/2.

Similarly, for double- tagged (S-VLAN = 30, C-VLAN = 1–100) packets coming from the provider network, the **rewrite ingress pop 1 symmetric** command causes the outer S-VLAN tag to be popped and the original C-VLAN tagged frame to be forwarded over bridge-domain 4000 out to .

The same scenario applies to Customer B.

Customer B

Method 2

QinQ is also supported when sending packets between an EFP and a trunk EFP. The same external behavior as Method 1 can be achieved with this configuration:

Customer A

Again, service instance 1 on is configured with the VLAN encapsulations used by the customer. These are forwarded on bridge-domain 30. The service provider facing port is configured as a trunk port. The trunk port pushes a tag matching the bridge-domain that the packet is forwarded on (in this case S-VLAN 30).

For double tagged (S-VLAN = 30, C-VLAN = 1 to 100) packets coming in from the provider network, the trunk port pops the outer S-VLAN (30) and forwards the packet on that bridge-domain.

Customer B

You can also combine the customer A and B configurations, as follows:

Customer A and B

For information about the effect on cost of service (CoS) for different EFT tagging operations, see the .

Layer 2 Protocol Tunneling

Customers at different sites connected across a service-provider network need to use various Layer 2 protocols to scale their topologies to include all remote sites, as well as the local sites. STP must run properly, and every VLAN should build a proper spanning tree that includes the local site and all remote sites across the service-provider network. Cisco Discovery Protocol (CDP) must discover neighboring Cisco devices from local and remote sites.

VLAN Trunking Protocol (VTP) must provide consistent VLAN configuration throughout all sites in the customer network that are participating in VTP. Similarly, DTP, LACP, LLDP, PAgP, and UDLD can also run across the service-provider network.

When protocol tunneling is enabled, edge switches on the inbound side of the service-provider network encapsulate Layer 2 protocol packets with a special MAC address (0100.0CCD.CDD0) and send them across the service-provider network. Core switches in the network do not process these packets but forward them as normal (unknown multicast data) packets. Layer 2 protocol data units (PDUs) for the configured protocols cross the service-provider network and are delivered to customer switches on the outbound side of the service-provider network. Identical packets are received by all customer ports on the same VLANs with these results:

- Users on each of a customer's sites can properly run STP, and every VLAN can build a correct spanning tree based on parameters from all sites and not just from the local site.
- CDP discovers and shows information about the other Cisco devices connected through the service-provider network.
- VTP provides consistent VLAN configuration throughout the customer network, propagating to all switches through the service provider that support VTP.

Customers use Layer 2 protocol tunneling to tunnel BPDUs through a service-provider network without interfering with internal provider network BPDUs.

In figure below, Customer X has four switches in the same VLAN, which are connected through the service-provider network. If the network does not tunnel PDUs, switches on the far ends of the network cannot properly run STP, CDP, and other Layer 2 protocols. For example, STP for a VLAN on a switch in Customer X, Site 1, will build a spanning tree on the switches at that site without considering convergence parameters based on Customer X's switch in Site 2. This could result in the topology shown in figure below.

Figure 1: Layer 2 Protocol Tunneling

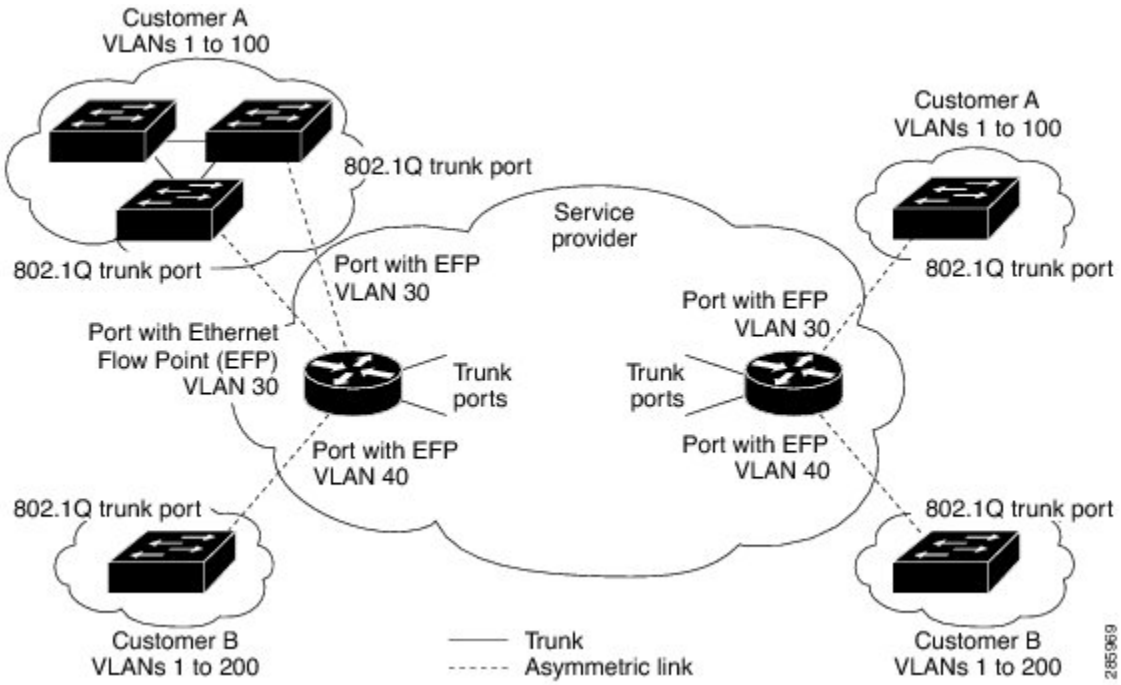
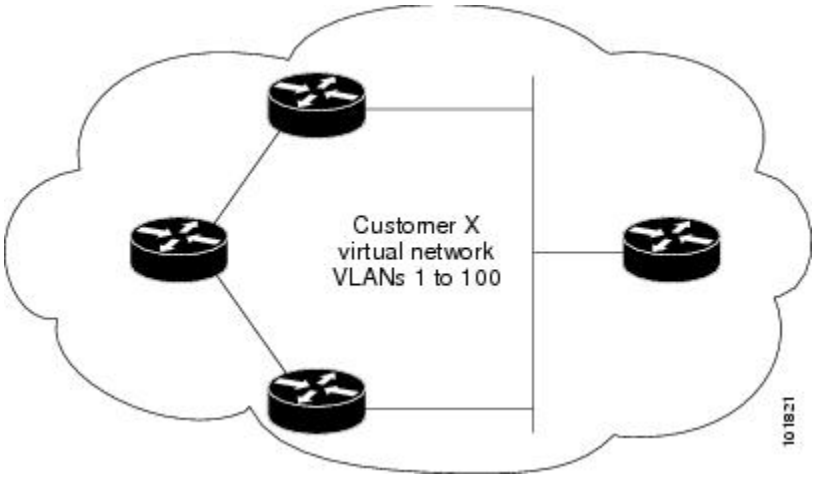


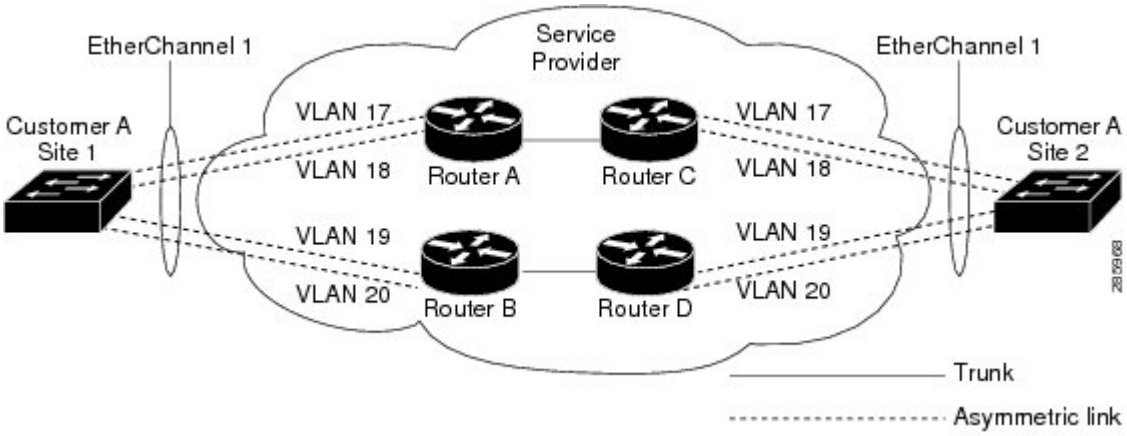
Figure 2: Layer 2 Network Topology without Proper Convergence



In a service-provider network, you can use Layer 2 protocol tunneling to enhance the creation of EtherChannels by emulating a point-to-point network topology. When you enable protocol tunneling (PAgP or LACP) on the service-provider switch, remote customer switches receive the PDUs and can negotiate the automatic creation of EtherChannels.

For example, in figure below, Customer A has two switches in the same VLAN that are connected through the SP network. When the network tunnels PDUs, switches on the far ends of the network can negotiate the automatic creation of EtherChannels without needing dedicated lines

Figure 3: Layer 2 Protocol Tunneling for EtherChannels



Use the **l2protocol tunnel***protocol* service-instance configuration command to enable Layer 2 protocol tunneling on a service instance

Valid protocols include CDP, LACP, LLDP, PAgP, STP, UDLD, and VTP. If a protocol is not specified for a service instance, the protocol frame is dropped at the interface.

This is an example of Layer 2 protocol tunneling configuration:

```
Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 10 Ethernet
Router (config-if-srv)# encapsulation untagged , dot1q 200 second-dot1q 300
Router (config-if-srv)# l2protocol tunnel cdp stp vtp pagp lacp
Router (config-if-srv)# bridge-domain 10
```

Note To enable tunneling of most Layer 2 protocol, you must configure **encapsulation untagged** because Layer 2 protocol PDUs are usually untagged.

Bridge Domain Routing

The switch supports IP routing and multicast routing for bridge domains, including Layer 3 and Layer 2 VPNs, using the BDI model. There are the limitations:

- You must configure BDIs for bridge-domain routing.
- The bridge domain must be in the range of 1 to 4094 to match the supported VLAN range.
- You can use bridge domain routing with only native packets.

  Bridge domain routing only works if proper tag popping is configured on the corresponding EFP BD. For example, if an EFP is configured with a single tag then **rewrite** should be **pop 1 symmetric**. If the EFP is configured with double tag then **rewrite** should be **pop 2 symmetric**. For double tag EFP, **pop 1 symmetric** and routing on the BDI is not supported.

Note Traffic engineering is not supported for BDI Routing.

Note You can configure the **platform bdi enable-state-up** global command to enable the BDI interface up without the **no shut** command when active. You can disable this functionality by using the **<*no*> platform bdi enable-state-up** command.

This is an example of configuring bridge-domain routing with a single tag EFP:

```
Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10
Router (config-if-srv)# rewrite ingress tag pop 1 symmetric
Router (config-if-srv)# bridge-domain 100

Router (config)# interface bdi 100
Router (config-if)# ip address 20.1.1.1 255.255.255.255
```

This is an example of configuring bridge-domain routing with two tags:

```
Router (config)# interface gigabitethernet0/2
Router (config-if)# service instance 1 Ethernet
Router (config-if-srv)# encapsulation dot1q 10 second-dot1q 20
Router (config-if-srv)# rewrite ingress tag pop 2 symmetric
Router (config-if-srv)# bridge-domain 100

Router (config)# interface bdi 100
Router (config-if)# ip address 20.1.1.1 255.255.255.255
```

EFPs and Trunk Port MAC Addresses

Because forwarding can occur between EFPs and trunk ports, MAC address movement can occur on learned addresses. Addresses learned on EFPs will have the format of interface + EFP ID, for example gigabitethernet 0/0/1 + EFP 1. When an address moves between a non-secured EFP and a trunk port, the behavior is similar to that of moving between trunk ports.

To see MAC address information for bridge domains, use the **show mac-address-table bdomain** *domain* command.

When an EFP property changes (bridge domain, rewrite, encapsulation, split-horizon, secured or unsecured, or a state change), the old dynamic MAC addresses are flushed from their existing tables. This is to prevent old invalid entries from lingering.

EFPs and MSTP

EFP bridge domains are supported by the Multiple Spanning Tree Protocol (MSTP). These restrictions apply when running STP with bridge domains.

- EVC supports only MSTP.
- All incoming VLANs (outer-most or single) mapped to a bridge domain must belong to the same MST instance or loops could occur.
- For all EFPs that are mapped to the same MST instance, you must configure backup EFPs on every redundant path to prevent loss of connectivity due to STP blocking a port.

MAC Address Forwarding, Learning and Aging on EFPs

- Layer 2 forwarding is based on the bridge domain ID and the destination MAC address. The frame is forwarded to an EFP if the binding between the bridge domain, destination MAC address, and EFP is known. Otherwise, the frame is flooded to all the EFPs or ports in the bridge domain.
- MAC address learning is based on bridge domain ID, source MAC addresses, and logical port number. MAC addresses are managed per bridge domain when the incoming packet is examined and matched against the EFPs configured on the interface. If there is no EFP configured, the bridge domain ID equal to the outer-most VLAN tag is used as forwarding and learning look-up key.

  If there is no matching entry in the Layer 2 forwarding table for the ingress frame, the frame is flooded to all the ports within the bridge domain. Flooding within the bridge domain occurs for unknown unicast, unknown multicast, and broadcast.
- Dynamic addresses are addresses learned from the source MAC address when the frame enters the router. All unknown source MAC addresses are sent to the CPU along with ingress logical port number and bridge domain ID for learning. Once the MAC address is learned, the subsequent frame with the destination MAC address is forwarded to the learned port. When a MAC address moves to a different port, the Layer 2 forwarding entry is updated with the corresponding port.

Note The router does not currently support the **no mac address-table** learning bridge-domain *bridge-id* global configuration command.

- Dynamic addresses are aged out if there is no frame from the host with the MAC address. If the aged-out frame is received by the switch, it is flooded to the EFPs in the bridge domain and the Layer 2 forwarding entry is created again. The default for aging dynamic addresses is 5 minutes. However, when MST undergoes a topology change, the aging time is reduced to the *forward-delay* time configured by the spanning tree. The aging time reverts back to the last configured value when the topology change expires.

  You can configure a dynamic address aging time per bridge domain using the **mac aging-time** *time* command. The range is in seconds and valid values are 120-300. The default value is 300. An aging time of 0 means that the address aging is disabled.
- MAC address movement is detected when the host moves from one port to another. If a host moves to another port or EFP, the learning lookup for the installed entry fails because the ingress logical port number does not match and a new learning cache entry is created. The detection of MAC address movement is disabled for static MAC addresses where the forwarding behavior is configured by the user.

Configuring a Static MAC Address

This section describes how to configure a static MAC address on the router.

Static MAC Addresses

The router supports multicast static MAC addresses, which allow you to enable multicast at the layer 2 level. You can use multicast static MAC addresses to forward multicast packets to specific EFPs on a network.

Limitations

The following limitations apply when configuring static MAC addresses:

- Static MAC addresses are supported only on egress ports.
- You can configure up to 1024 multicast static MAC addresses.
- You can assign up to 24 EFPs to a bridge domain configured with a multicast static MAC address.
- MAC entries configured across different bridge-domains are represented as separate entries in the router MAC table.
- Multicast static MAC addresses apply only to layer 2 traffic; layer 3 multicast traffic is not affected by a static MAC configuration and is forwarded to all EFPs in a bridge domain.

Configuring a Static MAC Address

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enter global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specify the port to attach to the policy map, and enter interface configuration mode. Valid interfaces are physical ports.
Step 3	**no ip address** **Example:** `Router(config-if)# no ip address`	To set a primary or secondary IP address for an interface, use the **ip address** interface configuration command. To remove an IP address or disable IP processing, use the **no** form of this command.
Step 4	**no negotiation auto** **Example:** `Router(config-if)# no negotiation auto`	Disables autonegotiation on Gigabit Ethernet interfaces.
Step 5	**service instance** *number* **ethernet** [*name*] **Example:** `Rotuer(config-if)# service instance 1 ethernet`	Configure an EFP (service instance) and enter service instance configuration) mode. • The number is the EFP identifier, an integer from 1 to 4000. • (Optional) **ethernet** name is the name of a previously configured EVC. You do not need to use an EVC name in a service instance.

	Command or Action	Purpose
Step 6	**encapsulation** {**default** | **dot1q** | **priority-tagged** | **untagged**} **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Configure encapsulation type for the service instance. • **default**—Configure to match all unmatched packets. • **dot1q**—Configure 802.1Q encapsulation. See for details about options for this keyword. • **priority-tagged**—Specify priority-tagged frames, VLAN-ID 0 and CoS value of 0 to 7. • **untagged**—Map to untagged VLANs. Only one EFP per port can have untagged encapsulation.
Step 7	**bridge-domain** *bridge-id* [**split-horizon group** *group-id*] **Example:** `Router(config-if-srv)# bridge-domain 100`	Configure the bridge domain ID. The range is from 1 to 4000. You can use the **split-horizon** keyword to configure the port as a member of a split horizon group. The *group-id* range is from 0 to 2.
Step 8	**mac static address** *address* **Example:** `Router(config-if-srv)# mac static address 0000.bbbb.cccc`	Specifies the multicast MAC address.
Step 9	**end** **Example:** `Router(config-if-srv)# end`	Return to privileged EXEC mode.

Monitoring EVC

Table 1: Supported show Commands

	Description
show ethernet service evc [**id** *evc-id* | **interface** *interface-id*] [**detail**]	Displays information about all EVCs, or a specific EVC when you enter ID, or all EVCs on an interface when you enter an interface ID. The **deta** provides additional information about the EVC.
show ethernet service instance [**id** *instance-id* **interface** *interface-id* | **interface** *interface-id*] {[**detail**] | [*stats*]}	Displays information about one or more service instance (EFPs). If you s EFP ID and interface, only data pertaining to that particular EFP is displa specify only an interface ID, data is displayed for all EFPs on the interfac

	Description
show bridge-domain [*n*]	When you enter *n*, this command displays all the members of the spec bridge-domain, if a bridge-domain with the specified number exists. If you do not enter *n*, the command displays all the members of all br in the system.
show bridge-domain *n* **split-horizon** [**group** {*group_id* | **all**}]	When you do not specify a **group** *group_id*, this command displays all of bridge-domain *n* that belong to split horizon group 0. If you specify a numerical *group_id*, this command displays all the m specified group id. When you enter **group all**, the command displays all members of any group.

show ethernet service instance detail	This command displays detailed service instance information, includi protocol information. This is an example of the output:
	`Router# show ethernet service instance detail`
	`Service Instance ID: 1`
	`Associated Interface: Ethernet0/0`
	`Associated EVC:`
	`L2protocol tunnel pagp`
	`CE-Vlans:`
	`State: Up`
	`EFP Statistics:`
	`Pkts In Bytes In Pkts Out Bytes Out`
	`0 0 0 0`

show mac address-table	This command displays dynamically learned or statically configured M addresses.
show mac address-table bridge-domain *bridge-domain id*	This command displays MAC address table information for the speci domain.
show mac address-table count bridge-domain *bridge-domain id*	This command displays the number of addresses present for the speci domain.
show mac address-table learning bridge-domain *bridge-domain id*	This command displays the learning status for the specified bridge do

This is an example of output from the **show ethernet service instance detail** command:

```
Router#
Service Instance ID: 1
Associated Interface:
Associated EVC: EVC_P2P_10
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 10 vlan protocol type 0x8100
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
```

```
       214      15408      97150    6994800
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 10
```

This is an example of output from the **show bridge-domain** command:

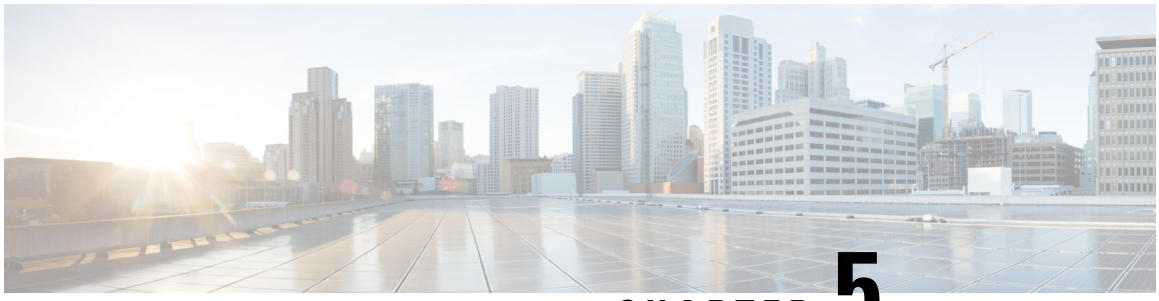
```
Router# show bridge-domain 100
Bridge-domain 100 (1 ports in all)
State: UP Mac learning: Enabled
Aging-Timer: 300 second(s)
Maximum address limit: 256000
GigabitEthernet0/0/0 service instance 1

Nile Mac Address Entries

BD mac addr type ports
------------------------------------------------------------------------------------------------------
100 0000.bbbb.cccc STATIC Gi0/0/0.Efp1

sh mac-address-table bdomain 100

Nile Mac Address Entries

BD mac addr type ports
------------------------------------------------------------------------------------------------------
100 0000.bbbb.cccc STATIC Gi0/0/0.Efp1
```

This is an example of output from the **show ethernet service instance** statistics command:

```
Router#
Service Instance 1, Interface
Pkts In   Bytes In   Pkts Out  Bytes Out
       214      15408      97150    6994800
```

This is an example of output from the **show mac-address table count** command:

```
Router# show mac address-table count bdomain 10

Mac Entries for BD   10:
---------------------------
Dynamic Address Count  : 20
Static  Address Count  : 0
Total Mac Addresses    : 20
```



CHAPTER 5

Configuring Ethernet Connectivity Fault Management in a Service Provider Network

Ethernet Connectivity Fault Management (CFM) is an end-to-end per-service-instance Ethernet layer operations, administration, and maintenance (OAM) protocol. It includes proactive connectivity monitoring, fault verification, and fault isolation for large Ethernet metropolitan-area networks (MANs) and WANs.

The advent of Ethernet as a MAN and WAN technology imposes a new set of OAM requirements on Ethernet's traditional operations, which were centered on enterprise networks only. The expansion of Ethernet technology into the domain of service providers, where networks are substantially larger and more complex than enterprise networks and the user base is wider, makes operational management of link uptime crucial. More importantly, the timeliness in isolating and responding to a failure becomes mandatory for normal day-to-day operations, and OAM translates directly to the competitiveness of the service provider.

Prerequisites for Configuring Ethernet CFM in a Service Provider Network

Business Requirements

- Network topology and network administration have been evaluated.
- Business and service policies have been established.
- Partial Route Computation (PRC) codes have been implemented for all supported commands related to configuring High Availability (HA) on a maintenance endpoint (MEP), maintenance intermediate point (MIP), level, service instance ID, cross-check timer, cross-check, and domain.

Restrictions for Configuring Ethernet CFM in a Service Provider Network

- CFM is supported *only* on EFP BD with no support on MPLS or Xconnect or VRF.
- CFM is not supported over trunk interface.
- Maintenance endpoints (MEP) statistics for hardware offloaded session do not work.
- We cannot have Port-MEP and MEP over untagged EFP at the same time on the same interface. This is true for default EFP as well, if CFM encapsulation command is not used.
- Hardware offloaded continuity check messages (CCM) intervals are not accurately displayed in the CFM database.
- For Port-MEP, untagged EFP is mandatory. It should be configured for directily connected interface.
- On a port-channel interface with untagged EFP configured, default CFM encapsulation configuration is not recommended.
- Sequence number for hardware offload session is always zero.
- UP MEP hardware session is supported from 16.9.1 release.
- UP MEP hardware CFM packets will be classified under qos-group 0 in egress policy.
- Double tag EFP without rewrite is not supported.
- Port-MEP cannot be configured under port-channel member interface.
- Error counters and output drops are seen in up MEP configured interface.
- MAC address entry for down MEP is not shown in the **show mac-address** table.
- MEP and MIP should not be configured under the same EFP.
- It is not recommended to configure MIPs for hardware offloaded CFM sessions.
- Maximum number of CFM sessions supported system wide is 300.
- Maximum number of CFM sessions per 1G interface is 40.
- Maximum number of CFM sessions per 10G interface is 300.
- CFM UP MEP session is not supported when access and core is configured as DOT1AD NNI.
- CFM UP MEP session is not supported when core is Dot1ad NNI and access as UNI-C.
- Both Software and Hardware CFM session should not be configured under the same EFP.
- Port-MEP and MEP over untagged EFP cannot be configured on the same interface at a time. This is true for default EFP as well, provided CFM encapsulation command is not used.
- Port MEP session should be configured for directly connected interface.
- Port MEP is not supported on port-channel members.
- CFM MIP level dynamic modification is not supported, you need to remove and add new MIP level.

- If UP MEP CFM session configured on a physically down interface, RMEP will not be learnt till the interface comes up.
- CFM over encapsulation priority tagged is not supported.

CFM Configuration over EFP Interface

Ethernet Connectivity Fault Management (CFM) is an end-to-end per-service-instance Ethernet layer OAM protocol that includes proactive connectivity monitoring, fault verification, and fault isolation. Currently, Ethernet CFM supports Up facing and Down facing Maintenance Endpoints (MEPs).

Information About Configuring Ethernet CFM in a Service Provider Network

Ethernet CFM

Ethernet CFM is an end-to-end per-service-instance Ethernet layer OAM protocol that includes proactive connectivity monitoring, fault verification, and fault isolation. End to end can be PE to PE or CE to CE. A service can be identified as a service provider VLAN (S-VLAN) or an EVC service.

Being an end-to-end technology is the distinction between CFM and other metro-Ethernet OAM protocols. For example, MPLS, ATM, and SONET OAM help in debugging Ethernet wires but are not always end-to-end. 802.3ah OAM is a single-hop and per-physical-wire protocol. It is not end to end or service aware.

Troubleshooting carrier networks offering Ethernet Layer 2 services is challenging. Customers contract with service providers for end-to-end Ethernet service and service providers may subcontract with operators to provide equipment and networks. Compared to enterprise networks, where Ethernet traditionally has been implemented, these constituent networks belong to distinct organizations or departments, are substantially larger and more complex, and have a wider user base. Ethernet CFM provides a competitive advantage to service providers for which the operational management of link uptime and timeliness in isolating and responding to failures is crucial to daily operations.

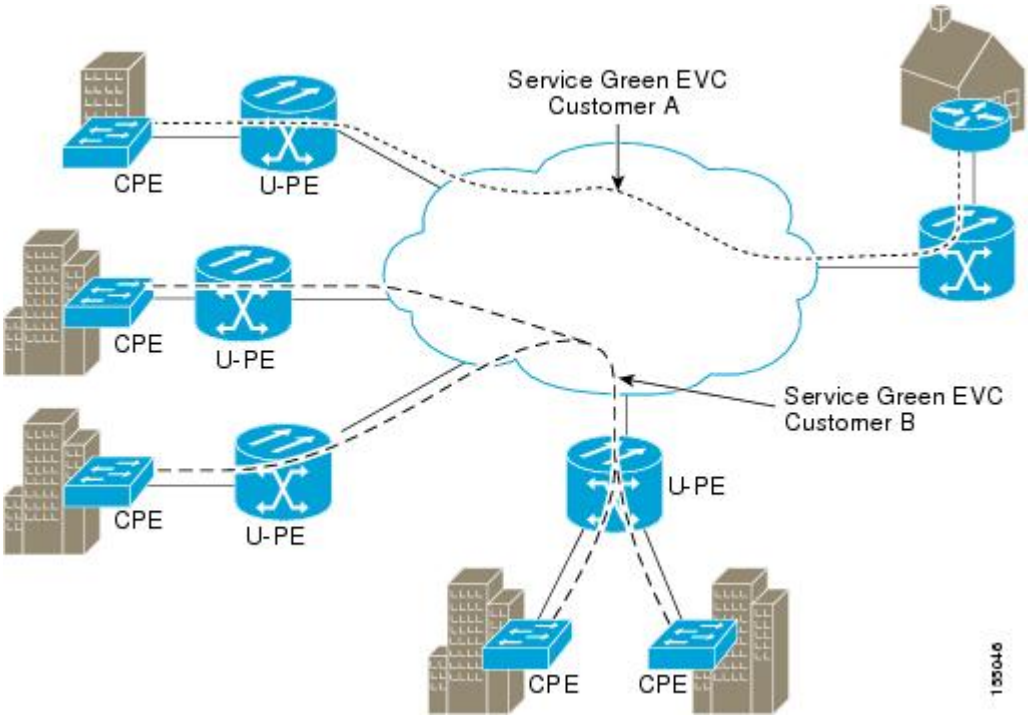
Benefits of Ethernet CFM

- End-to-end service-level OAM technology
- Reduced operating expense for service provider Ethernet networks
- Competitive advantage for service providers
- Supports both distribution and access network environments with the outward facing MEPs enhancement

Customer Service Instance

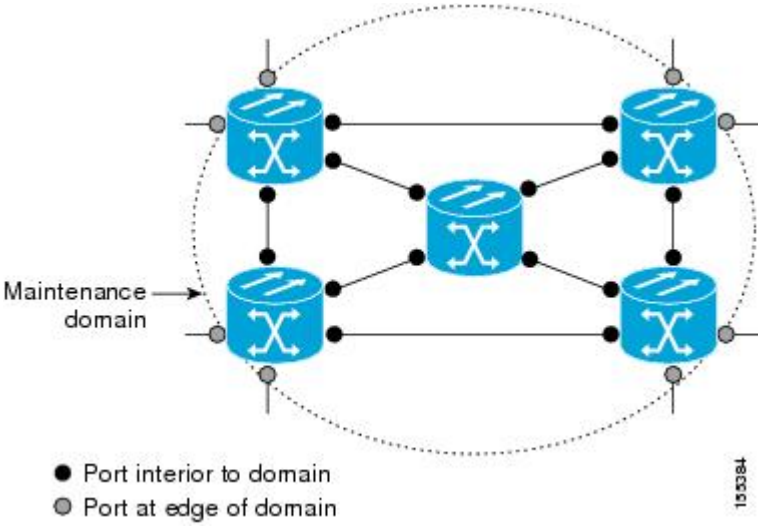
A customer service instance is an Ethernet virtual connection (EVC), which is identified by an S-VLAN within an Ethernet island, and is identified by a globally unique service ID. A customer service instance can be

point-to-point or multipoint-to-multipoint. The figure below shows two customer service instances. Service Instance Green is point to point; Service Instance Blue is multipoint to multipoint.



Maintenance Domain

A maintenance domain is a management space for the purpose of managing and administering a network. A domain is owned and operated by a single entity and defined by the set of ports internal to it and at its boundary. The figure below illustrates a typical maintenance domain.



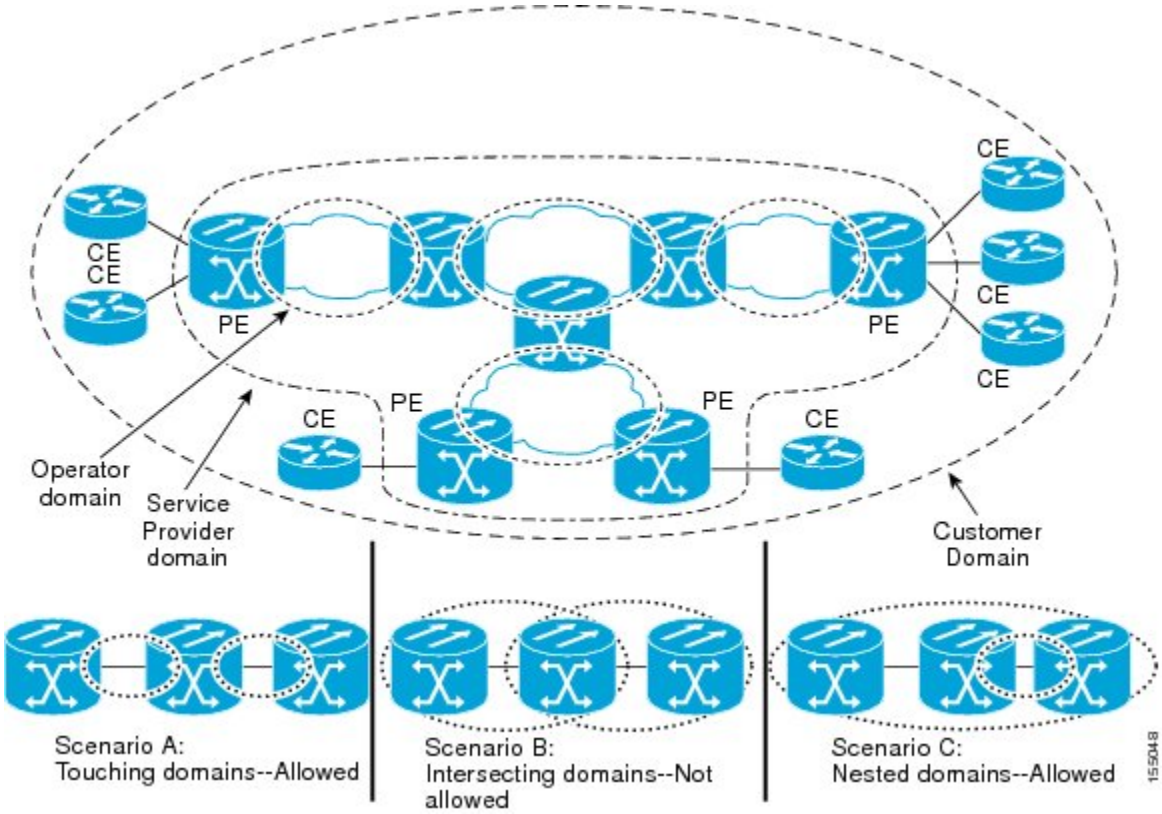
A unique maintenance level in the range of 0 to 7 is assigned to each domain by a network administrator. Levels and domain names are useful for defining the hierarchical relationship that exists among domains. The

hierarchical relationship of domains parallels the structure of customer, service provider, and operator. The larger the domain, the higher the level value. For example, a customer domain would be larger than an operator domain. The customer domain may have a maintenance level of 7 and the operator domain may have a maintenance level of 0. Typically, operators would have the smallest domains and customers the largest domains, with service provider domains between them in size. All levels of the hierarchy must operate together.

Domains should not intersect because intersecting would mean management by more than one entity, which is not allowed. Domains may nest or touch but when two domains nest, the outer domain must have a higher maintenance level than the domain nested within it. Nesting maintenance domains is useful in the business model where a service provider contracts with one or more operators to provide Ethernet service to a customer. Each operator would have its own maintenance domain and the service provider would define its domain—a superset of the operator domains. Furthermore, the customer has its own end-to-end domain which is in turn a superset of the service provider domain. Maintenance levels of various nesting domains should be communicated among the administering organizations. For example, one approach would be to have the service provider assign maintenance levels to operators.

CFM exchanges messages and performs operations on a per-domain basis. For example, running CFM at the operator level does not allow discovery of the network by the higher provider and customer levels.

Network designers decide on domains and configurations. The figure below illustrates a hierarchy of operator, service provider, and customer domains and also illustrates touching, intersecting, and nested domains.



Maintenance Associations and Maintenance Points

A maintenance association (MA) identifies a service that can be uniquely identified within the maintenance domain. The CFM protocol runs within a maintenance association. A maintenance point is a demarcation

point on an interface that participates in CFM within a maintenance domain. Maintenance points drop all lower-level frames and forward all higher-level frames. There are two types of maintenance points:

- Maintenance end points (MEPs) are points at the edge of the domain that define the boundaries and confine CFM messages within these boundaries. Outward facing or Down MEPs communicate through the wire side (connected to the port). Inward facing or Up MEPs communicate through the relay function side, not the wire side.

  CFM 802.1ag supports up and down per-VLAN MEPs, as well as port MEPs, which are untagged down MEPs that are not associated with a VLAN.

  Port MEPs are configured to protect a single hop and used to monitor link state through CFM. If a port MEP is not receiving continuity check messages from its peer (static remote MEP), for a specified interval, the port is put into an operational down state in which only CFM and OAM packets pass through, and all other data and control packets are dropped.

  - **Up MEP**—An up MEP sends and receives CFM frames through the relay function. It drops all CFM frames at its level or lower that come from the wire side, except traffic going to the down MEP. For CFM frames from the relay side, it processes the frames at its level and drops frames at a lower level. The MEP transparently forwards all CFM frames at a higher level, regardless of whether they are received from the relay or wire side. If the port on which MEP is configured is blocked by STP, the MEP cannot send or receive CFM messages through the relay function. CFM runs at the provider maintenance level (UPE-to-UPE), specifically with up MEPs at the user network interface (UNI).
  - **Down MEP**—A down MEP sends and receives CFM frames through the wire connected to the port on which the MEP is configured. It drops all CFM frames at its level or lower that come from the relay side. For CFM frames from the wire side, it processes all CFM frames at its level and drops CFM frames at lower levels except traffic going to the other lower-level down MEP. The MEP transparently forwards all CFM frames at a higher level, regardless of whether they are received from the relay or through the wire.

- Maintenance intermediate points (MIPs) are internal to a domain, not at the boundary, and respond to CFM only when triggered by traceroute and loopback messages. They forward CFM frames received from MEPs and other MIPs, drop all CFM frames at a lower level (if MIP filtering is enabled), and forward all CFM frames at a higher level and at a lower level and regardless of whether they are received from the relay or wire side. When MIP filtering is enabled, the MIP drops CFM frames at a lower level. MIPs also catalog and forward continuity check messages (CCMs), but do not respond to them.

  MIP filtering is disabled by default, and you can configure it to be enabled or disabled. When MIP filtering is disabled, all CFM frames are forwarded.

  You can manually configure a MIP or configure the device to automatically create a MIP. You can configure a MEP without a MIP. In case of a configuration conflict, manually created MIPs take precedence over automatically created MIPs.

Note MIP filtering and MIP auto-create is not supported.

Maintenance Point

A maintenance point is a demarcation point on an interface (port) that participates in CFM within a maintenance domain. Maintenance points on device ports act as filters that confine CFM frames within the bounds of a domain by dropping frames that do not belong to the correct level. Maintenance points must be explicitly configured on Cisco devices. Two classes of maintenance points exist, MEPs and MIPs.

Maintenance Endpoints

Maintenance endpoints (MEPs) have the following characteristics:

- Per maintenance domain (level) and service (S-VLAN or EVC)
- At the edge of a domain, define the boundary
- Within the bounds of a maintenance domain, confine CFM messages
- When configured to do so, proactively transmit Connectivity Fault Management (CFM) continuity check messages (CCMs)
- At the request of an administrator, transmit traceroute and loopback messages

Inward Facing MEPs

Inward facing means the MEP communicates through the Bridge Relay function and uses the Bridge-Brain MAC address. An inward facing MEP performs the following functions:

- Sends and receives CFM frames at its level through the relay function, not via the wire connected to the port on which the MEP is configured.
- Drops all CFM frames at its level (or lower level) that come from the direction of the wire.
- Processes all CFM frames at its level coming from the direction of the relay function.
- Drops all CFM frames at a lower level coming from the direction of the relay function.
- Transparently forwards all CFM frames at a higher level, independent of whether they come in from the relay function side or the wire side.

Note A MEP of level L (where L is less than 7) requires a MIP of level M > L on the same port; hence, CFM frames at a level higher than the level of the MEP will be catalogued by this MIP.

- If the port on which the inward MEP is configured is blocked by Spanning-Tree Protocol, the MEP can no longer transmit or receive CFM messages.

Outward Facing MEPs

Outward facing means that the MEP communicates through the wire.

An outward facing MEP performs the following functions:

- Sends and receives CFM frames at its level via the wire connected to the port where the MEP is configured.
- Drops all CFM frames at its level (or at a lower level) that come from the direction of the relay function.

- Processes all CFM frames at its level coming from the direction of the wire.
- Drops all CFM frames at a lower level coming from the direction of the wire.
- Transparently forwards all CFM frames at levels higher than the level of the outward facing MEP, independent of whether they come in from the relay function side or the wire side.
- If the port on which the outward MEP is configured is blocked by the Spanning-Tree Protocol, the MEP can still transmit and receive CFM messages via the wire.

Maintenance Intermediate Points

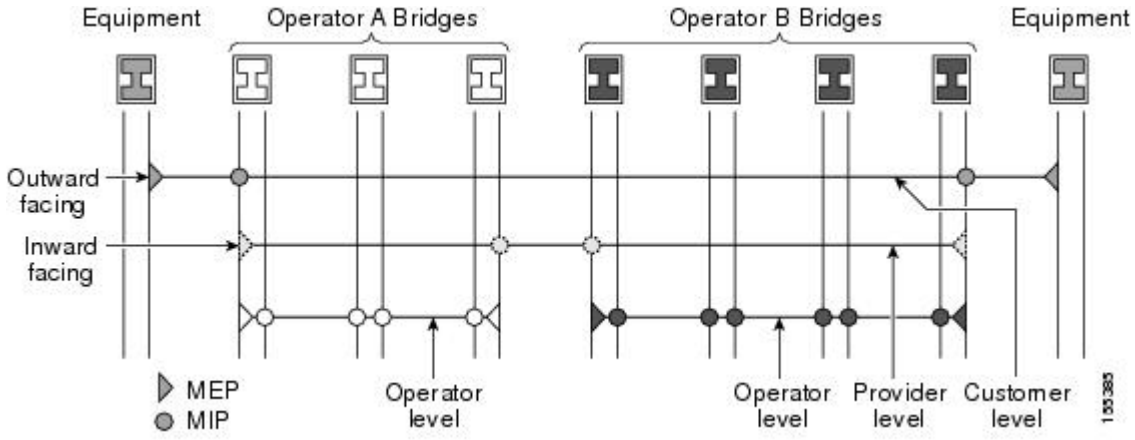
MIPs have the following characteristics:

- Per maintenance domain (level) and for all S-VLANs enabled or allowed on a port.
- Internal to a domain, not at the boundary.
- CFM frames received from MEPs and other MIPs are cataloged and forwarded, using both the wire and the relay function.
- All CFM frames at a lower level are stopped and dropped, independent of whether they originate from the wire or relay function.
- All CFM frames at a higher level are forwarded, independent of whether they arrive from the wire or relay function.
- MIPs respond only when triggered by CFM traceroute and loopback messages.
- Bridge-Brain MAC addresses are used.

If the port on which a MIP is configured is blocked by Spanning-Tree Protocol, the MIP cannot receive CFM messages or relay them toward the relay function side. The MIP can, however, receive and respond to CFM messages from the wire.

A MIP has only one level associated with it and the command-line interface (CLI) does not allow you to configure a MIP for a domain that does not exist.

The figure below illustrates MEPs and MIPs at the operator, service provider, and customer levels.



CFM Messages

CFM uses standard Ethernet frames. CFM frames are distinguishable by EtherType and for multicast messages by MAC address. CFM frames are sourced, terminated, processed, and relayed by bridges. Routers can support only limited CFM functions.

Bridges that cannot interpret CFM messages forward them as normal data frames. All CFM messages are confined to a maintenance domain and to an S-VLAN (PE-VLAN or Provider-VLAN). Three types of messages are supported:

- Continuity Check
- Loopback
- Traceroute

Continuity Check Messages

CFM CCMs are heartbeat messages exchanged periodically among MEPs. They allow MEPs to discover other MEPs within a domain and allow MIPs to discover MEPs. CCMs are confined to a domain and S-VLAN.

Table 2: Feature History Table

Feature Name	Release Information	Description
Unicast MAC for CCM Messages	Cisco IOS XE Bengaluru 17.6.1	Continuity Check Messages (CCM) use multicast destination MAC address by default. This feature enables you to unicast CCM messages to a specific remote MEP (RMEP) to avoid unnecessary traffic flood on the VLAN.

Effective Cisco IOS XE Bengaluru 17.6.1, you can override the multicast function and enable this feature to unicast CCM to destination Remote MEP.

CFM CCMs have the following characteristics:

- Transmitted at a configurable periodic interval by MEPs. The interval can be from 10 seconds to 65535 seconds, the default is 10.
- Contains a configurable hold-time value to indicate to the receiver the validity of the message. The default is 3.5 times the transmit interval.
- Catalogued by MIPs at the same maintenance level.
- Terminated by remote MEPs at the same maintenance level.
- Unidirectional and do not solicit a response.
- Carry the status of the port on which the MEP is configured.

Restrictions for Unicast MAC for CCM

- We recommend using the interface MAC address of the destination node for configuring MEP.
- Configure the correct destination MAC address to ensure reachability of unicast CCM.

- NCS 520 series Ethernet Access Devices nodes cannot interoperate with other platforms running on multicast mode.

Loopback Messages

CFM loopback messages are unicast frames that a MEP transmits, at the request of an administrator, to verify connectivity to a particular maintenance point. A reply to a loopback message indicates whether a destination is reachable but does not allow hop-by-hop discovery of the path. A loopback message is similar in concept to an Internet Control Message Protocol (ICMP) Echo (ping) message.

A CFM loopback message can be generated on demand using the CLI. The source of a loopback message must be a MEP; the destination may be a MEP or a MIP. CFM loopback messages are unicast; replies to loopback messages also are unicast. CFM loopback messages specify the destination MAC address, VLAN, and maintenance domain.

Traceroute Messages

CFM traceroute messages are multicast frames that a MEP transmits, at the request of an administrator, to track the path (hop-by-hop) to a destination MEP. They allow the transmitting node to discover vital connectivity data about the path, and allow the discovery of all MIPs along the path that belong to the same maintenance domain. For each visible MIP, traceroute messages indicate ingress action, relay action, and egress action. Traceroute messages are similar in concept to User Datagram Protocol (UDP) traceroute messages.

Traceroute messages include the destination MAC address, VLAN, and maintenance domain and they have Time To Live (TTL) to limit propagation within the network. They can be generated on demand using the CLI. Traceroute messages are multicast; reply messages are unicast.

Ethernet CFM and Ethernet OAM Interaction

To understand how CFM and OAM interact, you should understand the following concepts:

Ethernet Virtual Circuit

An EVC as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint Layer 2 circuit. EVC status can be used by a CE device either to find an alternative path in to the service provider network or in some cases, to fall back to a backup path over Ethernet or over another alternative service such as ATM.

OAM Manager

The OAM manager is an infrastructure element that streamlines interaction between OAM protocols. The OAM manager requires two interworking OAM protocols, in this case Ethernet CFM and Ethernet OAM. Interaction is unidirectional from the OAM manager to the CFM protocol and the only information exchanged is the user network interface (UNI) port status. Additional port status values available include

- REMOTE_EE—Remote excessive errors
- LOCAL_EE—Local excessive errors
- TEST—Either remote or local loopback

After CFM receives the port status, it communicates that status across the CFM domain.

CFM over Bridge Domains

Connectivity Fault Management (CFM) over bridge domains allows untagged CFM packets to be associated with a maintenance end point (MEP). An incoming untagged customer CFM packet has an EtherType of CFM and is mapped to an Ethernet virtual circuit (EVC) or bridge domain based on the encapsulation configured on the Ethernet flow point (EFP). The EFP is configured specifically to recognize these untagged packets.

An EFP is a logical demarcation point of an EVC on an interface and can be associated with a bridge domain. The VLAN ID is used to match and map traffic to the EFP. VLAN IDs have local significance per port similar to an ATM virtual circuit. CFM is supported on a bridge domain associated with an EFP. The association between the bridge domain and the EFP allows CFM to use the encapsulation on the EFP. All EFPs in the same bridge domain form a broadcast domain. The bridge domain ID determines the broadcast domain.

The distinction between a VLAN port and the EFP is the encapsulation. VLAN ports use a default dot1q encapsulation. For EFPs, untagged, single tagged, and double tagged encapsulation exists with dot1q and IEEE dot1ad EtherTypes. Different EFPs belonging to the same bridge domain can use different encapsulations.

Both up MEP, down MEP and MIP are supported. If an up MEP is configured under an EFP within a bridge domain, CFM messages would be routed into the bridge, and the rest members of the same bridge domain would be able to receive messages from this MEP. If a down MEP is configured, the messages will not goes into the bridge domain.

How to Set Up Ethernet CFM in a Service Provider Network

Designing CFM Domains

Note To have an operator, service provider, or customer domain is optional. A network may have a single domain or multiple domains. The steps listed here show the sequence when all three types of domains will be assigned.

Before you begin

- Knowledge and understanding of the network topology.
- Understanding of organizational entities involved in managing the network; for example, operators, service providers, network operations centers (NOCs), and customer service centers.
- Understanding of the type and scale of services to be offered.
- Agreement by all organizational entities on the responsibilities, roles, and restrictions for each organizational entity.
- Determination of the number of maintenance domains in the network.
- Determination of the nesting and disjoint maintenance domains.
- Assignment of maintenance levels and names to domains based on agreement between the service provider and operator or operators.
- Determination of whether the domain should be inward or outward.

Procedure

Step 1 Determine operator level MIPs.

Follow these steps:

- Starting at lowest operator level domain, assign a MIP at every interface internal to the operator network to be visible to CFM.
- Proceed to next higher operator level and assign MIPs.
- Verify that every port that has a MIP at a lower level does not have maintenance points at a higher level.
- Repeat steps a through d until all operator MIPs are determined.

Step 2 Determine operator level MEPs.

Follow these steps:

- Starting at the lowest operator level domain, assign a MEP at every UNI that is part of a service instance.
- Assign a MEP at the network to network interface (NNI) between operators, if there is more than one operator.
- Proceed to next higher operator level and assign MEPs.
- A port with a MIP at a lower level cannot have maintenance points at a higher level. A port with a MEP at a lower level should have either a MIP or MEP at a higher level.

Step 3 Determine service provider MIPs.

Follow these steps:

- Starting at the lowest service provider level domain, assign service provider MIPs at the NNI between operators (if more than one).
- Proceed to next higher service provider level and assign MIPs.
- A port with a MIP at a lower level cannot have maintenance points at a higher level. A port with a MEP at a lower level should not have either a MIP or a MEP at a higher level.

Step 4 Determine service provider MEPs.

Follow these steps:

- Starting at the lowest service provider level domain, assign a MEP at every UNI that is part of a service instance.
- Proceed to next higher service provider level and assign MEPs.
- A port with a MIP at a lower level cannot have maintenance points at a higher level. A port with a MEP at a lower level should have either a MIP or a MEP at a higher level.

Step 5 Determine customer MIPs.

Customer MIPs are allowed only on the UNIs at the uPEs if the service provider allows the customer to run CFM. Otherwise, the service provider can configure Cisco devices to block CFM frames.

- Configure a MIP on every uPE, at the UNI port, in the customer maintenance domain.
- Ensure the MIPs are at a maintenance level that is at least one higher than the highest level service provider domain.

Step 6 Determine customer MEPs.

Customer MEPs are on customer equipment. Assign an outward facing MEP within an outward domain at the appropriate customer level at the handoff between the service provider and the customer.

Configuring Ethernet CFM

Configuring Ethernet CFM consists of the following tasks:

Configuring CFM

Procedure

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 **ethernet cfm domain** *domain-name* **level** *level-id*

Example:

```
Device(config)# ethernet cfm domain Customer level 7
```

Defines a CFM maintenance domain at a particular maintenance level and enters Ethernet CFM configuration mode.

Step 4 **service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down**

Example:

```
Device(config-ecfm)# service s41 evc 41 vlan 41 direction down
```

Configures a maintenance association within a maintenance domain and enters Ethernet connectivity fault management (CFM) service configuration mode.

Note The **direction down** is used only for Down or Outward-facing MEPs. For Up MEPs or Inward-facing MEPs, do not specify **direction down**.

Step 5 **continuity-check**

Example:

```
Device(config-ecfm-srv)# continuity-check
```

Enables the transmission of continuity check messages (CCMs).

Step 6 **continuity-check** [**interval** *cc-interval*]

Example:

```
Device(config-ecfm-srv)# continuity-check interval 10s
```

Configures the time period between CCMs transmission. The default interval is 10 seconds.

Step 7 **exit**

Example:

```
Device(config-ecfm-srv)# exit
```

Returns to Ethernet connectivity fault management configuration mode.

Step 8 **mep archive-hold-time** *minutes*

Example:

```
Device(config-ecfm)# mep archive-hold-time 60
```

Sets the amount of time that data from a missing MEP is kept in the continuity check database or that entries are held in the error database before they are purged.

Step 9 **exit**

Example:

```
Device(config-ecfm)# exit
```

Returns to global configuration mode.

Step 10 **ethernet cfm global**

Example:

```
Device(config)# ethernet cfm global
```

Enables CFM processing globally on the device.

Step 11 **etheret cfm ieee**

Example:

```
Router(config)# ethernef cfm ieee
```

Enables CFM IEEE version of CFM.

This command is automatically issued when the ethernet cfm global command is issued.

Step 12 **ethernet cfm traceroute cache**

Example:

```
Device(config)# ethernet cfm traceroute cache
```

Enables caching of CFM data learned through traceroute messages.

Step 13 **ethernet cfm traceroute cache size** *entries*

Example:

```
Device(config)# ethernet cfm traceroute cache size 200
```

Sets the maximum size for the CFM traceroute cache table.

Step 14 **ethernet cfm traceroute cache hold-time** *minutes*

Example:

```
Device(config)# ethernet cfm traceroute cache hold-time 60
```

Sets the amount of time that CFM traceroute cache entries are retained.

Step 15 **snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**]

Example:

```
Device(config)# snmp-server enable traps ethernet cfm cc mep-up mep-down config loop
cross-connect
```

Enables SNMP trap generation for Ethernet CFM continuity check events.

Step 16 **snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**]

Example:

```
Device(config)# snmp-server enable traps ethernet cfm crosscheck mep-unknown mep-missing
service-up
```

Enables SNMP trap generation for Ethernet CFM continuity check events in relation to the cross-check operation between statically configured MEPS and those learned via CCMs.

Step 17 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 18 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet0/0/1
```

Specifies an interface and enters interface configuration mode.

Step 19 **service instance** *id* **ethernet** [*evc-name*]

Example:

```
Device(config-if)# service instance 333 ethernet evc1
```

Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.

Step 20 **encapsulation** *encapsulation-type*

Example:

```
Device(config-if-srv)# encapsulation dot1q 5
```

Sets the encapsulation method used by the interface.

Step 21 **bridge-domain** *bridge-id*

Example:

```
Device(config-if-srv)# bridge-domain 100
```

Binds a service instance to a bridge domain instance.

Step 22 **cfm mep domain** *domain-name* **mpid** *id*

Example:

```
Device(config-if-srv)# cfm mep domain L4 mpid 4001
```

Configures the MEP domain and the ID.

Step 23 **end**

Example:

```
Device(config-if-srv)# end
```

Returns to privileged EXEC mode.

Configuring Unicast MAC for CCM

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	**ethernet cfm domain** *domain-name* **level** *level-id* **Example:** `Device(config)# ethernet cfm domain Customer level 7`	Defines a CFM maintenance domain at a particular maintenance level and enters Ethernet CFM configuration mode.
Step 4	**service** *short-ma-name* **evc** *evc-name* **vlan** *vlanid* **direction down** **Example:** `Device(config-ecfm)# service s41 evc 41 vlan 41 direction down`	Configures a maintenance association within a maintenance domain and enters Ethernet connectivity fault management (CFM) service configuration mode. **Note** The **direction down** is used only for Down or Outward-facing MEPs. For Up MEPs or Inward-facing MEPs, do not specify **direction down**.
Step 5	**continuity-check** **Example:**	Enables the transmission of continuity check messages (CCMs).

	Command or Action	Purpose
	`Device(config-ecfm-srv)# continuity-check`	
Step 6	**continuity-check** [**interval** *cc-interval*] **Example:** `Device(config-ecfm-srv)# continuity-check interval 10s`	Configures the time period between CCMs transmission. The default interval is 10 seconds.
Step 7	**exit** **Example:** `Device(config-ecfm-srv)# exit`	Returns to Ethernet connectivity fault management configuration mode.
Step 8	**mep archive-hold-time** *minutes* **Example:** `Device(config-ecfm)# mep archive-hold-time 60`	Sets the amount of time that data from a missing MEP is kept in the continuity check database or that entries are held in the error database before they are purged.
Step 9	**exit** **Example:** `Device(config-ecfm)# exit`	Returns to global configuration mode.
Step 10	**ethernet cfm global** **Example:** `Device(config)# ethernet cfm global`	Enables CFM processing globally on the device.
Step 11	**etheret cfm ieee** **Example:** `Router(config)# ethernef cfm ieee`	Enables CFM IEEE version of CFM. This command is automatically issued when the ethernet cfm global command is issued.
Step 12	**ethernet cfm traceroute cache** **Example:** `Device(config)# ethernet cfm traceroute cache`	Enables caching of CFM data learned through traceroute messages.
Step 13	**ethernet cfm traceroute cache size** *entries* **Example:** `Device(config)# ethernet cfm traceroute cache size 200`	Sets the maximum size for the CFM traceroute cache table.
Step 14	**ethernet cfm traceroute cache hold-time** *minutes* **Example:** `Device(config)# ethernet cfm traceroute cache hold-time 60`	Sets the amount of time that CFM traceroute cache entries are retained.

	Command or Action	Purpose
Step 15	**snmp-server enable traps ethernet cfm cc** [**mep-up**] [**mep-down**] [**config**] [**loop**] [**cross-connect**] **Example:** `Device(config)# snmp-server enable traps ethernet cfm cc mep-up mep-down config loop cross-connect`	Enables SNMP trap generation for Ethernet CFM continuity check events.
Step 16	**snmp-server enable traps ethernet cfm crosscheck** [**mep-unknown** | **mep-missing** | **service-up**] **Example:** `Device(config)# snmp-server enable traps ethernet cfm crosscheck mep-unknown mep-missing service-up`	Enables SNMP trap generation for Ethernet CFM continuity check events in relation to the cross-check operation between statically configured MEPS and those learned via CCMs.
Step 17	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 18	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet0/0/1`	Specifies an interface and enters interface configuration mode.
Step 19	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Device(config-if)# service instance 333 ethernet evc1`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.
Step 20	**encapsulation** *encapsulation-type* **Example:** `Device(config-if-srv)# encapsulation dot1q 5`	Sets the encapsulation method used by the interface.
Step 21	**bridge-domain** *bridge-id* **Example:** `Device(config-if-srv)# bridge-domain 100`	Binds a service instance to a bridge domain instance.
Step 22	**cfm mep domain** *domain-name* **mpid** *id* **Example:** `Device(config-if-srv)#  cfm mep domain cust1 mpid 1 unicast 00f6.6321.6d95`	Configures the MEP in unicast mode.

	Command or Action	Purpose
Step 23	**end** **Example:** `Device(config-if-srv)# end`	Returns to privileged EXEC mode.

Verifying Unicast MAC for CCM

You can use the **show ethernet cfm maintenance-points local detailed** command to get detailed information on Unicast MAC for CCM on the Gigabit Ethernet interface.

```
Router(config)#show ethernet cfm maintenance-points local detail
Local MEPs:
----------
MPID: 1
DomainName: cust1
Domain ID: cust1
MA Name: s1
Level: 7
Direction: Up
EVC: evc1
Bridge Domain: 10
Service Instance: 10
Interface: Te0/0/22
CC Offload: No
CC-Status: Enabled
CC Loss Threshold: 3
MAC: 00a7.42d1.5ebf
CC Transmission Mode: Unicast
CC Unicast Triggered  Via: Static
CC Unicast Remote Mep Mac Address: 00f6.6321.6d95
LCK-Status: Enabled
LCK Period: 60000(ms)
LCK Expiry Threshold: 3.5
Level to transmit LCK: Default
Defect Condition: No Defect
presentRDI: FALSE
AIS-Status: Enabled
AIS Period: 60000(ms)
AIS Expiry Threshold: 3.5
Level to transmit AIS: Default
Suppress Alarm configuration: Enabled
Suppressing Alarms: No
Source: Static
```

You can use the **show ethernet cfm maintenance-points local** command to view information on Unicast MAC for CCM on the Gigabit Ethernet interface.

```
Router#show ethernet cfm maintenance-points local
Local MEPs:
--------------------------------------------------------------------------------
MPID Domain Name                                Lvl   MacAddress     Type  CC
Ofld Domain Id                                  Dir   Port           Id
     MA Name                                          SrvcInst       Source
     EVC name
     CCM Mode
--------------------------------------------------------------------------------
1    cust1                                      7     00a7.42d1.5ebf BD-V  Y
No   cust1                                      Up    Te0/0/22       10
     s1                                               10             Static
```

```
     evc1
     Unicast
3    sp1                                          5     00a7.42d1.5e82 BD-V  Y
Yes  sp1                                          Down  Gi0/0/2        10
     sa1                                                10             Static
     evc1
     Unicast

Total Local MEPs: 2

Local MIPs: None
```

You can use the **show ethernet cfm maintenance-points remote** command to view information on Unicast MAC for CCM on the Gigabit Ethernet interface.

```
Router(config)#show ethernet cfm maintenance-points remote
-------------------------------------------------------------------------------
MPID  Domain Name                                MacAddress          IfSt  PtSt
 Lvl  Domain ID                                  Ingress
 RDI  MA Name                                    Type Id             SrvcInst
      EVC Name                                                       Age
      Local MEP Info
-------------------------------------------------------------------------------
2     cust1                                      00f6.6321.6dce      Up    Up
 7    cust1                                      Gi0/0/2
 -    s1                                         BD-V 10             10
      evc1                                                           0s
      MPID: 1 Domain: cust1 MA: s1
4     sp1                                        00f6.6321.6d90      Up    Up
 5    sp1                                        Gi0/0/2
 -    sa1                                        BD-V 10             10
      evc1                                                           3s
      MPID: 3 Domain: sp1 MA: sa1

Total Remote MEPs: 2
```

CFM Use Cases

Example For Configuring CFM over Bridge Domain

```
ethernet cfm ieee
ethernet cfm global
ethernet cfm domain cust1 level 7
  service s1 evc 1 vlan 1
   continuity-check
   continuity-check interval 3.3ms

service instance 1 ethernet 1
  encapsulation dot1q 1
  bridge-domain 1
  cfm mep domain cust1 mpid 1
```

Example For Configuring CFM over Default Encapsulation

```
ethernet cfm domain oper2 level 7
service cust1 evc 1000 vlan 1500 direction down
  continuity-check
  continuity-check interval 3.3ms

service instance 1000 ethernet 1000
  encapsulation default
  bridge-domain 1500
```

```
cfm mep domain cust1 mpid 8191
cfm encapsulation dot1q 1500
```

Verification Commands for CFM

Use the following commands to verify CFM:

- **show ethernet cfm maintenance-points local**
- **show ethernet cfm maintenance-points remote**
- **show ethernet cfm statistics**
- **show ethernet cfm ccm-learning-database**
- **show ethernet cfm errors**

SNMP Traps

The support provided by the Cisco IOS XE software implementation of Ethernet CFM traps is Cisco proprietary information. MEPs generate two types of Simple Network Management Protocol (SNMP) traps, continuity check (CC) traps and cross-check traps.

CC Traps

- MEP up--Sent when a new MEP is discovered, the status of a remote port changes, or connectivity from a previously discovered MEP is restored after interruption.
- MEP down--Sent when a timeout or last gasp event occurs.
- Cross-connect--Sent when a service ID does not match the VLAN.
- Loop--Sent when a MEP receives its own CCMs.
- Configuration error--Sent when a MEP receives a continuity check with an overlapping MPID.

Cross-Check Traps

- Service up--Sent when all expected remote MEPs are up in time.
- MEP missing--Sent when an expected MEP is down.
- Unknown MEP--Sent when a CCM is received from an unexpected MEP.

Steps to Generate SNMP Traps for CFM

To generate SNMP traps, following commands need to be configured on the router.

```
ethernet cfm logging
logging snmp-trap 0 7
logging history debugging
```

Send Trap to SNMP Server

```
snmp-server enable traps ethernet cfm cc [mep-up] [mep-down] [config] [loop] [cross-connect]
snmp-server enable traps ethernet cfm crosscheck [mep-unknown] [mep-missing] [ service-up]
```

Note If syslog trap is enabled, by default trap is generated for messages of severity level emergency, alert, critical, error and warning (0-4). For other severity levels need to enable **logging snmp-trap 0 7** and **logging history debugging**

```
Router(config)#ethernet cfm logging
Router(config)#logging snmp-trap 0 7
Router(config)#logging history debugging
Router(config)#snmp-server enable traps ethernet cfm cc
Router(config)#snmp-server enable traps ethernet cfm crosscheck
```

Logs for MEP going DOWN

```
Console-logs:

Router(config)#
*Oct 26 21:32:06.663 IST: %E_CFM-3-REMOTE_MEP_DOWN: Remote MEP mpid 10 evc 2 vlan 2 MA name
 s2 in domain cust2 changed state to down with event code TimeOut.
*Oct 26 21:32:06.664 IST: %E_CFM-6-ENTER_AIS: local mep with mpid 20 level 2 BD/VLAN 2 dir
 D Interface Te0/3/1 enters AIS defect condition
*Oct 26 21:32:09.147 IST: %E_CFM-3-FAULT_ALARM: A fault has occurred in the network for the
 local MEP having mpid 20 evc 2 vlan 2 for service MA name s2 with the event code
DefRemoteCCM.
```

SNMP Server Side Logs

Received SNMPv2c Trap

```
Community: public
From: 7.32.22.154
sysUpTimeInstance = 04:00:54.27
snmpTrapOID.0 = clogMessageGenerated
clogHistFacility.76 = E_CFM
clogHistSeverity.76 = error(4)
clogHistMsgName.76 = REMOTE_MEP_DOWN
clogHistMsgText.76 = Remote MEP mpid 10 evc 2 vlan 2 MA name s2 in domain cust2 changed
state to down with event code TimeOut.
clogHistTimestamp.76 = 04:00:54.27
```

Received SNMPv2c Trap

```
Community: public
From: 7.32.22.154
sysUpTimeInstance = 04:00:54.27
snmpTrapOID.0 = clogMessageGenerated
clogHistFacility.77 = E_CFM
clogHistSeverity.77 = info(7)
clogHistMsgName.77 = ENTER_AIS
clogHistMsgText.77 = local mep with mpid 20 level 2 BD/VLAN 2 dir D Interface Te0/3/1 enters
 AIS defect condition
clogHistTimestamp.77 = 04:00:54.27
```

Received SNMPv2c Trap

```
Community: public
From: 7.32.22.154
sysUpTimeInstance = 04:00:56.75
snmpTrapOID.0 = dot1agCfmFaultAlarm
dot1agCfmMepHighestPrDefect.10.2.20 = defRemoteCCM(3)
```

Received SNMPv2c Trap

```
Community: public
From: 7.32.22.154
sysUpTimeInstance = 04:00:56.75
snmpTrapOID.0 = clogMessageGenerated
clogHistFacility.78 = E_CFM
clogHistSeverity.78 = error(4)
clogHistMsgName.78 = FAULT_ALARM
clogHistMsgText.78 = A fault has occurred in the network for the local MEP having mpid 20
evc 2 vlan 2 for service MA name s2 with the event code DefRemoteCCM.
clogHistTimestamp.78 = 04:00:56.75
```

Logs for MEP Coming Up

Console-logs

```
================================================
Router(config)#
*Oct 26 21:35:03.780 IST: %E_CFM-6-REMOTE_MEP_UP: Continuity Check message is received from
 a remote MEP with mpid 10 evc 2 vlan 2 MA name s2 domain cust2 interface status Up event
code Returning.
*Oct 26 21:35:03.781 IST: %E_CFM-6-EXIT_AIS: local mep with mpid 20 level 2 BD/VLAN 2 dir
D Interface Te0/3/1 exited AIS defect condition
```

SNMP Server Side Logs

Received SNMPv2c Trap

```
================================================
Community: public
From: 7.32.22.154
sysUpTimeInstance = 04:03:51.39
snmpTrapOID.0 = clogMessageGenerated
clogHistFacility.79 = E_CFM
clogHistSeverity.79 = info(7)
clogHistMsgName.79 = REMOTE_MEP_UP
clogHistMsgText.79 = Continuity Check message is received from a remote MEP with mpid 10
evc 2 vlan 2 MA name s2 domain cust2 interface status Up event code Returning.
clogHistTimestamp.79 = 04:03:51.38
```

Received SNMPv2c Trap

```
Community: public
From: 7.32.22.154
```

```
sysUpTimeInstance = 04:03:51.39
snmpTrapOID.0 = clogMessageGenerated
clogHistFacility.80 = E_CFM
clogHistSeverity.80 = info(7)
clogHistMsgName.80 = EXIT_AIS
clogHistMsgText.80 = local mep with mpid 20 level 2 BD/VLAN 2 dir D Interface Te0/3/1 exited
 AIS defect condition
clogHistTimestamp.80 = 04:03:51.38
```

Troubleshooting Tips

To verify and isolate a fault, start at the highest level maintenance domain and do the following:

- Check the device error status.
- When an error exists, perform a loopback test to confirm the error.
- Run a traceroute to the destination to isolate the fault.
- If the fault is identified, correct the fault.
- If the fault is not identified, go to the next lower maintenance domain and repeat these four steps at that maintenance domain level.
- Repeat the first four steps, as needed, to identify and correct the fault.

Troubleshooting CFM Features

Provides troubleshooting solutions for the CFM features.

Table 3: Troubleshooting Scenarios for CFM Features

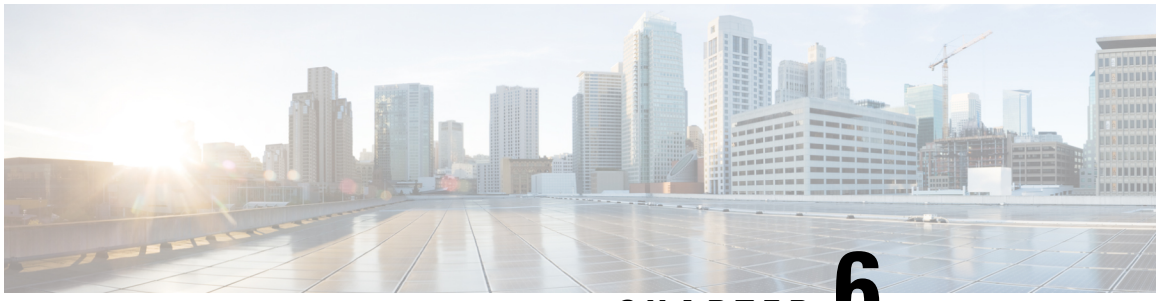
Problem	Solution
When you configure CFM, the message “Match registers are not available” is displayed.	For more information on match registers, see Ethernet Co Fault Management at http://www.cisco.com/en/US/docs/ios/12_2sr/12_2sra/feature/guide/sr CFM uses two match registers to identify the control pa and each VLAN spanning tree also uses a match register its control packet type. For both protocols to work on t system, each line card should support three match regis at least one supporting only a 44 bit MAC match.
CFM configuration errors	CFM configuration error occurs when when a MEP rec continuity check with an overlapping MPID. To verify of the error, use the command **show ethernet cfm erro configuration** or **show ethernet cfm errors**.

Problem	Solution
CFM ping and traceroute result is "not found"	Complete these steps: 1. Use **show run | i ethernet cfm** to view all CFM configurations. 2. Use **show ethernet cfm statistics** to view loca their CCM statistics 3. Use **trace ethernet cfm** command to start a CF
CFM connectivity is down and issues at the maintenance domain levels	Use the **ping ethernet {mac-address | mpid** *id* **| m** **domain** *domain-name* { **vlan v***lan-id* **| port | evc** *ev* **the traceroute ethernet {mac-address | mpid** *id* **}** *domain-name* { **vlan** *vlan-id* **| port | evc evc-name}** to verify ethernet CFM connectivity. Share the outp for further investigation. **Note** CFM multicast ping with packet size greater not supported.

Loop trap error	Use the **show ethernet cfm error** command to che Trap errors as shown here:
	`CE(config-if)#do sh ethernet cfm err`
	`Level Vlan MPID Remote MAC Reason Service ID`
	`5 711 550 1001.1001.1001 Loop Trap Er`
	`PE#sh ethernet cfm err`
	`Level Vlan MPID Remote MAC Reason Service ID`
	`5 711 550 1001.1001.1001 Loop Trap Er`

Module has insufficient match registers	Complete these steps: 1. Verify and confirm if a unsupported line card is the router. 2. If yes, perform an OIR of the unsupported line
CFM is deactivated	Complete these steps: 1. Check if all the line cards have free match reag 2. Check if CFM is activated on supervisor cards. supported on supervisor cards that has two matc this scenario, CFM is automatically disabled on and enabled on the remaining line cards.

Problem	Solution
ethernet cfm logging	In a scale scenario, you configure either the console log rate-limiting using **logging rate-limit** or using **logging** instead of using **logging console**. The suggested rate-limi 30 messages per second.



CHAPTER 6

Configuring Ethernet Local Management Interface at a Provider Edge

The advent of Ethernet as a metropolitan-area network (MAN) and WAN technology imposes a new set of Operation, Administration, and Management (OAM) requirements on Ethernet's traditional operations, which had centered on enterprise networks only. The expansion of Ethernet technology into the domain of service providers, where networks are substantially larger and more complex than enterprise networks and the user-base is wider, makes operational management of link uptime crucial. More importantly, the timeliness in isolating and responding to a failure becomes mandatory for normal day-to-day operations, and OAM translates directly to the competitiveness of the service provider.

The “Configuring Ethernet Local Management Interface at a Provide Edge” module provides general information about configuring an Ethernet Local Management Interface (LMI), an OAM protocol, on a provider edge (PE) device.

Prerequisites for Configuring Ethernet Local Management Interface at a Provider Edge

- Ethernet Operation, Administration, and Management (OAM) must be operational in the network.
- For Ethernet OAM to operate, the provider edge (PE) side of a connection must be running Ethernet Connectivity Fault Management (CFM) and Ethernet Local Management Interface (LMI).
- All VLANs used on a PE device to connect to a customer edge (CE) device must also be created on that CE device.
- To use nonstop forwarding (NSF) and In Service Software Upgrade (ISSU), stateful switchover (SSO) must be configured and working properly.

Restrictions for Configuring Ethernet Local Management Interface at a Provider Edge

- Ethernet Local Management Interface (LMI) is not supported on routed ports, EtherChannel port channels, ports that belong to an EtherChannel, private VLAN ports, IEEE 802.1Q tunnel ports, Ethernet over Multiprotocol Label Switching (MPLS) ports, or Ethernet Flow Points (EFPs) on trunk ports.
- Ethernet LMI cannot be configured on VLAN interfaces.
- The high availability (HA) features NSF/SSO—E-LMI Support and ISSU--E-LMI Support are not supported on a customer edge (CE) device.

Information About Configuring Ethernet Local Management Interface at a Provider Edge

Ethernet Virtual Circuits Overview

An Ethernet virtual circuit (EVC) as defined by the Metro Ethernet Forum is a port level point-to-point or multipoint-to-multipoint Layer 2 circuit. EVC status can be used by a customer edge (CE) device to find an alternative path in to the service provider network or in some cases to fall back to a backup path over Ethernet or another alternative service such as ATM.

Ethernet LMI Overview

Ethernet Local Management Interface (LMI) is an Ethernet Operation, Administration, and Management (OAM) protocol between a customer edge (CE) device and a provider edge (PE) device. Ethernet LMI provides CE devices with the status of Ethernet virtual circuits (EVCs) for large Ethernet metropolitan-area networks (MANs) and WANs and provides information that enables CE devices to autoconfigure. Specifically, Ethernet LMI runs on the PE-CE User-Network Interface (UNI) link and notifies a CE device of the operating state of an EVC and the time when an EVC is added or deleted. Ethernet LMI also communicates the attributes of an EVC.

Ethernet LMI interoperates with Ethernet Connectivity Fault Management (CFM), an OAM protocol that runs within the provider network to collect OAM status. Ethernet CFM runs at the provider maintenance level (user provider edge [UPE] to UPE at the UNI). Ethernet LMI relies on the OAM Ethernet Infrastructure (EI) to interwork with CFM to learn the end-to-end status of EVCs across CFM domains.

Ethernet LMI is disabled globally by default. When Ethernet LMI is enabled globally, all interfaces are automatically enabled. Ethernet LMI can also be enabled or disabled at the interface to override the global configuration. The last Ethernet LMI command issued is the command that has precedence. No EVCs, Ethernet service instances, or UNIs are defined, and the UNI bundling service is bundling with multiplexing.

Ethernet CFM Overview

Ethernet Connectivity Fault Management (CFM) is an end-to-end per-service-instance (per VLAN) Ethernet layer Operation, Administration, and Management (OAM) protocol that includes proactive connectivity monitoring, fault verification, and fault isolation. End-to-end CFM can be from provider edge (PE) device to PE device or from customer edge (CE) device to CE device. For more information about Ethernet CFM, see "Configuring Ethernet Connectivity Fault Management in a Service Provider Network" in the *Carrier Ethernet Configuration Guide*.

OAM Manager Overview

The OAM manager is an infrastructure element that streamlines interaction between Operation, Administration, and Management (OAM) protocols. The OAM manager requires two interworking OAM protocols, Ethernet Connectivity Fault Management (CFM) and Ethernet Local Management Interface (LMI). No interactions are required between Ethernet LMI and the OAM manager on the customer edge (CE) side. On the User Provider-Edge (UPE) side, the OAM manager defines an abstraction layer that relays data collected from Ethernet CFM to the Ethernet LMI device.

Ethernet LMI and the OAM manager interaction is unidirectional, from the OAM manager to Ethernet LMI on the UPE side of the device. An information exchange results from an Ethernet LMI request or is triggered by the OAM manager when it receives notification from the OAM protocol that the number of UNIs has changed. A change in the number of UNIs may cause a change in Ethernet virtual circuit (EVC) status.

The OAM manager calculates EVC status given the number of active user network interfaces (UNIs) and the total number of associated UNIs. You must configure CFM to notify the OAM manager of all changes to the number of active UNIs or to the remote UNI ID for a given service provider VLAN (S-VLAN) domain.

The information exchanged is as follows:

- EVC name and availability status (active, inactive, partially active, or not defined)
- Remote UNI name and status (up, disconnected, administratively down, excessive frame check sequence [FCS] failures, or not reachable)
- Remote UNI counts (the total number of expected UNIs and the number of active UNIs)

Benefits of Ethernet LMI at a Provider Edge

- Communication of end-to-end status of the Ethernet virtual circuit (EVC) to the customer edge (CE) device
- Communication of EVC and user network interface (UNI) attributes to a CE device
- Competitive advantage for service providers

HA Features Supported by Ethernet LMI

In access and service provider networks using Ethernet technology, high availability (HA) is a requirement, especially on Ethernet operations, administration, and management (OAM) components that manage Ethernet virtual circuit (EVC) connectivity. End-to-end connectivity status information is critical and must be maintained on a hot standby Route Processor (RP) (a standby RP that has the same software image as the active RP and

supports synchronization of line card, protocol, and application state information between RPs for supported features and protocols).

End-to-end connectivity status is maintained on the customer edge (CE), provider edge (PE), and access aggregation PE (uPE) network nodes based on information received by protocols such as Ethernet Local Management Interface (LMI), Connectivity Fault Managment (CFM), and 802.3ah. This status information is used to either stop traffic or switch to backup paths when an EVC is down.

Metro Ethernet clients (E-LMI, CFM, 802.3ah) maintain configuration data and dynamic data, which is learned through protocols. Every transaction involves either accessing or updating data in the various databases. If the database is synchronized across active and standby modules, the modules are transparent to clients.

The Cisco infrastructure provides component application programming interfaces (APIs) that are helpful in maintaining a hot standby RP. Metro Ethernet HA clients (E-LMI, HA/ISSU, CFM HA/ISSU, 802.3ah HA/ISSU) interact with these components, update the database, and trigger necessary events to other components.

Benefits of Ethernet LMI HA

- Elimination of network downtime for Cisco software image upgrades, resulting in higher availability.
- Elimination of resource scheduling challenges associated with planned outages and late night maintenance windows
- Accelerated deployment of new services and applications and faster implementation of new features, hardware, and fixes due to the elimination of network downtime during upgrades
- Reduced operating costs due to outages while the system delivers higher service levels due to the elimination of network downtime during upgrades

NSF SSO Support in Ethernet LMI

The redundancy configurations stateful switchover (SSO) and nonstop forwarding (NSF) are supported in Ethernet Local Management Interface (LMI) and are automatically enabled. A switchover from an active to a standby Route Processor (RP) or a standby Route Switch Processor (RSP) occurs when the active RP or RSP fails, is removed from the networking device, or is manually taken down for maintenance. The primary function of Cisco NSF is to continue forwarding IP packets following an RP or RSP switchover. NSF also interoperates with the SSO feature to minimize network downtime following a switchover.

For detailed information about the SSO and NSF features, see the *High Availability Configuration Guide*.

ISSU Support in Ethernet LMI

In Service Software Upgrade (ISSU) allows you to perform a Cisco software upgrade or downgrade without disrupting packet flow. Ethernet Local Management Interface (LMI) performs updates of the parameters within the Ethernet LMI database to the standby route processor (RP) or standby route switch processor (RSP). This checkpoint data requires ISSU capability to transform messages from one release to another. All the components that perform active processor to standby processor updates using messages require ISSU support. ISSU is automatically enabled in Ethernet LMI.

ISSU lowers the impact that planned maintenance activities have on network availability by allowing software changes while the system is in service. For detailed information about ISSU, see the *High Availability Configuration Guide*.

How to Configure Ethernet Local Management Interface at a Provider Edge

Configuring Ethernet LMI Interaction with CFM

For Ethernet Local Management Interface (LMI) to function with Connectivity Fault Management (CFM), you must configure Ethernet virtual circuits (EVCs), Ethernet service instances including untagged Ethernet flow points (EFPs), and Ethernet LMI customer VLAN mapping. Most of the configuration occurs on the provider edge (PE) device on the interfaces connected to the customer edge (CE) device. On the CE device, you need only enable Ethernet LMI on the connecting interface. Also, you must configure operations, administration, and management (OAM) parameters; for example, EVC definitions on PE devices on both sides of a metro network.

CFM and OAM interworking requires an inward facing Maintenance Entity Group End Point (MEP).

Configuring the OAM Manager

Note If you configure, change, or remove a user network interface (UNI) service type, Ethernet virtual circuit (EVC), Ethernet service instance, or customer edge (CE)-VLAN configuration, all configurations are checked to ensure that the configurations match (UNI service type with EVC or Ethernet service instance and CE-VLAN configuration). The configuration is rejected if the configurations do not match.

Perform this task to configure the OAM manager on a provider edge (PE) device.

Procedure

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 **ethernet cfm domain** *domain-name* **level** *level-id*

Example:

```
Device(config)# ethernet cfm domain cstmr1 level 3
```

Defines a Connectivity Fault Management (CFM) domain, sets the domain leve,l and enters Ethernet CFM configuration mode.

Step 4 **service** *csi-id* **evc** *evc-name* **vlan** *vlan-id*

Example:

```
Device(config-ecfm)# service csi2 evc evc_1 vlan 10
```

Defines a universally unique customer service instance (CSI) and VLAN ID within the maintenance domain, and enters Ethernet CFM service configuration mode.

Step 5 **continuity-check**

Example:

```
Device(config-ecfm-srv)# continuity-check
```

Enables the transmission of continuity check messages (CCMs).

Step 6 **continuity-check interval** *time*

Example:

```
Device(config-ecfm-srv)# continuity-check interval 1s/10s/1m/10m
```

Enables the transmission of continuity check messages (CCMs) at specific intervals.

Step 7 **exit**

Example:

```
Device(config-ecfm-srv)# exit
```

Returns to Ethernet CFM configuration mode.

Step 8 **exit**

Example:

```
Device(config-ecfm)# exit
```

Returns to global configuration mode.

Step 9 **ethernet evc** *evc-id*

Example:

```
Device(config)# ethernet evc 50
```

Defines an EVC and enters EVC configuration mode.

Step 10 **oam protocol** {**cfm domain** *domain-name* | **ldp**}

Example:

```
Device(config-evc)# oam protocol cfm domain cstmr1
```

Configures the Ethernet virtual circuit (EVC) operations, administration, and management (OAM) protocol as CFM for the CFM domain maintenance level as configured in Steps 3 and 4.

Note If the CFM domain does not exist, this command is rejected, and an error message is displayed.

Step 11 **uni count** *value*

Example:

```
Device(config-evc)# uni count 3
```

(Optional) Sets the User Network Interface (UNI) count for the EVC.

- If this command is not issued, the service defaults to a point-to-point service. If a value of 2 is entered, point-to-multipoint service becomes an option. If a value of 3 or greater is entered, the service is point-to-multipoint.

Note If you enter a number greater than the number of endpoints, the UNI status is partially active even if all endpoints are up. If you enter a UNI count less than the number of endpoints, status might be active, even if all endpoints are not up.

Step 12 **exit**

Example:

```
Device(config-evc)# exit
```

Returns to global configuration mode.

Step 13 Repeat Steps 3 through 12 to define other CFM domains that you want OAM manager to monitor.

Example:

—

Step 14 **interface** *type number*

Example:

```
Device(config)# interface gigabitethernet 0/0/2
```

Specifies a physical interface connected to the CE device and enters interface configuration mode.

Step 15 **service instance** *id* **ethernet** [*evc-id*]

Example:

```
Device(config-if)# service instance 400 ethernet 50
```

Configures an Ethernet service instance on the interface and enters Ethernet service configuration mode.

- The Ethernet service instance identifier is a per-interface service identifier and does not map to a VLAN.

Step 16 **ethernet lmi ce-vlan map** {*vlan-id* [**untagged**] | **any** | **default** | **untagged**}

Example:

```
Device(config-if-srv)# ethernet lmi ce-vlan map 30
```

Configures an Ethernet LMI customer VLAN-to-EVC map for a particular UNI.

Note To specify both VLAN IDs and untagged VLANs in the map, specify the VLAN IDs first and then specify the **untagged** keyword as follows: **ethernet lmi ce-vlan map 100,200,300,untagged**. Also, if the **untagged** keyword is not specified in the map configuration, the main interface line protocol on the Customer Edge (CE) device will be down.

Step 17 **service instance** *service-instance-id* **ethernet**

Example:

```
Device(config-if)# service instance 22 ethernet
```

Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.

Step 18 **encapsulation untagged**

Example:

```
Device(config-if-srv)# encapsulation untagged
```

Defines the matching criteria to map untagged ingress Ethernet frames on an interface to the appropriate service instance.

Step 19 **l2protocol peer**

Example:

```
Device(config-if-srv)# l2protocol peer
```

Configures transparent Layer 2 protocol peering on the interface.

Step 20 **bridge-domain** *bridge-domain-number*

Example:

```
Device(config-if-srv)# bridge-domain 1
```

Binds a service instance to a bridge domain instance.

Step 21 **exit**

Example:

```
Device(config-if)# exit
```

Returns to interface configuration mode.

Step 22 **ethernet uni** [**bundle** [**all-to-one**] | **id** *uni-id* | **multiplex**]

Example:

```
Device(config-if)# ethernet uni bundle
```

Sets UNI bundling attributes.

Step 23 **end**

Example:

```
Device(config-if)# end
```

Returns to privileged EXEC mode.

Enabling Ethernet LMI

The order in which the global and interface configuration commands are issued determines the configuration. The last command that is issued has precedence.

Perform this task to enable Ethernet Local Management Interface (LMI) on a device or on an interface.

Procedure

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 ethernet 1/3
```

Defines an interface to configure as an Ethernet LMI interface and enters interface configuration mode.

Step 4 **ethernet lmi interface**

Example:

```
Device(config-if)# ethernet lmi interface
```

Configures Ethernet LMI on the interface.

- When Ethernet LMI is enabled globally, it is enabled on all interfaces unless you disable it on specific interfaces. If Ethernet LMI is disabled globally, you can use this command to enable it on specified interfaces.

Step 5 **ethernet lmi** {**n393** *value* | **t392** *value*}

Example:

```
Device(config-if)# ethernet lmi n393 10
```

Configures Ethernet LMI parameters for the UNI.

Step 6 **end**

Example:

```
Device(config-if)# end
```

Returns to privileged EXEC mode.

Displaying Ethernet LMI and OAM Manager Information

Perform this task to display Ethernet Local Management Interface (LMI) or Operation, Administration, and Management (OAM) manager information. After step 1, all the steps are optional and can be performed in any order.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show ethernet lmi** {{**evc** [**detail** *evc-id* [**interface** *type number*] | **map interface** *type number*]} | {**parameters** | **statistics**} **interface** *type number* | **uni map** [**interface** *type number*]}

Example:

```
Device# show ethernet lmi evc
```

Displays information that was sent to the customer edge (CE).

Step 3 **show ethernet service evc** [**detail** | **id** *evc-id* [**detail**] | **interface** *type number* [**detail**]]

Example:

```
Device# show ethernet service evc
```

Displays information about all Ethernet virtual circuits (EVCs) or about a specified EVC.

Step 4 **show ethernet service instance** [**detail** | **id** *id* | **interface** *type number* | **policy-map** | **stats**]

Example:

```
Device# show ethernet service instance detail
```

Displays information about customer service instances.

Step 5 **show ethernet service interface** [*type number*] [**detail**]

Example:

```
Device# show ethernet service interface ethernet 1/3 detail
```

Displays interface-only information about Ethernet customer service instances for all interfaces or for a specified interface.

Examples

The following example shows sample output from the **show ethernet lmi** command using the **evc** keyword:

```
Device# show ethernet lmi evc

St  EVC Id                                                          Port
--- ------------------------------------------------------------ --------------
A  EVC_MP2MP_101                                                  Gi0/1
A  EVC_P2P_110                                                    Gi0/1
```

The following example is sample output from the **show ethernet service evc** command:

```
Device# show ethernet service evc

Identifier                    Type  Act-UNI-cnt Status
50                            MP-MP     0       NotDefined
```

The following is sample output from the **show ethernet service interface** command using the **detail** keyword:

```
Device# show ethernet service interface gigabitethernet 0/0/2 detail

Interface: Gigabitethernet 0/0/2
ID: uni2
CE-VLANS: 30
EVC Map Type: Bundling
Associated EVCs:
    EVC-ID                          CE-VLAN
    50                              30
Associated Service Instances:
    Service-Instance-ID CE-VLAN
    400                 30
```

The following is sample output from the **show ethernet service instance** command using the **detail** keyword:

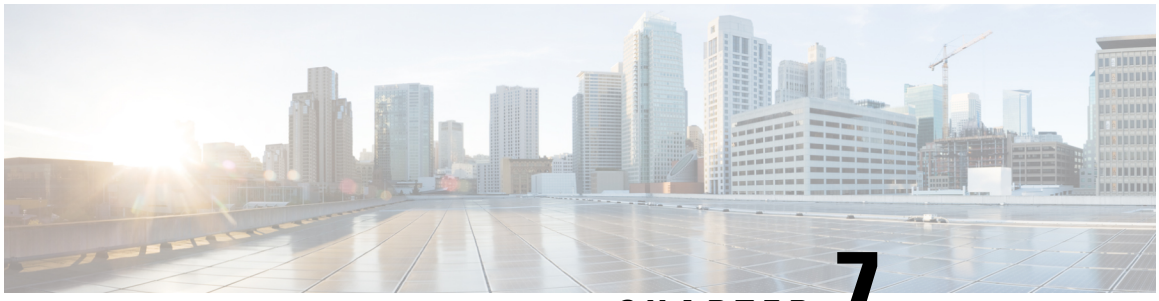
```
Device# show ethernet service instance detail

Service Instance ID: 400
Associated Interface: GigabitEthernet0/0/2
Associated EVC: 50
CE-Vlans: 30
State: AdminDown
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0
```

Configuration Examples for Ethernet Local Management Interface at a Provider Edge

Example: Ethernet OAM Manager on a PE Device Configuration

This example shows a sample configuration of Operation, Administration, and Management (OAM) manager, Connectivity Fault Management (CFM), and Ethernet Local Management Interface (LMI) on a provider edge (PE) device. In this example, a bridge domain is specified.

```
Device> enable
Device# configure terminal
Device(config)# ethernet cfm global
Device(config)# ethernet cfm domain provider level 4
Device(config-ecfm)# service customer_1 evc test1 vlan 10
Device(config-ecfm-srv)# continuity-check
Device(config-ecfm-srv)# continuity-check interval 1s/10s/1m/10m
Device(config-ecfm-srv)# exit
Device(config-ecfm)# exit
Device(config)# ethernet evc test1
Device(config-evc)# uni count 3
Device(config-evc)# oam protocol cfm domain provider
Device(config-evc)# exit
Device(config)#  interface gigabitEthernet 0/0/2
Device(config-if)# ethernet lmi interface
Device(config-if)# ethernet uni id CISCO
Device(config-if)# service instance 1 ethernet
Device(config-if-srv)# encapsulation untagged
Device(config-if-srv)# l2protocol peer
Device(config-if-srv)# bridge-domain 1
Device(config-if-srv)# exit
Device(config-if)# service instance 2 ethernet1
Device(config-if-srv)# ethernet lmi ce-vlan map 101
Device(config-if-srv)# encapsulation dot1q 2
Device(config-if-srv)# bridge-domain 2
Device(config-if-srv)# cfm mep domain provider mpid 10
Device(config-if-srv-ecfm-mep)# end
```



CHAPTER 7

ITU-T Y.1731 Performance Monitoring in a Service Provider Network

ITU-T Y.1731 performance monitoring provides standard-based Ethernet performance monitoring that encompasses the measurement of Ethernet frame delay, frame-delay variation, and throughput as outlined in the ITU-T Y.1731 specification and interpreted by the Metro Ethernet Forum (MEF). Service providers offer service level agreements (SLAs) that describe the level of performance customers can expect for services. This document describes the Ethernet performance management aspect of SLAs.

Prerequisites for ITU-T Y.1731 Performance Monitoring in a Service Provider Network

- For Y.1731 performance monitoring to work, connectivity fault management (CFM) sessions should be up and running.
- Continuity check messages (CCM) database should be populated.

Restrictions for ITU-T Y.1731 Performance Monitoring in a Service Provider Network

- Y.1731 performance monitoring supports only synthetic loss measurement (SLM) and two-way delay measurement (DMM).
- Y.1731 performance monitoring does not support one-way frame-delay measurement (1DM), loss measurement management (LMM) and clocksync.
- Y.1731 performance monitoring sessions cannot be initiated from port maintenance end points (port-MEP) and trunk ethernet flow points (trunk EFP).

- It is not recommended to change the default frame interval and frame sizes in Y.1731 performance monitoring. Default frame interval of only 1000 ms is supported.
- Maximum number of Y.1731 performance monitoring sessions supported is 100.
- While Y.1731 performance monitoring sessions are running, it is not recommneded to perform dynamic encapsulation modification.
- Y.1731 performance monitoring over CFM encapsulation default/untagged should have cos bit set to zero.
- Y1731 performance monitoring is not supported if the core dot1ad nni interface is configured as trunk.

Information About ITU-T Y.1731 Performance Monitoring in a Service Provider Network

Frame Delay and Frame-Delay Variation

The Frame Delay parameter can be used for on-demand OAM measurements of frame delay and frame-delay variation. When a maintenance end point (MEP) is enabled to generate frames with frame-delay measurement (ETH-DM) information, it periodically sends frames with ETH-DM information to its peer MEP in the same maintenance entity. Peer MEPs perform frame-delay and frame-delay variation measurements through this periodic exchange during the diagnostic interval.

An MEP requires the following specific configuration information to support ETH-DM:

- MEG level—MEG level at which the MEP exists
- Priority
- Transmission rate
- Total interval of ETH-DM

A MEP transmits frames with ETH-DM information using the TxTimeStampf information element. TxTimeStampf is the time stamp for when the ETH-DM frame was sent. A receiving MEP can compare the TxTimeStampf value with the RxTimef value, which is the time the ETH-DM frame was received, and calculate one-way delay using the formula *frame delay = RxTimef – TxTimeStampf*.

One-way frame-delay measurement (1DM) requires that clocks at both the transmitting MEP and the receiving MEPs are synchronized. Measuring frame-delay variation does not require clock synchronization and the variation can be measured using 1DM or a frame-delay measurement message (DMM) and a frame-delay measurement reply (DMR) frame combination.

If it is not practical to have clocks synchronized, only two-way frame-delay measurements can be made. In this case, the MEP transmits a frame containing ETH-DM request information and the TxTimeStampf element, and the receiving MEP responds with a frame containing ETH-DM reply information and the TxTimeStampf value copied from the ETH-DM request information.

Two-way frame delay is calculated as *(RxTimeb–TxTimeStampf)–(TxTimeStampb–RxTimeStampf)*, where RxTimeb is the time that the frame with ETH-DM reply information was received. Two-way frame delay and variation can be measured using only DMM and DMR frames.

To allow more precise two-way frame-delay measurement, the MEP replying to a frame with ETH-DM request information can also include two additional time stamps in the ETH-DM reply information:

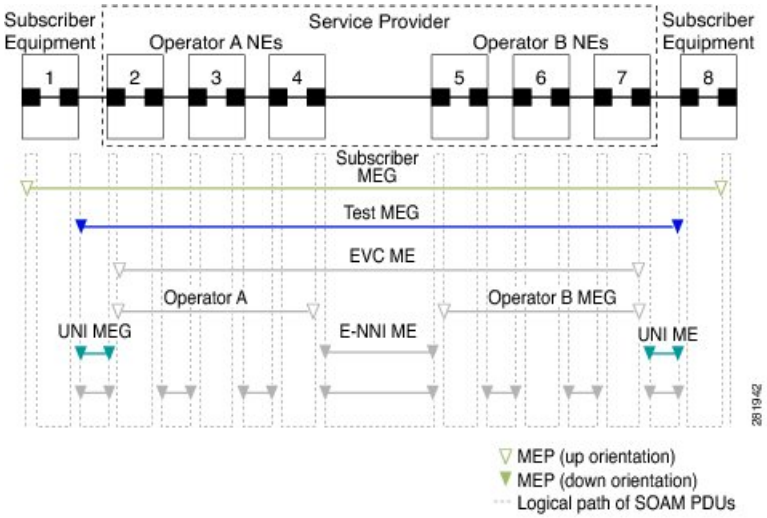
- RxTimeStampf—Time stamp of the time at which the frame with ETH-DM request information was received.
- TxTimeStampb—Time stamp of the time at which the transmitting frame with ETH-DM reply information was sent.
- The timestamping happens at the hardware level for DMM operations.

Note The frame-loss, frame-delay, and frame-delay variation measurement processes are terminated when faults related to continuity and availability occur or when known network topology changes occur.

An MIP is transparent to the frames with ETH-DM information; therefore, an MIP does not require information to support the ETH-DM function.

The figure below shows a functional overview of a typical network in which Y.1731 performance monitoring is used.

Figure 4: Y.1731 Performance Monitoring



Frame Loss Ratio

Ethernet Frame Loss Ratio (ETH-LM: FLR), also known as frame loss, measures the availability of synthetic frames in the network. Availability is defined in terms of the ratio of frames lost to frames sent, or Frame Loss Ratio (FLR).

Ethernet Synthetic Loss Measurement (ETH-SLM) is used to collect counter values applicable for ingress and egress synthetic frames where the counters maintain a count of transmitted and received synthetic frames between a pair of MEPs.

ETH-SLM transmits synthetic frames with ETH-SLM information to a peer MEP and similarly receives synthetic frames with ETH-SLM information from the peer MEP. Each MEP performs frame loss measurements,

which contribute to unavailable time. A near-end frame loss refers to frame loss associated with ingress data frames. A far-end frame loss refers to frame loss associated with egress data frames. Both near-end and far-end frame loss measurements contribute to near-end severely errored seconds and far-end severely errored seconds, which together contribute to unavailable time. ETH-SLM is measured using SLM and SLR frames.

There are the two methods of frame loss measurement, defined by the ITU-T Y.1731 standard ETH-LM and ETH-SLM. However, the Cisco NCS 520 router supports only single-ended ETH-SLM.

Single-ended ETH-SLM

Each MEP transmits frames with the ETH-SLM request information to its peer MEP and receives frames with ETH-SLR reply information from its peer MEP to carry out synthetic loss measurements.

Benefits of ITU-T Y.1731 Performance Monitoring

Combined with IEEE-compliant connectivity fault management (CFM), Y.1731 performance monitoring provides a comprehensive fault management and performance monitoring solution for service providers. This comprehensive solution in turn lessens service providers' operating expenses, improves their service-level agreements (SLAs), and simplifies their operations.

How to Configure ITU-T Y.1731 Performance Monitoring in a Service Provider Network

Configuring Ethernet Two-Way Delay Measurement

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	**ip sla** *operation-number* **Example:** `Device(config-term)# ip sla 10`	Begins configuring an IP SLAs operation and enters IP SLA configuration mode.

	Command or Action	Purpose
Step 4	**ethernet y1731 delay** {**DMM** | **DMMv1**} [**burst**] **domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} {**mpid** *target-mp-id* | **mac-address** *target-address*} **cos** *cos* {**source** {**mpid** *source-mp-id* | **mac-address** *source-address*}} **Example:** `Device(config-ip-sla)# ethernet y1731 delay dmm domain xxx evc yyy mpid 101 cos 4 source mpid 100`	Configures a two-way delay measurement and enters IP SLA Y.1731 delay configuration mode. • **DMM**—Configures two-way delay measurement. • **DMMv1**—Configures two-way delay measurement version 1. • **EVC**—Specifies the ethernet virtual circuit name. • **domain** *domain-name*—Specifies the name of the Ethernet Connectivity Fault Management (CFM) maintenance domain. • **vlan** *vlan-id*—Specifies the VLAN identification number. The range is from 1 to 4094. • **mpid** *target-mp-id*—Specifies the maintenance endpoint identification numbers of the MEP at the destination. The range is from 1 to 8191. • **mac-address** *target-address*—Specifies the MAC address of the MEP at the destination. • **cos** *cos*—Specifies, for this MEP, the class of service (CoS) that will be sent in the Ethernet message. The range is from 0 to 7. • **source**—Specifies the source MP ID or MAC address. • **mpid** *source-mp-id*—Specifies the maintenance endpoint identification numbers of the MEP being configured. The range is from 1 to 8191. • **mac-address** *source-address*—Specifies the MAC address of the MEP being configured.
Step 5	**aggregate interval** *seconds* **Example:** `Device(config-sla-y1731-delay)#`	(Optional) Configures the length of time during which the performance measurements are conducted and the results stored.

	Command or Action	Purpose
	`aggregate interval 900`	
Step 6	**distribution** {**delay** | **delay-variation**} **two-way** *number-of-bins boundary*[**,...,***boundary*] **Example:** `Device(config-sla-y1731-delay)# distribution delay-variation two-way 5 5000, 10000,15000,20000,-1`	(Optional) Specifies measurement type and configures bins for statistics distributions kept.
Step 7	**frame interval** *milliseconds* **Example:** `Device(config-sla-y1731-delay)# frame interval 1000`	(Optional) Sets the gap between successive frames. • *milliseconds*—Specifies the length of time in milliseconds (ms) between successive synthetic frames. The default is 1000
Step 8	**frame offset** *offset-value* **Example:** `Device(config-sla-y1731-delay)# frame offset 1`	(Optional) Sets value for calculating delay variation values.
Step 9	**frame size** *bytes* **Example:** `Device(config-sla-y1731-delay)# frame size 64`	(Optional) Configures padding size for frames. • *bytes*—Specifies the padding size, in four-octet increments, for the synthetic frames. The default is 64.
Step 10	**history interval** *intervals-stored* **Example:** `Device(config-sla-y1731-delay)# history interval 2`	(Optional) Sets the number of statistics distributions kept during the lifetime of an IP SLAs Ethernet operation. • *intervals-stored*—Specifies the number of statistics distributions. The range is from 1 to 10. The default is 2.
Step 11	**max-delay** *milliseconds* **Example:** `Device(config-sla-y1731-delay)# max-delay 5000`	(Optional) Sets the amount of time an MEP waits for a frame.
Step 12	**end** **Example:**	Exits to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-sla-y1731-delay)# end`	

What to do next

Once the DMM is configured, you have to schedule an IP SLA operation.

Configuring an SLM

To configure an SLM, execute the following commands:

Procedure

Step 1 **enable**

Example:

```
Router > enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal** *operation number*

—Identifies the IP SLAs' operation you want to configure.

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip sla** *operation number*

Example:

```
Router(config)# ip sla 11
```

Configures an IP SLA operation and enters IP SLA configuration mode.

- *operation-number*—Identifies the IP SLAs' operation you want to configure.

Step 4 **ethernet y1731 loss SLM domain** *domain-name* {**evc** *evc-id* | **vlan** *vlan-id*} {**mpid** *target-mp-id* | **mac-address**-*target -address*}**cos** *cos*{**source**{**mpid** *source-mp-id* | **mac-address** *source-address*}}

Example:

```
Router(config-ip-sla)# ethernet y1731 loss SLM domain xxx evc yyy mpid 101 cos 4 source
mpid 100
```

Configures a single-ended synthetic loss measurement and enters IP SLA Y.1731 loss configuration mode.

- **EVC**—Specifies the ethernet virtual circuit name.

- **SLM**—Specifies that the frames sent are Synthetic Loss Measurement (SLM) frames.
- **domain** *domain-name*—Specifies the name of the Ethernet Connectivity Fault Management (CFM) maintenance domain.
- **vlan** *vlan-id*—Specifies the VLAN identification number. The range is from 1 to 4094.
- **mpid** *target-mp-id*—Specifies the maintenance endpoint identification numbers of the MEP at the destination. The range is from 1 to 8191.
- **mac-address** *target-address*—Specifies the MAC address of the MEP at the destination.
- **cos** *cos*—Specifies, for this MEP, the class of service (CoS) that will be sent in the Ethernet message. The range is from 0 to 7.
- **source**—Specifies the source MP ID or MAC address.
- **mpid** *source-mp-id*—Specifies the maintenance endpoint identification numbers of the MEP being configured. The range is from 1 to 8191.
- **mac-address** *source-address*—Specifies the MAC address of the MEP being configured.

Step 5 **aggregate interval** *seconds*

Example:

```
Router(config-sla-y1731-loss)# aggregate interval 900
```

(Optional) Configures the length of time during which the performance measurements are conducted and the results stored.

- *seconds*—Specifies the length of time in seconds. The range is from 1 to 65535. The default is 900.

Step 6 **availability algorithm** { **sliding-window** | **static-window 1** } **symmetric**

Example:

```
Router(config-sla-y1731-loss)# availability algorithm static-window
```

(Optional) Specifies availability algorithm used.

- **sliding-window**—Specifies a sliding-window control algorithm.
- **static-window**—Specifies static-window control algorithm.

Step 7 **frame consecutive** *value*

Example:

```
Router(config-sla-y1731-loss)# frame consecutive 10.
```

(Optional) Specifies number of consecutive measurements to be used to determine availability or unavailability status.

- *value*—Specifies the number of consecutive measurements. The range is from 1 to 10. The default is 10.

Step 8 **frame interval** *milliseconds*

Example:

```
Router(config-sla-y1731-loss)# frame interval 1000
```

(Optional) Sets the gap between successive frames.

- *milliseconds*—Specifies the length of time in milliseconds (ms) between successive synthetic frames. The default is 1000

Step 9 **frame size** *bytes*

Example:

```
Router(config-sla-y1731-loss)# frame size 64
```

(Optional) Configures padding size for frames.

- *bytes*—Specifies the padding size, in four-octet increments, for the synthetic frames. The default is 64.

Step 10 **history interval** *intervals-stored*

Example:

```
Router(config-sla-y1731-loss)# history interval 2
```

(Optional) Sets the number of statistics distributions kept during the lifetime of an IP SLAs Ethernet operation.

- *intervals-stored*—Specifies the number of statistics distributions. The range is from 1 to 10. The default is 2.

Step 11 **exit**

Example:

```
Router(config-sla-y1731-loss)# exit
```

Exits IP SLA Y.1731 loss configuration mode and enters IP SLA configuration mode.

Step 12 **ip sla reaction-configuration** *operation-number* [**react** {**unavailableDS** |**unavailableSD** | **loss-ratioDS** | **loss-ratioSD**}] [**threshold-type** {**average** [*number -of-measurements*] | **consecutive** [*occurences*] | **immediate**}] [**threshold-value** *upper -threshold lower-threshold*]

Example:

```
Router(config)# ip sla reaction-configuration 11 react unavailableDS
```

(Optional) Configures proactive threshold monitoring for frame loss measurements.

- *operation-number*—Identifies the IP SLAs operation for which reactions are to be configured.
- **react**—(Optional) Specifies the element to be monitored for threshold violations.
- **unavailableDS**—Specifies that a reaction should occur if the percentage of destination-to-source Frame Loss Ratio (FLR) violates the upper threshold or lower threshold.
- **unavailableSD**—Specifies that a reaction should occur if the percentage of source-to-destination FLR violates the upper threshold or lower threshold.
- **loss-ratioDS**—Specifies that a reaction should occur if the one-way destination-to-source loss-ratio violates the upper threshold or lower threshold.

- **loss-ratioSD**—Specifies that a reaction should occur if the one way source-to-destination loss-ratio violates the upper threshold or lower threshold.
- **threshold-type average**[*number-of-measurements*]—(Optional) When the average of a specified number of measurements for the monitored element exceeds the upper threshold or when the average of a specified number of measurements for the monitored element drops below the lower threshold, perform the action defined by the action-type keyword. The default number of 5 averaged measurements can be changed using the number-of-measurements argument. The range is from 1 to 16.
- **threshold-type consecutive**[*occurrences*] —(Optional) When a threshold violation for the monitored element is met consecutively for a specified number of times, perform the action defined by the action-type keyword. The default number of 5 consecutive occurrences can be changed using the occurrences argument. The range is from 1 to 16.
- **threshold-type immediate**—(Optional) When a threshold violation for the monitored element is met, immediately perform the action defined by the **action-type** keyword.
- **threshold-value***upper-threshold lower-threshold*—(Optional) Specifies the upper-threshold and lower-threshold values of the applicable monitored elements.

Step 13 **ip sla logging traps**

Example:

```
Router(config)# ip sla logging traps
```

(Optional) Enables IP SLAs syslog messages from CISCO-RTTMON-MIB.

Step 14 **exit**

Example:

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

Exits global configuration mode and enters privileged EXEC mode.

What to do next

Once the SLM is configured, you have to schedule an IP SLA operation.

Scheduling an IP SLA Operation

To schedule an IP SLA operation, execute the following commands:

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables the privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters the global configuration mode.

Step 3 **ip sla schedule** *operation-number* [**life** { **forever** | *seconds* }] [**start-time** {*hh* **:** *mm* [**:** *ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh* **:** *mm* **:** *ss* | **random** *milliseconds*}]

Example:

```
Router(config)# ip sla schedule 10 start-time now life forever
```

Configures the scheduling parameters for an individual IP SLA operation or Specifies an IP SLA operation group number and the range of operation numbers to be scheduled for a multi-operation scheduler.

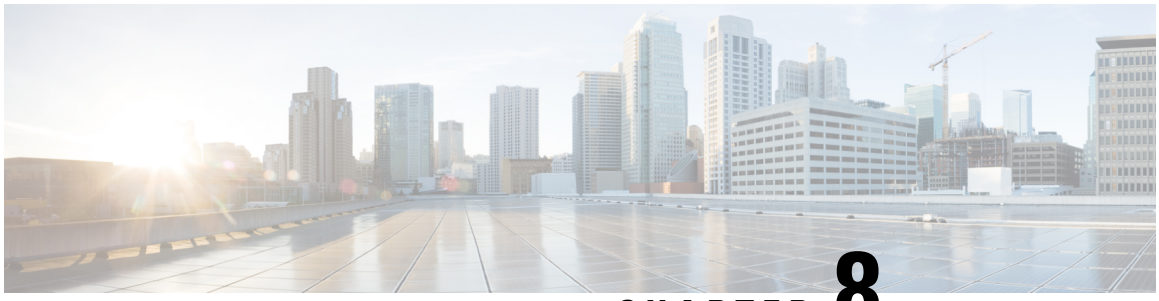
- *operation-number*—Identifies the IP SLAs operation for which reactions are to be configured.
- **life forever**— (Optional) Schedules the operation to run indefinitely.
- **life** *seconds* —(Optional) Number of seconds the operation actively collects information. The default is 3600 seconds (one hour).
- **start-time** —(Optional) Time when the operation starts.
- *hh***:***mm*[**:***ss*]—Specifies an absolute start time using hour, minute, and (optionally) second. Use the 24-hour clock notation. For example, start-time 01:02 means “start at 1:02 a.m.,” and start-time 13:01:30 means “start at 1:01 p.m. and 30 seconds.” The current day is implied unless you specify a month and day.
- *month* —(Optional) Name of the month to start the operation in. If month is not specified, the current month is used. Use of this argument requires that a day be specified. You can specify the month by using either the full English name or the first three letters of the month.
- *day* —(Optional) Number of the day (in the range 1 to 31) to start the operation on. If a day is not specified, the current day is used. Use of this argument requires that a month be specified.
- **pending** —(Optional) No information is collected. This is the default value.
- **now** —(Optional) Indicates that the operation should start immediately.
- **after** *hh***:***mm***:***ss*—(Optional) Indicates that the operation should start hh hours, mm minutes, and ss seconds after this command was entered.
- **random** *milliseconds*—(Optional) Adds a random number of milliseconds (between 0 and the specified value) to the current time, after which the operation will start. The range is from 0 to 10000.

Step 4 **exit**

Example:

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

Exits the global configuration mode and enters the privileged EXEC mode.



CHAPTER 8

Using Link Layer Discovery Protocol in Multivendor Networks

Link Layer Discovery Protocol (LLDP), standardized by the IEEE as part of 802.1ab, enables standardized discovery of nodes, which in turn facilitates future applications of standard management tools such as Simple Network Management Protocol (SNMP) in multivendor networks. Using standard management tools makes physical topology information available and helps network administrators detect and correct network malfunctions and inconsistencies in configuration.

Media Endpoint Discovery (MED) is an LLDP enhancement that was formalized by the Telecommunications Industry Association (TIA) for voice over IP (VoIP) applications.

The Cisco implementation of LLDP is based on the IEEE 802.1ab standard. This document describes LLDP and LLDP-MED and how they are supported in Cisco software.

Prerequisites for Using Link Layer Discovery Protocol in Multivendor Networks

- Type-Length-Value (TLV) types 0 through 127
- To support LLDP-MED, the following organizationally specific TLVs must be implemented:
  - Extended Power-via-Media Dependent Interface (MDI)
  - Inventory
  - LLDP-MED Capabilities
  - MAC/PHY Configuration Status
  - Network Policy
  - Port VLAN ID

Restrictions for Using Link Layer Discovery Protocol in Multivendor Networks

- Use of LLDP is limited to 802.1 media types such as Ethernet, Token Ring, and Fiber Distributed Data Interface (FDDI) networks.
- The maximum number of neighbor entries per chassis is limited on MED-capable network connectivity devices.

Information About Using Link Layer Discovery Protocol in Multivendor Networks

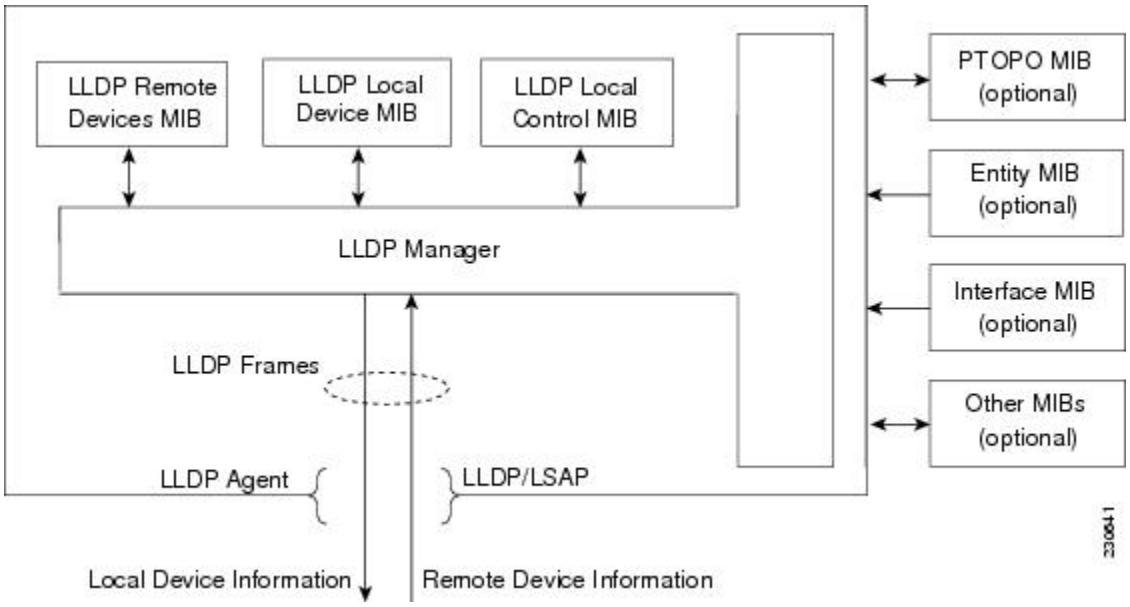
IEEE 802.1ab LLDP

IEEE 802.1ab Link Layer Discovery Protocol (LLDP) is an optional link layer protocol for network topology discovery in multivendor networks. Discovery information includes device identifiers, port identifiers, versions, and other details. As a protocol that aids network management, LLDP provides accurate network mapping, inventory data, and network troubleshooting information.

LLDP is unidirectional, operating only in an advertising mode. LLDP does not solicit information or monitor state changes between LLDP nodes. LLDP periodically sends advertisements to a constrained multicast address. Devices supporting LLDP can send information about themselves while they receive and record information about their neighbors. Additionally, devices can choose to turn off the send or receive functions independently. Advertisements are sent out and received on every active and enabled interface, allowing any device in a network to learn about all devices to which it is connected. Applications that use this information include network topology discovery, inventory management, emergency services, VLAN assignment, and inline power supply.

Note LLDP and Cisco Discovery Protocol can operate on the same interface.

The figure below shows a high-level view of LLDP operating in a network node.



When you configure LLDP or Cisco Discovery Protocol location information on a per-port basis, remote devices can send Cisco medianet location information to the switch. For more information, see the *Using Cisco Discovery Protocol module.*

LLDP-MED

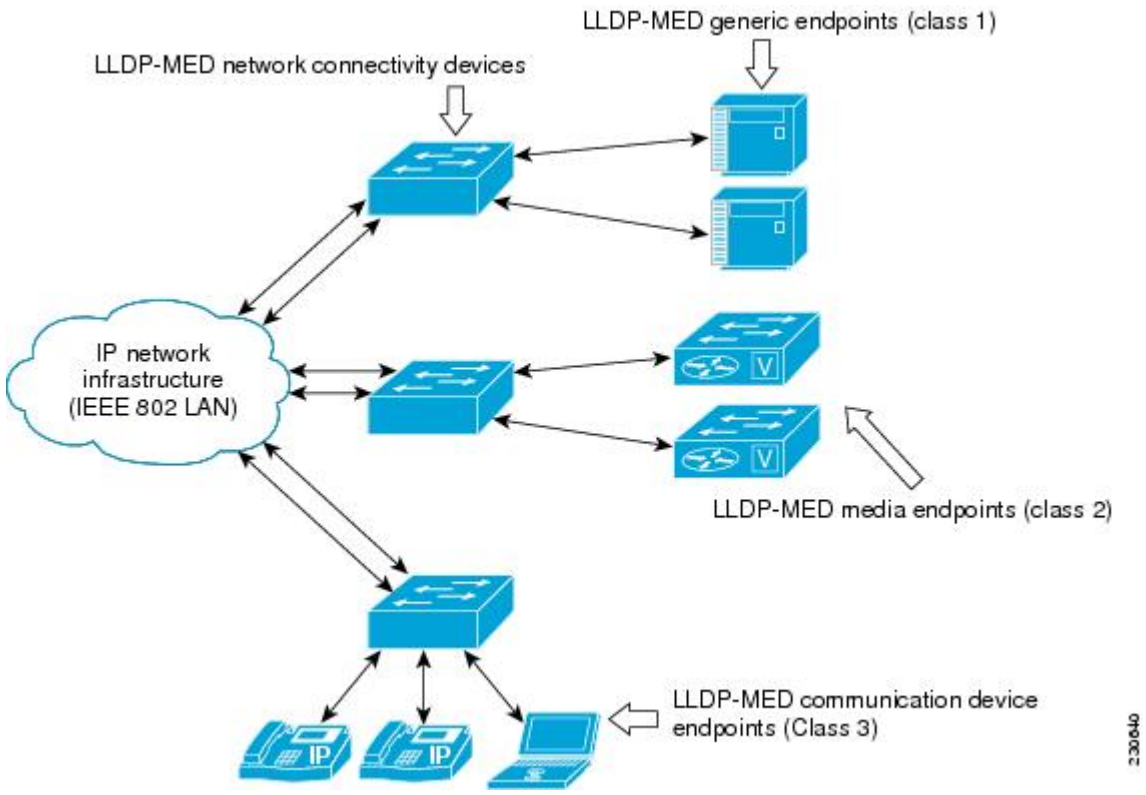
LLDP-MED operates between several classes of network equipment such as IP phones, conference bridges, and network connectivity devices such as routers and switches. By default, a network connectivity device sends out only LLDP packets until it receives LLDP-MED packets from an endpoint device. The network device then sends out LLDP-MED packets until the remote device to which it is connected ceases to be LLDP-MED capable.

Classes of Endpoints

LLDP-MED network connectivity devices provide IEEE 802 network access to LLDP-MED endpoints. LLDP-MED supports the following three classes of endpoints:

- Generic (class 1)—Basic participant endpoints; for example, IP communications controllers.
- Media (class 2)—Endpoints that support media streams; for example, media gateways and conference bridges.
- Communication Device (class 3)—Endpoints that support IP communications end users; for example, IP phones and Softphone.

The figure below shows an LLDP-MED-enabled LAN.



Types of Discovery Supported

LLDP-MED provides support to discover the following types of information, which are crucial to efficient operation and management of endpoint devices and the network devices supporting them:

- **Capabilities** —Endpoints determine the types of capabilities that a connected device supports and which ones are enabled.
- **Inventory** —LLDP-MED support exchange of hardware, software, and firmware versions, among other inventory details.
- **LAN speed and duplex** —Devices discover mismatches in speed and duplex settings.
- **Location identification** —An endpoint, particularly a telephone, learns its location from a network device. This location information may be used for location-based applications on the telephone and is important when emergency calls are placed.
- **Network policy** —Network connectivity devices notify telephones about the VLANs they should use.
- **Power** —Network connectivity devices and endpoints exchange power information. LLDP-MED provides information about how much power a device needs and how a device is powered. LLDP-MED also determines the priority of the device for receiving power.

Benefits of LLDP-MED

- Follows an open standard
- Supports E-911 emergency service, which is aided by location management

- Provides fast start capability
- Supports interoperability between multivendor devices
- Supports inventory management (location, version, etc.)
- Provides MIB support
- Supports plug and play installation
- Provides several troubleshooting (duplex, speed, network policy) mechanisms

TLV Elements

Link Layer Discovery Protocol (LLDP) and LLDP-Media Endpoint Discovery (MED) use Type-Length-Values (TLVs) to exchange information between network and endpoint devices. TLV elements are embedded in communications protocol advertisements and used for encoding optional information. The size of the type and length fields is fixed at 2 bytes. The size of the value field is variable. The type is a numeric code that indicates the type of field that this part of the message represents, and the length is the size of the value field, in bytes. The value field contains the data for this part of the message.

LLDP-MED supports the following TLVs:

- LLDP-MED capabilities TLV—Allows LLDP-MED endpoints to determine the capabilities that the connected device supports and has enabled.
- Network policy TLV—Allows both network connectivity devices and endpoints to advertise VLAN configurations and associated Layer 2 and Layer 3 attributes for the specific application on that port. For example, the switch can notify a phone of the VLAN number that it should use. The phone can connect to any switch, obtain its VLAN number, and then start communicating with the call control.

By defining a network-policy profile TLV, you can create a profile for voice and voice signalling by specifying the values for VLAN, class of service (CoS), differentiated services code point (DSCP), and tagging mode. These profile attributes are then maintained centrally on the switch and propagated to the phone.

- Power management TLV—Enables advanced power management between LLDP-MED endpoint and network connectivity devices. Allows switches and phones to convey power information, such as how the device is powered, power priority, and how much power the device needs. Supports advertisement of fractional wattage power requirements, endpoint power priority, and endpoint and network connectivity-device power status but does not provide for power negotiation between the endpoint and the network connectivity devices. When LLDP is enabled and power is applied to a port, the power TLV determines the actual power requirement of the endpoint device so that the system power budget can be adjusted accordingly. The switch processes the requests and either grants or denies power based on the current power budget. If the request is granted, the switch updates the power budget. If the request is denied, the switch turns off power to the port, generates a syslog message, and updates the power budget. If LLDP-MED is disabled or if the endpoint does not support the LLDP-MED power TLV, the initial allocation value is used throughout the duration of the connection.

Note A system power budget is the default power allocated to a device based on its device class. However, the total power that can be sourced from a switch is finite, and there will be some power budgeting done by the power module based on the number of ports already being served, total power that can be served, and how much new ports are requesting.

- Inventory management TLV—Allows an endpoint to send detailed inventory information about itself to the switch, including information hardware revision, firmware version, software version, serial number, manufacturer name, model name, and asset ID TLV.
- Location TLV—Provides location information from the switch to the endpoint device. The location TLV can send this information:
  - Civic location information—Provides the civic address information and postal information. Examples of civic location information are street address, road name, and postal community name information.
  - ELIN location information—Provides the location information of a caller. The location is determined by the Emergency location identifier number (ELIN), which is a phone number that routes an emergency call to the local public safety answering point (PSAP) and which the PSAP can use to call back the emergency caller.

Benefits of LLDP

- Follows IEEE 802.1ab standard.
- Enables interoperability among multivendor devices.
- Facilitates troubleshooting of enterprise networks and uses standard network management tools.
- Provides extension for applications such as VoIP.

How to Configure Link Layer Discovery Protocol in Multivendor Networks

Enabling and Disabling LLDP Globally

LLDP is disabled globally by default. This section describes the tasks for enabling and disabling LLDP globally.

Enabling LLDP Globally

Perform this task to enable LLDP globally.

Procedure

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 **lldp run**

Example:

```
Device(config)# lldp run
```

Enables LLDP globally.

Note To disable LLDP globally, use the **no lldp run** command.

Step 4 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Disabling LLDP Globally

Perform this task to disable LLDP 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 lldp run** **Example:** `Device(config)# no lldp run`	Disables LLDP globally.
Step 4	**end** **Example:**	Returns to privileged EXEC mode.

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

Disabling and Enabling LLDP on a Supported Interface

LLDP is enabled by default on all supported interfaces. This section describes the tasks for disabling and enabling LLDP on a supported interface.

Disabling LLDP on a Supported Interface

Perform this task to disable LLDP on a supported interface.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface Gigabitethernet 0/1
```

Specifies the interface type and number and enters interface configuration mode.

Step 4 **no lldp** {**med-tlv-select** *tlv* | **receive** | **transmit**}

Example:

```
Device(config-if)# no lldp receive
```

Disables an LLDP-MED TLV or LLDP packet reception on a supported interface.

Note To enable LLDP on a Supported Interface, use the **lldp** {**med-tlv-select** *tlv* | **receive** | **transmit** command.

Step 5 **end**

Example:

```
Device(config-if)# end
```

Returns to privileged EXEC mode.

Enabling LLDP on a Supported Interface

LLDP information can be transmitted and received only on an interface where LLDP is configured and enabled. Perform this task to enable LLDP.

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 ethernet 0/1`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**lldp** {**med-tlv-select** *tlv* | **receive** | **transmit**} **Example:** `Device(config-if)# lldp transmit`	Enables an LLDP-MED TLV or LLDP packet transmission on a supported interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Setting LLDP Packet Hold Time

Hold time is the duration that a receiving device should maintain LLDP neighbor information before aging it. Perform this task to define a hold time for an LLDP-enabled device.

Procedure

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 **lldp holdtime** *seconds*

Example:

```
Device(config)# lldp holdtime 100
```

Specifies the hold time.

Step 4 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Setting LLDP Packet Frequency

Perform this task to specify an interval at which the Cisco software sends LLDP updates to neighboring devices.

Procedure

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 **lldp timer** *rate*

Example:

```
Device(config)# lldp timer 75
```

Specifies the rate at which LLDP packets are sent every second.

Step 4 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Monitoring and Maintaining LLDP in Multivendor Networks

Perform this task to monitor and maintain LLDP in multivendor networks. This task is optional, and Steps 2 and 3 can be performed in any sequence.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show lldp** [**entry** {***** | *word*} | **errors** | **interface** [**ethernet** *number*]| **neighbors** [**ethernet** *number*| **detail**]| **traffic**]

Example:

```
Device# show lldp entry *
```

Displays summarized and detailed LLDP information.

Note When the **show lldp neighbors** command is issued, if the device ID has more than 20 characters, the ID is truncated to 20 characters in command output because of display constraints.

Step 3 **clear lldp** {**counters** | **table**}

Example:

```
Device# clear lldp counters
```

Resets LLDP traffic counters and tables to zero.

Step 4 **end**

Example:

```
Device# end
```

Returns to user EXEC mode.

Enabling and Disabling LLDP TLVs

LLDP TLV support is enabled by default if LLDP is enabled globally and locally on a supported interface. Specific TLVs, however, can be enabled and suppressed.

Enabling LLDP TLVs

Perform this task to enable an LLDP TLV on a supported 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 Gigabitethernet 0/1`	Specifies the interface type and number on which to enable LLDP-MED and enters interface configuration mode.
Step 4	**lldp tlv-select** *tlv* **Example:** `Device(config-if)# lldp tlv-select power-management`	Enables a specific LLDP TLV on a supported interface. **Note** To disable LLDP TLVs, use the **no lldp tlv-select** *tlv*
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Disabling LLDP TLVs

Perform this task to disable an LLDP TLV on a supported 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 ethernet 0/1`	Specifies the interface type and number on which to disable LLDP-MED and enters interface configuration mode.
Step 4	**no lldp tlv-select** *tlv* **Example:** `Device(config-if)# no lldp tlv-select system-description`	Disables a specific LLDP TLV on a supported interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Enabling and Disabling LLDP-MED TLVs

LLDP-MED TLV support is enabled by default if LLDP is enabled globally and locally on a supported interface. Specific TLVs, however, can be enabled and suppressed.

Enabling LLDP-MED TLVs

Perform this task to enable a specific LLDP-MED TLV on a supported interface.

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	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 0/1`	Specifies the interface type and number on which to enable LLDP-MED and enters interface configuration mode.
Step 4	**lldp med-tlv-select** *tlv* **Example:** `Device(config-if)# lldp med-tlv-select inventory-management`	Enables a specific LLDP-MED TLV on a supported interface. **Note** To disable LLDP-MED TLVs, use the **no lldp med-tlv-select** *tlv* command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Disabling LLDP-MED TLVs

Perform this task to disable a specific LLDP-MED TLV from a supported 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 ethernet 0/1`	Specifies the interface type and number on which to disable LLDP-MED and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**no lldp med-tlv-select** *tlv* **Example:** `Device(config-if)# no lldp med-tlv-select inventory-management`	Disables a specific LLDP-MED TLV from a supported interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuration Examples for Link Layer Discovery Protocol in Multivendor Networks

Example Configuring LLDP on Two Devices

The following example shows how to configure LLDP timer, hold time, and TLVs on two devices in a network. In each case we assume that the Ethernet interfaces being configured are in the UP state.

```
! Configure LLDP on Device 1 with hold time, timer, and TLV options.

Device1> enable
Device1# configure terminal
Device1(config)# lldp run
Device1(config)# lldp holdtime 150
Device1(config)# lldp timer 15
Device1(config)# lldp tlv-select port-vlan
Device1(config)# lldp tlv-select mac-phy-cfg
Device1(config)# interface ethernet 0/0
Device1(config-if)# end
00:08:32: %SYS-5-CONFIG_I: Configured from console by console
! Show the updated running configuration. LLDP is enabled with hold time, timer, and TLV
options configured.

Device1# show running-config

Building configuration...
Current configuration : 1397 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname Device1
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
```

```
clock timezone PST -8
ip subnet-zero
!
!
lldp timer 15
lldp holdtime 150
!


! Configure LLDP on Device 2 with hold time, timer, and TLV options.

Device2> enable
Device2# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device2(config)# lldp run
Device2(config)# lldp holdtime 150
Device2(config)# lldp timer 15
Device2(config)# lldp tlv-select port-vlan
Device2(config)# lldp tlv-select mac-phy-cfg
Device2(config)# interface ethernet 0/0
Device2(config-if)# end
00:08:32: %SYS-5-CONFIG_I: Configured from console by console

! Show the updated running configuration on Device 2. LLDP is enabled with hold time, timer,
 and TLV options configured.

Device2# show running-config
Building configuration...
Current configuration : 1412 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname R2
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
clock timezone PST -8
ip subnet-zero
!
!
lldp timer 15
lldp holdtime 150
!


! After both devices are configured for LLDP, issue the show
 command from each device to view traffic and device information.

Device1# show lldp traffic
LLDP traffic statistics:
    Total frames out: 20
    Total entries aged: 0
    Total frames in: 15
    Total frames received in error: 0
    Total frames discarded: 0
    Total TLVs unrecognized: 0
Device1# show lldp neighbors
Capability codes:
```

```
    (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
    (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
Device ID           Local Intf     Hold-time  Capability      Port ID
Device2             Et0/0          150        R               Et0/0
Total entries displayed: 1
Device2# show lldp traffic
LLDP traffic statistics:
    Total frames out: 15
    Total entries aged: 0
    Total frames in: 17
    Total frames received in error: 0
    Total frames discarded: 2
    Total TLVs unrecognized: 0
Device2# show lldp neighbors
Capability codes:
    (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
    (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
Device ID           Local Intf     Hold-time  Capability      Port ID
Device1             Et0/0          150        R               Et0/0
Total entries displayed: 1
```

Additional References for Using Link Layer Discovery Protocol in Multivendor Networks

Related Documents

Related Topic	Document Title
Cisco IOS commands: master list of commands with complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Master Command List, All Releases*
Carrier Ethernet commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Carrier Ethernet Command Reference*
LLDP	*Link Layer Discovery Protocol*
Per Port Location configurations	*Per Port Location Configuration*
Comparison of LLDP Media Endpoint Discovery (MED) and Cisco Discovery Protocol	*LLDP-MED and Cisco Discovery Protocol*

Standards and RFCs

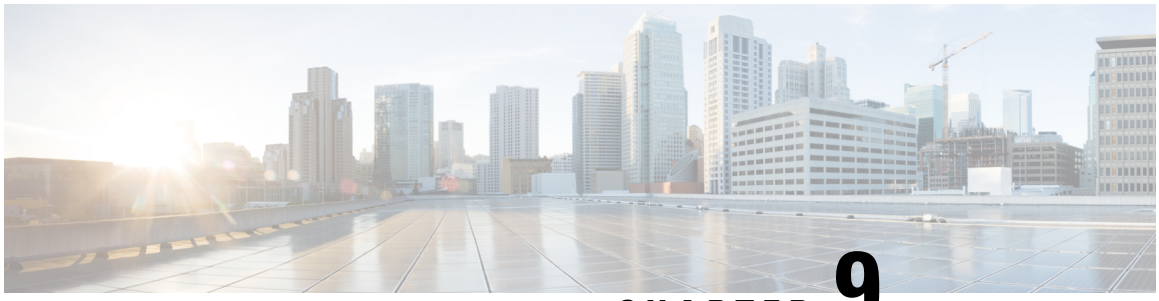
Standards/RFCs	Title
IEEE 802.1ab	*Station and Media Access Control Connectivity Discovery*
RFC 2922	Physical Topology MIB

MIBs

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

Technical Assistance

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



CHAPTER 9

Configuring Switched Port Analyzer

This document describes how to configure local Switched Port Analyzer (SPAN) and remote SPAN (RSPAN) on the router.

Prerequisites for Configuring Local SPAN and RSPAN

Local SPAN

- Use a network analyzer to monitor interfaces.

RSPAN

- Before configuring RSPAN sessions, you must first configure:
  1. Source interface
  2. Destination Bridge Domain over VPLS

Restrictions for Local Span and RSPAN

Local Span

- Local SPAN is only supported on physical ports.
- VLAN filtering is not supported.

- SPAN monitoring of port-channel interfaces or port-channel member-links is *not* supported.
- Combined Egress local SPAN bandwidth supported is 1 GB.
- Local SPAN is not supported on logical interfaces such as VLANs or EFPs.
- Only one local SPAN destination interface is supported. You *cannot* configure a local SPAN destination interface to receive ingress traffic.
- Outgoing Cisco Discovery Protocol (CDP) and Bridge Protocol Data Unit (BPDU) packets are not replicated.
- When enabled, local SPAN uses any previously entered configuration.
- When you specify source interfaces and do not specify a traffic direction (**Tx**, **Rx**, or **both**), **both** is used by default.
- Local SPAN destinations never participate in any spanning tree instance. Local SPAN includes BPDUs in the monitored traffic, so any BPDUs seen on the local SPAN destination are from the local SPAN source.
- Local SPAN sessions with overlapping sets of local SPAN source interfaces or VLANs are *not* supported.
- EFP/TEFP shut will not stop traffic flow.
- Only one Interface can be configured as monitor destination for Local SPAN.
- Only Rx (BPDU) control packets will be replicated in local SPAN.
- More than one interface as source will work in local SPAN.
- Monitor source interface can be part of one Local SPAN session only. If source interface is configured as SPAN destination in another session, the following error is prompted. Interfaces Gi 0/0/1 is already configured as monitor sources in session 1, hence rejecting the entire bundle of requests under this submode.
- Destination interface can be part of one Local Span session only. If same destination is programmed for two sessions, the following error will be prompted. Interfaces Gi0/0/0 already configured as monitor destinations in other monitor sessions.
- Once maximum scale is reached. You need to remove SPAN sessions and then reconfigure new session.
- Incoming packets will be mirrored based on packets on wire for LSPAN/RSPAN.
- Egress packets will be mirrored based on packets after applying rewrite / QoS on packets.
- Any change to EFP or interface configs, requires reconfigure LSPAN/RSPAN.
- Dynamic modification of SPAN/RSPAN is not supported.

RSPAN

- Only Rx is supported for RSPAN with Filter vlan, BD.
- If configuration change is done to EFP which is part of SPAN session, it will not stop traffic flow.
- Port channel RSPAN is not supported.
- Per member link RSPAN is not supported.

- VLAN filtering is supported.
- If two RSPAN configurations sessions are configured on two RSPAN BDs associated to the same Trunk EFP, the traffic from the first session flows to the second session after it is configured.
- RSPAN spans the Rx traffic even when the classifying service instance of the receiving port is in admin down state.
- EFP/TEFP shut will not stop traffic flow. RSPAN traffic does not Egress out of both the EFP attached to BD in same interface.
- Multiple source ports are not supported for RSPAN. Port-range not supported for RSPAN
- Filtering option will be supported only in interface mode.
- Filtering will be supported only on single and double encapsulation.
- Once maximum scale is reached. You need to remove RSPAN sessions and then reconfigure new session.
- RSPAN will mirror all BD traffic configured on TEFP.
- Filter option will not work for default and untagged.
- Incoming packets will be mirrored based on packets on wire for LSPAN/RSPAN.
- Egress packets will be mirrored based on packets after applying rewrite / QoS on packets.
- Any change to EFP or interface configs, requires reconfigure LSPAN/RSPAN.
- Dynamic modification of SPAN/RSPAN is not yet supported.

Note Incomplete configuration of RSPAN / LSPAN will result in traffic drop issues.

Scale Support for Port Mirroring

In total 8 logical ports (4 for Ingress and 4 for Egress) are assigned in Broadcom for Mirroring (including Local and Remote SPAN).

For Local SPAN, one logical port is assigned for Ingress mirroring and one logical port is assigned for Egress mirroring. In case both are selected then 2 logical ports are assigned.

For Remote SPAN, 2 logical ports are assigned for Ingress mirroring and 2 logical port is assigned for Egress mirroring.

Maximum scale depends on consumption of these logical ports.

Understanding Local SPAN and RSPAN

Information About Local SPAN Session and RSPAN Session

Local SPAN Session

A local Switched Port Analyzer (SPAN) session is an association of a destination interface with a set of source interfaces. You can configure local SPAN sessions to monitor all traffic in a specified direction. Local SPAN sessions allow you to monitor traffic on one or more interfaces and to send either ingress traffic, egress traffic, or both to one destination interface.

Local SPAN sessions do not interfere with the normal operation of the switch. You can enable or disable SPAN sessions with command-line interface (CLI) commands. When enabled, a local SPAN session might become active or inactive based on various events or actions, and this would be indicated by a syslog message. The **show monitor session span session number** command displays the operational status of a SPAN session.

A local SPAN session remains inactive after system power-up until the destination interface is operational.

The following configuration guidelines apply when configuring local SPAN on the router:

- When enabled, local SPAN uses any previously entered configuration.
- Use the **no monitor session** *session number* command with no other parameters to clear the local SPAN session number.

Local SPAN Traffic

Network traffic, including multicast, can be monitored using SPAN. Multicast packet monitoring is enabled by default. In some SPAN configurations, multiple copies of the same source packet are sent to the SPAN destination interface. For example, a bidirectional (both ingress and egress) SPAN session is configured for sources a1 and a2 to a destination interface d1. If a packet enters the switch through a1 and gets switched to a2, both incoming and outgoing packets are sent to destination interface d1; both packets would be the same (unless a Layer-3 rewrite had occurred, in which case the packets would be different).

RSPAN Session

An RSPAN source session is an association of source ports or VLAN across your network with an RSPAN Vlan. The RSPAN VLAN/BD on the router is the destination RSPAN session.

RSPAN Traffic for RSP2 Module

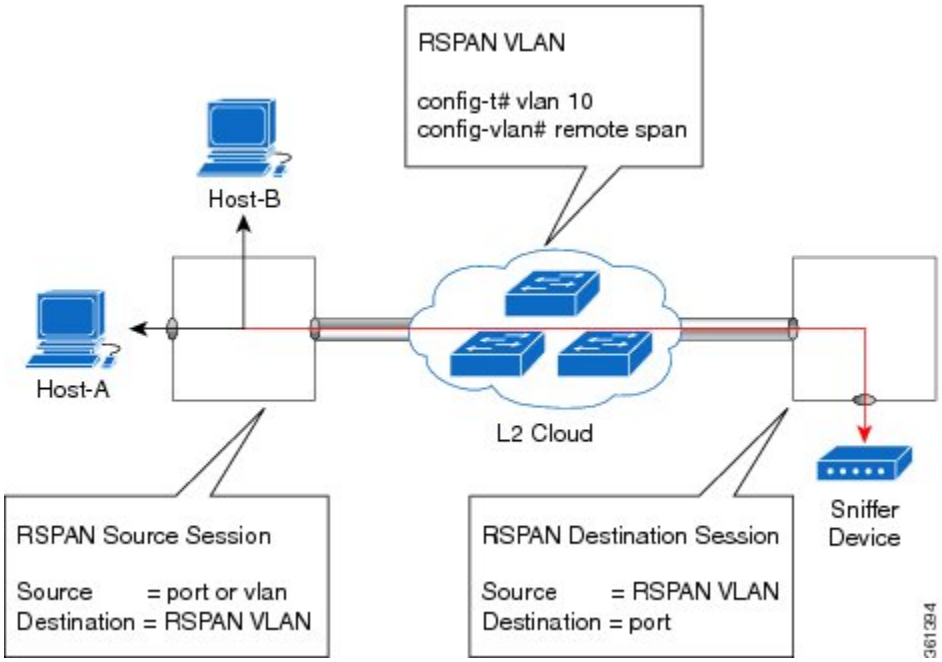
RSPAN supports source ports and source VLANs in the source switch and destination as RSPAN VLAN/BD.

The figure below shows the original traffic from the Host A to Host B via the source ports or VLANs on Host A. The source ports or VLANs of Host A is mirrored to Host B using RSPAN VLAN 10. The traffic for each RSPAN session is carried over a user-specified RSPAN VLAN that is dedicated for that RSPAN session in all participating devices. The traffic from the source ports or VLANs are mirrored into the RSPAN VLAN

and forwarded over Trunk or the EVC bridge domain (BD) ports carrying the RSPAN VLAN to a destination session monitoring the RSPAN VLAN.

Each RSPAN source must have either ports or VLANs as RSPAN sources. On RSPAN destination, the RSPAN VLAN is monitored and mirrored to the destination physical port connected to the sniffer device.

Figure 5: RSPAN Traffic



RSPAN allows remote monitoring of traffic where the source and destination switches are connected by L2VPN networks

The RSPAN source is either ports or VLANs as in a traditional RSPAN. However, the SPAN source and destination devices are connected through a L2 pseudowire associated with the RSPAN VLAN over an MPLS/IP network. The L2 pseudowire is dedicated for only RSPAN traffic. The mirrored traffic from the source port or VLAN is carried over the pseudowire associated with the RSPAN VLAN towards the destination side. On the destination side, a port belonging to the RSPAN VLAN or EVC BD is connected to sniffer device.

Destination Interface

A destination interface, also called a monitor interface, is a switched interface to which SPAN or RSPAN sends packets for analysis. You can have only one destination interface for SPAN sessions.

An interface configured as a destination interface cannot be configured as a source interface. Specifying a trunk interface as a SPAN or RSPAN destination interface stops trunking on the interface.

Source Interface

A source interface is an interface monitored for network traffic analysis. An interface configured as a destination interface cannot be configured as a source interface.

Configuring Local SPAN and RSPAN

Configuring Sources and Destinations for Local SPAN

To configure sources and destinations for a SPAN session:

Procedure

Step 1 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 2 **monitor session {***session_number***} type local**

Example:

```
Router(config)# monitor session 1 type local
```

Specifies the local SPAN session number and enters the local monitoring configuration mode.

- *session_number*—Indicates the monitor session. The valid range is 1 through 14.

Step 3 **source interface** *interface_type slot/subslot/port* **[, | - | rx | tx | both]**

Example:

```
Router(config-mon-local)# source interface gigabitethernet 0/2/1 rx
```

Specifies the source interface and the traffic direction:

- *interface_type*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface.
  - *slot/subslot/port*—The location of the interface.
- “,”—List of interfaces
- “–”—Range of interfaces
- rx—Ingress local SPAN
- tx—Egress local SPAN
- both

Step 4 **destination interface** *interface_type slot/subslot/port* **[, | -]**

Example:

```
Router(config-mon-local)# destination interface gigabitethernet 0/2/4
```

Specifies the destination interface that sends both ingress and egress local spanned traffic from source port to the prober or sniffer.

- *interface_type*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface.

- *slot/subslot/port*—The location of the interface.

- ","—List of interfaces
- "–"—Range of interfaces

Step 5 **no shutdown**

Example:

```
Router(config-mon-local)# no shutdown
```

Enables the local SPAN session.

Step 6 **End**

Removing Sources or Destinations from a Local SPAN Session

To remove sources or destinations from a local SPAN session, use the following commands beginning in EXEC mode:

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **no monitor session** *session-number*

Example:

```
Router(config)# no monitor session 2
```

Clears existing SPAN configuration for a session.

Configuring RSPAN Source Session

To configure the source for a RSPAN session:

Procedure

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 **monitor session** *RSPAN_source_session_number* **type rspan-source**

Example:

```
Router(config)# monitor session 1
 type rspan-source
```

Configures an RSPAN source session number and enters RSPAN source session configuration mode for the session.

- *RSPAN_source_session_number*—

  Valid sessions are 1 to 14.
- **rspan-source**—Enters the RSPAN source-session configuration mode.

Step 4 **Filter vlan***vlan id*

Example:

```
filter vlan 100
```

Applies the VLAN access map to the VLAN ID; valid values are from 1 to 4094.

Step 5 **source** {*single_interface* slot/subslot/port| *single_vlan* [**rx** | **tx** | **both**]

Example:

```
Router(config-mon-rspan-src)# source interface gigabitethernet 0/2/1 tx
```

Specifies the RSPAN session number, the source interfaces and the traffic direction to be monitored.

- *single_interface*—Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface.
  - *slot/subslot/port*—The location of the interface.
- *single_vlan*

  —Specifies the single VLAN.
- **both**

  —(Optional) Monitors the received and the transmitted traffic.

- **rx**

  —(Optional) Monitors the received traffic only.
- **tx**—(Optional) Monitors the transmitted traffic only.

Step 6 **no shutdown**

Example:

```
Router(config-mon-rspan-src)# no shutdown
```

Enables RSPAN source.

Step 7 **end**

Example:

```
Router(config-mon-rspan-src)# end
```

Exists the configuration.

Configuring RSPAN Destination Session

To configure the destination for a RSPAN session for remote Vlan:

Procedure

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 **monitor session** *RSPAN_destination_session_number* **type rspan-destination**

Example:

```
Router(config)# monitor session 1 type rspan-destination
```

Configures a RPAN session.

- *RSPAN_destination_session_number*—Valid sessions are 1 to 80.
- **rspan-destination**—Enters the RSPAN destination-session configuration mode.

Step 4 **source remote vlan** *rspan_vlan_ID*

Example:

```
Router(config-mon-rspan-dst)# source remote vlan2
```

Associates the RSPAN destination session number RSPAN VLAN.

- *rspan_vlan_ID*—Specifies the Vlan ID

Step 5 **destination** {*single_interface slot/subslot/port}*

Example:

```
Router(config-mon-rspan-dst)# destination interface gigabitethernet 0/0/1
```

Associates the RSPAN destination session number with the destination port.

- *single_interface* —Specifies the Gigabit Ethernet or Ten Gigabit Ethernet interface.
  - *slot/subslot/port*—The location of the interface.

Step 6 **no shutdown**

Example:

```
Router(config-mon-rspan-dst)# no shutdown
```

Restarts the interface

Step 7 **end**

Example:

```
Router(config-mon-rspan-dst)# end
```

Exists the configuration

Removing Sources or Destinations from a RSPAN Session

To remove source or destination from a RSPAN session, delete and recreate the RSPAN session. The following are the steps:

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **no monitor session** *session number*

Example:

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

Exits monitor session.

Step 4 **end**

Example:

```
Router(config-mon-rspan-src)# end
```

Exits configuration mode.

Sample Configurations

The following sections contain configuration example for SPAN and RSPAN on the router.

Configuration Example: Local SPAN

The following example shows how to configure local SPAN session 8 to monitor bidirectional traffic from source interface Gigabit Ethernet interface to destination:

```
Router(config)# monitor session 8 type local
Router(config)# source interface gigabitethernet 0/0/10
Router(config)# destination interface gigabitethernet 0/0/3
Router(config)# no shut
```

Configuration Example: Removing Sources or Destinations from a Local SPAN Session

This following example shows how to remove a local SPAN session:

```
Router(config)# no monitor session 8
```

Configuration Example: RSPAN Source

The following example shows how RSPAN session 2 to monitor bidirectional traffic from source interface Gigabit Ethernet 0/0/1:

```
Router(config)# monitor session 2 type RSPAN-source
Router(config-mon-RSPAN-src)# source interface gigabitEthernet0/0/1 [tx |rx|both]
Router(config-mon-RSPAN-src)# destination remote VLAN 100
Router(config-mon-RSPAN-src)# no shutdown
Router(config-mon-RSPAN-src)# end
```

The following example shows how RSPAN session 3 to monitor bidirectional traffic from source Vlan 200:

```
Router(config)# monitor session 3 type RSPAN-source
Router(config-mon-RSPAN-src)# filter vlan 100
Router(config-mon-RSPAN-src)# source interface Te0/0/23 rx
Router(config-mon-RSPAN-src)# destination remote VLAN 200
Router(config-mon-RSPAN-src)# no shutdown
Router(config-mon-RSPAN-src)# end
```

Configuration Example: RSPAN Destination

The following example shows how to configure interface Gigabit Ethernet 0/0/1 as the destination for RSPAN session 2:

```
Router(config)# monitor session 2 type RSPAN-destination
Router(config-mon-RSPAN-dst)# source remote VLAN 100
Router(config-mon-RSPAN-dst)# destination interface gigabitEthernet 0/0/1
Router(config-mon-RSPAN-dst)# end
```

Verifying Local SPAN and RSPAN

Use the **show monitor session** command to view the sessions configured.

- The following example shows the Local SPAN source session with Tx as source:

```
Router# show monitor session 8
Session 8
---------
Type : Local Session
Status : Admin Enabled
Source Ports :
TX Only : Gi0/0/10
Destination Ports : Gi0/0/3
MTU : 1464
Dest RSPAN VLAN : 100
```

- The following example shows the RSPAN source session with Gigabit Ethernet interface 0/0/1 as source:

```
Router# show monitor session 2
Session 2
---------
Type                  : Remote Source Session
Status                : Admin Enabled
Source Ports          :
    Both              : Gi0/0/1
MTU                   : 1464
```

- The following example shows the RSPAN source session with Vlan 20 as source:

```
Router# show monitor session 3
Session 3
---------
Type                  : Remote Source Session
Status                : Admin Enabled
Source VLANs          :
    RX Only           : 20
MTU                   : 1464
```

- The following example shows the RSPAN destination session with Gigabit Ethernet interface 0/0/1 as destination:

```
Router# show monitor session 2
Session 2
---------
Type                  : Remote Destination Session
Status                : Admin Enabled
Destination Ports     : Gi0/0/1
MTU                   : 1464
Source RSPAN VLAN : 100
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/mcl/allreleasemcl/all-book.html

Standards and RFCs

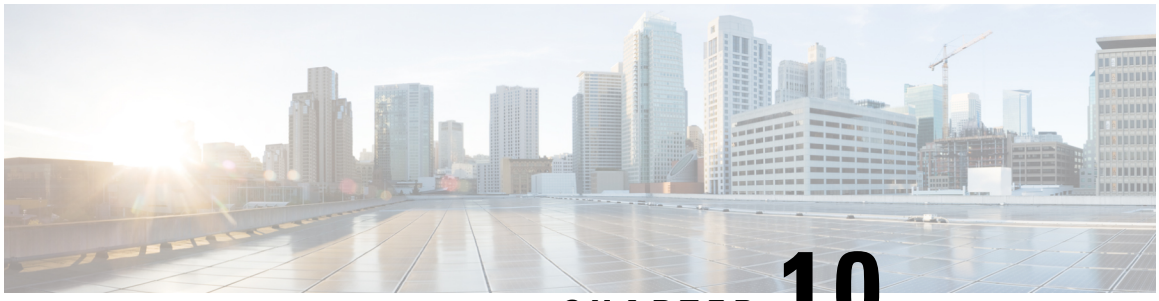
Standard/RFC	Title
No specific Standards and RFCs are supported by the features in this document.	—

MIBs

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

Technical Assistance

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



CHAPTER 10

MAC Limiting

This document describes how to configure MAC limiting.

Information About Global MAC Address Limiting on Bridge Domain

Table 4: Feature History

Feature Name	Release Information	Description
Mac Address Limiting Per Bridge Domain	Cisco IOS XE Bengaluru 17.4.1	This feature restricts the number of MAC addresses that the router learns in a bridge-domain on an EFP or trunk EFP to a specified number. Use the feature to enable warning and limit actions when a violation occurs.
MAC Entry Flooding Limiting	Cisco IOS XE Cupertino 17.8.1	This feature allows the user to disable unknown unicast flooding on a certain bdomain using the flood sub action. This flood sub action is initiated only when the limit action is configured and violation has occurred.

MAC address limiting per bridge-domain restricts the number of MAC addresses that the router learns in a bridge-domain on an EFP or trunk EFP to a specified number.

Note Use the **mac-address-table limit bdomain** *num* **maximum** *0* **action limit** command to disable mac-learning on bridge-domain.

When the total number of MAC addresses (dynamic MAC addresses) in a bridge-domain exceeds the maximum number, then the router takes a violation action. The router either restricts further learning on bridge-domain by itself with a syslog or just intimate the user through a syslog to take further action.

You can enable the following actions when violation occurs:

- Warning—The violation is logged as a syslog message and no further action is taken. There is one syslog message received, when the MAC count exceeds the configured limit (exceed notification) and no more syslog messages are received for the bridge-domain (bdomain) unless the violation is no longer valid (drop notification). When you select the warning action, the further learning of new MAC addresses and forwarding of traffic continue to happen irrespective of violation.
- Limit—When the Limit option is selected as an action for violation, the MAC learning on the bdomain is disabled when violation occurs. No new MAC addresses are learnt on the bdomain until the recovery mechanism gets started. Even though new MAC addresses are not learned but frames are still flooded in the system. If user needs to stop flooding, then a sub action flood can also be used along with limit action.

Note The threshold value must be 80% of the maximum value configured for the recovery mechanism.

- Flood—The flood sub action allows the user to disable unknown unicast flooding on a given bdomain. This flood sub action is initiated only when the limit action is configured and violation has occurred. Unknown unicast flooding is disabled only for the interval necessary to limit the entries. Using this option, improves the performance and the flooding is re-enabled when the total number of MAC entries are dropped below the threshold value.

Note **Warning** is the default action when no action is configured.

For the limit and warning actions, the recovery mechanism is initiated when the total MAC limit count drops to equal or below a threshold value. The threshold value is dependent on the maximum limit configured on bridge domain (the threshold value is 80% of the limit value). The recovery mechanism reverts the action taken during violation. For example, if the MAC address learning is disabled as a violation action, then it will be re-enabled.

If no maximum value or action option in specified through the **mac address-table limit bdomain id maximum num action** command, then the default action (warning) and a default maximum value of 500 is configured.

Note For a MAC limit of 0 with the action limit, limit flood, the violation action occurs when the user configures it irrespective of MAC address learning on the bridge domain. The recovery mechanism is to disable the feature through the **no mac address-table limit bdomain id** command.

Restrictions for MAC Limiting

MAC limiting is supported on the following interface types:

- You can apply MAC limiting only to bridge-domains.
- MAC limiting is supported for dynamic MAC addresses.
- With **limit** keyword, the router sends a syslog message and generates a trap. MAC learning is disabled on the bridge-domain. The flooding of frames with new MAC address continues. To disable flooding, use the **flood** keyword. Flooding resumes only after the total number of MAC entries drop below the threshold value (80%).
- The allowed MAC limit range is from 0 through 16000.
- You can enable the MAC limit feature up to a maximum of 1000 bridge domains.

Configuring MAC Limiting

Procedure

Step 1 **configure terminal**

Enter global configuration mode.

Step 2 **mac-address-table limit** *bdomain id maximum num action {warning | limit} [flood]*

Sets the specific limit and any optional actions to be imposed at the bridge-domain level.

The default **maximum** value is 500.

Step 3 **end**

Return to privileged EXEC mode.

Step 4 **show mac-address-table limit bdomain** *bdomain id*

Displays the information about the MAC-address table.

Step 5 **copy running-config startup-config**

(Optional) Save your entries in the configuration file.

Example of Enabling Per-Bridge-Domain MAC Limiting

This example shows how to enable per-bridge-domain MAC limiting.

```
Router# enable
Router# configure terminal
Router(config)# mac-address-table limit bdomain 10 maximum 100 action limit flood
Router(config)# end
```

Verifying the MAC Limiting on Bridge Domain

Use the **show mac address-table limit** command to verify the information related to configured MAC limit per bridge domain.

This example shows how to display the information related to configured MAC limit per bridge domain.

```
Router#show mac address-table limit bdomain 10
  bdomain       action      flood       maximum     Total entries     Current state
-------------+----------+-----------+-----------+---------------+---------------
   10           limit      Disable        100            0             Within Limit
```